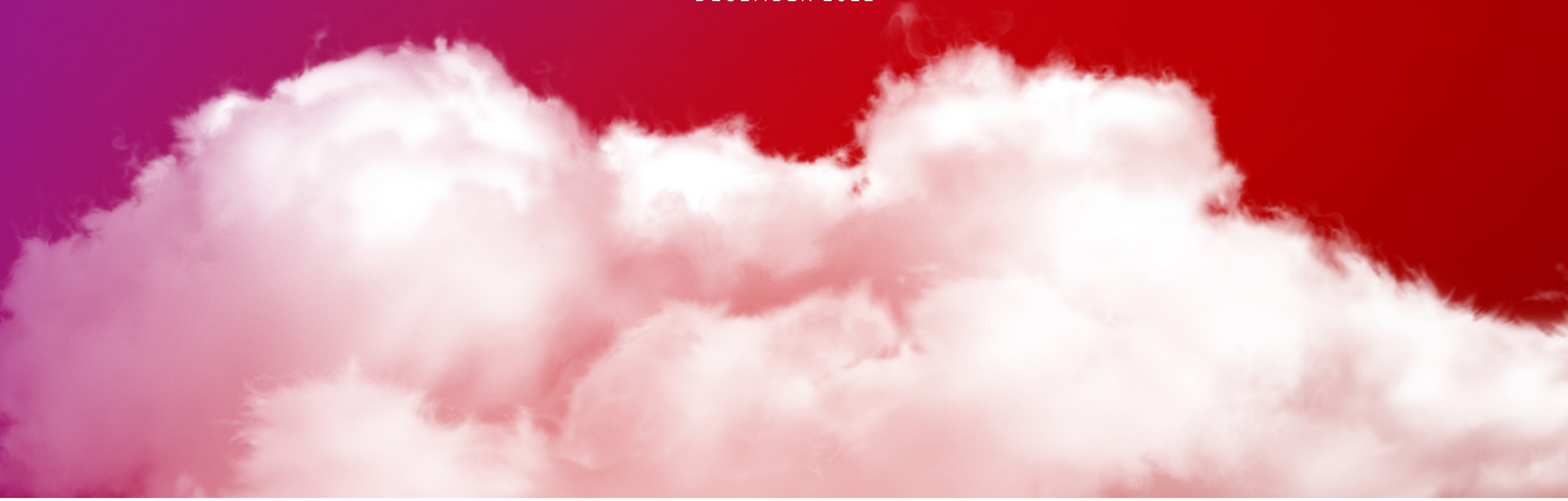
THE DEFINITIVE GUIDE TO

Telco Cloud Security

DECEMBER 2022



Rakuten Symphony

Ready. Set. Race.

Hacking is a race. As a company, you do not get to choose when a hacker starts racing you, but you can choose how well you prepare.

To secure 5G networks, your most important preparation revolves around design and implementation choices for new technologies, including virtualization, containerization and orchestration. These technologies, while new to telecom networks, are well-proven in other industries. It is our mission to learn from these industries and secure the future of telecom.

The design and configuration choices presented by modern network and cloud technologies open a wide spectrum of possible outcomes. This can increase security risk compared to previous network generations. However, when configured according to best practices, cloud native 5G and Open RAN networks can become the most secure mobile network generation deployed to date.

In this guide, we detail a five-step path to making strong choices around decoupling access, enforcing segregation, patching and hardening, verifying deployments, and hacking monitoring.

We also share Rakuten's experience implementing the five-step 5G security path in its work with Rakuten Mobile in Japan.

“ This (new network architectures) can increase security risk compared to previous network generations. However, when configured according to best practices, cloud native 5G and Open RAN networks can become the most secure mobile network generation deployed to date.”

How to implement proven cloud security strategies in modern telecom networks

Prepared by the Rakuten Mobile & Rakuten Symphony Expert Security Group

John Carse, CISO

Table of Contents

1 | Introduction

"The Definitive Guide to Telco Cloud Security" explores the main hacking vulnerabilities of modern mobile networks, and provides guidance on how to mitigate and manage these risks.

5G and Open RAN introduce new technologies that change potential hacking risk exposure. Specifically, the cloud technologies introduced in modern 5G networks provide design and configuration choices that offer a spectrum of security challenges if not managed correctly.

When configured weakly, these technologies can increase hacking exposure and hacking impact compared to previous network generations. However, when configured according to best practices, 5G and Open RAN networks can become the most secure mobile network generation deployed to date.

The good news is that we already know the critical security configuration settings required to make cloud-native 5G and Open RAN networks secure. This guide summarizes proven best practices from other industries that have successfully adopted open cloud technologies. It introduces a five-step path for securing the IT infrastructure of a 5G or Open RAN network and shares Rakuten Symphony's experience implementing these steps.

We will start by contrasting a well-secured network with one that does not pay sufficient attention to security design and hardening.

"We already know the critical security configuration settings required to make cloud-native 5G and Open RAN networks secure."

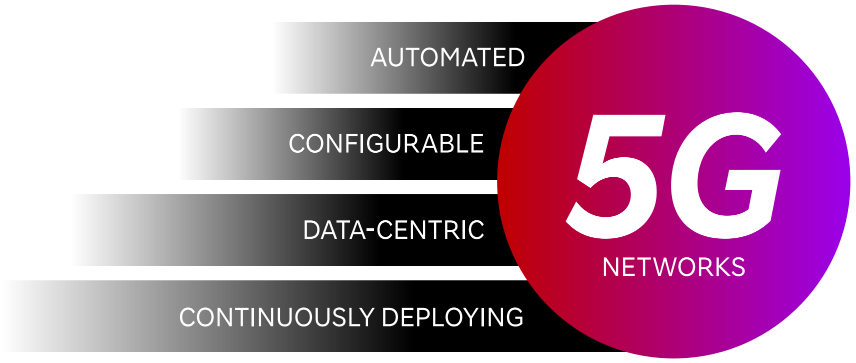
2 | Weak vs. strong 5G network design

The evolution to 5G enables faster and more flexible mobile networks. Some of these networks will excel in information security, while others will introduce additional risks compared to earlier mobile network generations. The difference between weak and strong 5G networks mostly comes down to design and configuration choices, as the remainder of this study illustrates.

Let us compare weak vs. strong choices when implementing four design principles found in most modern 5G networks, specifically in Open RAN networks.

Modern 5G networks are:

1. Automated
2. Flexibly configurable
3. Data-centric
4. Continuously deploying

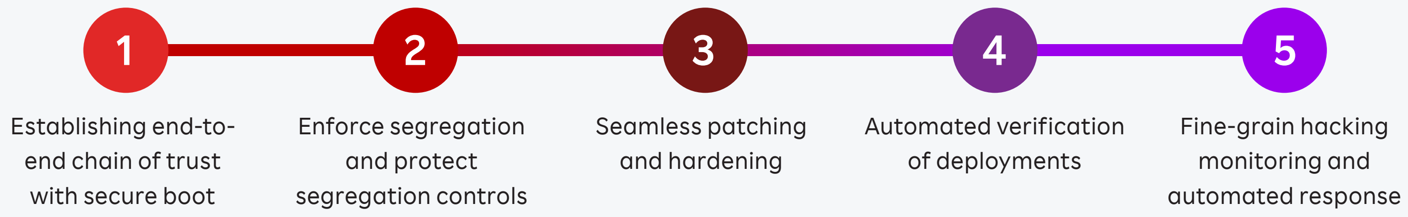


Each of the four design principles brings about new technology design choices and configuration settings. In each, weak choices can diminish the hacking resistance of a mobile network, while strong choices can strengthen a network compared to earlier technology generations:

Security down-side created by bad choices	5G network design principles	Up-side created by strong choices (five-step path)
Overall, more technology components for the hacker to target	Automated	1. Network access is decoupled from human operators
Changes can disrupt network operations more easily	Flexibly configurable	2. Fine-grained segregation contains risks
Disruptive changes can be caused from more places	Continuously deployment with choice of strategy (canary, rolling, immutable, …)	3. Patching and hardening happens seamless and continuous 4. Deployments are automatically security-tested
There is more activity for hackers to hide in	Data-centric	5. Fine-grained hacking monitoring enables automated hacking response

3 | Five-step path to securing 5G deployments

Each 5G deployment chooses between weak and strong design and configuration options. This chapter details a five-step path to making strong choices, while the next chapter shares Rakuten Symphony's experience and implementation challenges in following the five-step path.


1
Establishing end-to-end chain of trust with secure boot
2
Enforce segregation and protect segregation controls
3
Seamless patching and hardening
4
Automated verification of deployments
5
Fine-grain hacking monitoring and automated response

3.1 | Establishing end-to-end chain of trust with secure boot

The human factor has been a security sore point in all technology generations. With cloud-based 5G networks, we have an opportunity to decouple human access from technical infrastructure.

Cloud infrastructure is configured through scripts ("infrastructure as code") rather than through human operators. Humans ultimately maintain these scripts, but security checks can happen between the creation and execution of each script. When done right, decoupling limits both the impact of human errors and the effects of social engineering.

In addition, network built from code can recover much faster after an incident.

3.2 | Enforce segregation and protect segregation controls

Network segregation, especially of critical components, limits the possible reach of a hacking incident. The technology underpinning 5G networks makes segregation both easier and harder.

Segregation in cloud-native networks is easier because software-defined networks allow any granularity of segregation simply through configuration, and can flexibly be adopted to evolving functional requirements. No firewalls are required.

It is also harder to implement. New connections between network elements arise from virtualization/containerization whereas in earlier network generations, the only connection between elements was a network cable. Today, different elements typically reside as containers in the same cluster.

For segregation to be effective, containers need to be configured strongly to prevent hackers from being able to escape. Strong container configuration can be enforced centrally through automated tools in the deployment pipeline.

"For cloud-native networks, it is best practice to maintain "golden" master images with the latest patches and configuration settings, and to base all containers on these images."

3.3 | Seamless patching and hardening

Regular patching and strong configuration settings (known as 'hardening') remain critical in shortening the time window during which network elements are exploitable through newly-discovered security vulnerabilities.

Once again, cloud-native 5G makes patching both easier and harder compared to earlier network generations. Easier, because network elements can be rebuilt on the fly through an automated deployment pipeline, each time including the latest patches and configuration settings. Harder, because virtually all patching must be done through these automated processes due to the larger scale and complexity of 5G compared to earlier generations.

For cloud-native networks, it is best practice to maintain "golden" master images with the latest patches and configuration settings, and to base all containers on these images. A packer automatically builds network elements by combining a golden image with element-specific software, and stores the images in a repository. Automated deployment processes use these artifacts to deploy any number of required network elements, each including the latest security baseline.

3.4 | Automated verification of deployments

Security testing happens as part of the development and deployment pipelines. Yet again, cloud-native networks require much faster assurance processes to match their continuous deployment practices.

Tests must be fully automated as part of deployment pipelines and should uncover programming issues, patching gaps, and configuration issues through vulnerability assessment tools and source code scanners.

Feedback to developers and DevOps engineers should be immediate and automated by blocking issues from being committed into the pipeline.

3.5 | Fine-grain hacking monitoring and automated response

Not all hacking attacks can be fully prevented, creating a need for security monitoring to detect hackers as they move through the network.

Modern technology produces a lot of telemetry, which—when analyzed correctly—provides plenty of opportunity to detect and alert on hacking activity.

To respond at the speed of hackers, some evasion actions need to be automated. For example, suspicious instances should be saved for forensics purposes while replacements are immediately rebuilt with the latest security baseline. While the suspicious instance is being analyzed, other instances automatically take over the load. Speeding up recovery while at the same time resolving the inherent conflict of objectives between forensics and business continuity is another strength of containerized deployments.

3.6 | Baseline controls

The five-step path leads to more secure networks but relies on additional baseline security controls that are commonly found across IT infrastructures. As these standards controls are widely documented elsewhere, including in common security certifications, we mention them only briefly. The following controls are required for securely operating any complex IT infrastructure:

- User and access management
- Secure software development
- Third-party / supply-chain security
- Physical security
- Disaster recovery

Modern technology produces a lot of telemetry, which—when analyzed correctly—provides plenty of opportunity to detect and alert on hacking activity.

“The Rakuten telco environment, typical for a cloud-native 5G network, comes at unprecedented scale: We run thousands of containers in hundreds of clusters spread across the country.”

4 | Rakuten Symphony 5G security journey

At Rakuten, we implemented the five-step 5G security path. This chapter shares our experience, provides concrete advice for similar deployments, and details non-trivial complications with some controls.

4.1 | Strict orchestration

The Rakuten telco environment, typical for a cloud-native 5G network, comes at unprecedented scale: We run thousands of containers in hundreds of clusters spread across the country.

At this scale, deployment, management, and monitoring are only possible with complete automation.

In our orchestrated environment, users leverage automated workflows for deployment into hundreds of clusters without any manual intervention. Post-deployment workflows such as monitoring, auto healing, upgrades are also automated so that the environment can be managed with a lean operations team.

Let us look at the example of deploying a DU. DUs – or Distributed Units – are controllers for one or more antennas. We have thousands of these.

Deploying a DU involves BIOS configuration, OS installation, OS configuration, Kubernetes installation and configuration, Rakuten Symworld™ installation and configuration, and application deployment and configuration.

Before heavy automation, we had to give access to the hardware, OS, Kubernetes, and Symworld Platform to different human operators. Now with end-to-end orchestration in place, no human operator needs to have access to any of these components. This automation scales the effectiveness of operations exponentially while at the same time reducing the deployment time and improving the security of the platform by keeping access credentials tightly managed.

4.2 | Segregated multi-vendor environment

In Rakuten, as in any other telco environment, a lot of vendor applications need to be deployed in various stages of the life cycle (sandbox, staging, production). The applications need to run efficiently without causing noisy-neighbor issues, and without enabling hackers to laterally move between them.

We implemented four levels of isolation to allow applications to co-exist on the same cluster.

- **Network isolation:** Applications should not be able to see other applications' traffic. In Kubernetes, this is implemented using Network Policies and Istio. 5G environments complicate the policy definition, however, as each Kubernetes pod has different interfaces for different types of traffic (midhaul, backhaul, fronthaul) and network policies only work on the primary interface. To extend the policy onto non-primary interfaces, we isolate the traffic on these interfaces using VLANs. VLAN isolation is configured automatically in our platform.
- **Physical isolation:** Even though Kubernetes/Containers implement isolation, some resources such as the host filesystem (for writing logs) are still shared. For latency-sensitive applications, such as 5G DUs, we dedicate physical nodes so as to avoid interference from other applications. In Kubernetes, this is implemented using Labels/Selectors or Taints/Tolerations.

  Even with these constraints in place, you cannot stop someone with permission to deploy Kubernetes pods from activating a pod on nodes which they are not supposed to manage. To solve this, we created a new Kubernetes primitive called Resource Pools, which segregate the physical nodes into pools. Users and automation scripts can only deploy pods in their assigned resource pools.

  Resource pools also allow users to isolate the storage traffic within the resource pool boundaries. By allocating the storage from the same resource pool where pods are deployed, it secures the storage traffic. Other applications which do not have access to a particular resource pool will not be able to see traffic flowing in any other resource pool.

- **User isolation:** Kubernetes supports logical isolation using namespaces, but we need a higher-level primitive, which can be easily understood and managed. Many organizations already have a directory (ActiveDirectory, LDAP) for managing users, Rakuten included. We integrate with ActiveDirectory for authentication and import users into Symworld Platform seamlessly. This also allows us to manage users across thousands of Symworld clusters. Rather than giving user permission to each Kubernetes object type/namespace, Symworld Platform defines roles at higher levels like Application Administrator/Operator, Cluster Administrator. Symworld Platform handles the complex mapping of these higher-level roles into granular Kubernetes privileges.
- **Vendor isolation:** Different vendors need to have grouping of their users under one primitive where they can manage permissions for resource pools or namespaces. We define this in a primitive we call Tenant, which makes managing vendor users across different clusters easy and less error-prone.

  Like any production-ready multi-vendor environment, we need an easy way to manage and enforce limits and quotas. Each tenant in Symworld Platform is limited in terms of resources it can use (CPU, memory, storage). We also track resource usage per tenant for capacity planning and to charge for infrastructure costs.

Between these levels of isolation, we limit the reach of any successful hack. Next, let us discuss how to reduce these hacking threats before they happen.

4.3 | Holistic vulnerability management

In complex IT environments, new security issues are continuously discovered as hackers continue to learn new vulnerabilities. A modern vulnerability management process can quickly identify and address upcoming issues.

To handle the challenges of country-wide 5G deployments, we implemented a streamlined vulnerability management process with five phases:

1. **Discover technology**
2. **Find vulnerabilities**
3. **Prioritize issues**
4. **Map to remediation actions**
5. **Support teams in remediation**

1 Discover technology

In evolving IT infrastructures, keeping track of every single security-relevant technology component – network appliance, servers, operating system packages, middleware, software libraries, and applications – is a large challenge. Instead of trying to solve this challenge through a central asset directory, we instead combine readily available data sources to create an approximate inventory. This ad-hoc inventory aggregates information from:

- Server management systems
- Network management systems
- Security scans (both network-based and authenticated)
- The software bill of material of Docker container images

By combining these data sets, we reach a close approximation of the IT infrastructure and software components running in our network, which provides the foundation for the next step in vulnerability management: Finding vulnerabilities.

The overview also allows for quick queries around newly-discovered issue, along the lines of "where does software X run in version Y". These queries enable ad-hoc emergency patching for the most severe issues.

2 Find vulnerabilities

No single scanning tool covers all technologies we are using, hence we employ a range of scanners including standard vulnerability assessment tools, application-level scanners, and cloud-specific tools, among others. In addition to automated tools, periodic hacking simulations ("red team" exercises) and a private bug bounty program allow us to spot additional security issues.

The flip side of running a large range of scanners is an overwhelming and partly-redundant amount of vulnerability information, which prompts for strict prioritization.

3 Prioritize vulnerabilities

Our focus is on increasing hacking resilience, that is: Closing those vulnerabilities that hackers like to exploit.

To choose which vulnerabilities to address first, we use prioritization tools that rank issues by their attractiveness to hackers. The most attractive vulnerabilities typically include unpatched bugs, for which exploits are readily available, and severe configuration mistakes including weak or default passwords.

To decide which vulnerabilities matter, our CSIRT team exchange information with equivalent teams at other companies and security vendors regarding incidents worldwide. We find that over 95% of the issues reported by standard tools have no relevance for hacking resilience and can be de-prioritized. The remaining issues are packaged into remediation actions.

4 Map to remediation actions

A number of teams contribute to the remediation of security issues – including sysadmins, DevOps engineers, and developers.

In our experience, most contributors are experts in IT, but not necessarily in security. Presenting them with a list of vulnerabilities often does not provide enough information to address the issues.

We map vulnerabilities into remediation packages, which are described in the domain-language of the remediation contributor. Instead of vulnerabilities, we talk about patching processes; instead of configuration weaknesses, we detail hardening steps; and instead of weak credentials, we explain strong credential management practices.

5 Support teams in remediation

We support the remediation contributors in two ways, until the relevant issues are closed:

First, we track their progress, both in absolute terms and as a relative ranking between the teams. This "gamification" of vulnerability remediation assures that the most effective contributors encourage others to follow in their footsteps.

Second, we remain available for questions throughout the process, to ensure every contributor feels enabled to execute remediation actions even when they encounter a particular remediation package for the first time.

Based on tracking and supporting, we find that decentral remediation of issues continuously reduces the number of vulnerabilities in our network, starting with those the hackers are most interested in.

4.4 | Deployment pipeline checks

By implementing several checks in our deployment pipeline, we are able to detect security defects six to eight weeks earlier than before, thereby mitigating risks and reducing the mitigation overhead. This was achieved with three tool-based capabilities and one overarching process.

We start with the CI/CD workflow in which developers create code, commit it to central repositories, then test and deploy mostly automatically. Initially, many security issues were only detected after deployment, causing reactive remediation rather than proactive remediation.

We introduced three capabilities to detect security defects earlier. Static Application Security Testing (SAST) and Software Composition Analysis (SCA) assure that known security issues are detected before deployment. Cloud Security Posture Management (CSPM) ensures that new security issues are quickly addressed after deployment.

1. **SAST** (Static Application Security Testing). When source code is committed to our repositories, we automatically scan for common security defects. Feedback is provided directly to the developer who wants to commit code, allowing them to address defects while still in the flow of working on a particular piece of code. In our experience, this greatly reduced both the time required and the perceived “annoyance” of having to address security defects. Furthermore, developers generally start adopting more secure coding practices based on the immediate feedback.

   We find that SAST tools, including open-source options, cover many different programming languages and are relatively easy to implement as they hook into standard code versioning platforms and CI/CD tools.

2. **SCA** (Software Composition Analysis). Next, applications are packaged into container images, which can also introduce security issues, for example by including outdated libraries. SCA tools check the software bill of material (BOM), for such imported vulnerabilities and provide immediate feedback to the developer or DevOps engineer building the container image. The SCA tools also generally suggest how to mitigate the issue, for example by updating to a more recent library version.

3. **CSPM** (Cloud Security Posture Management). Once containers are deployed, new (“zero day”) security issues can be discovered in their software components.

A BOM is automatically generated when creating container images and is used to identify such newly-discovered vulnerabilities in already-deployed containers. A CSPM analyses the BOM and other data and flags security issues to the DevOps engineer, and optionally to developers, usually triggering a rebuild of the image with updated components.

To be most effective, the three tool-based capabilities need to be integrated into an overall pipeline governance process, which ensures that security checks are enforced and that feedback is well-understood and acted upon by developers and engineers.

For us, this Pipeline Governance Process achieves two objectives:

1. **Early identification:** Issues are generally identified 6-8 weeks earlier when compared to a CI/CD process without embedded security checks
2. **Reduced remediation effort:** With the identification of the issues during a development sprint, remediation of security issues is now integrated within the development cycle, causing less overhead for developers

4.5 | Holistic security monitoring

Ramping up security monitoring can take a long time to adapt to new environments. Specifically for cloud-native 5G deployments, the monitoring scope also continuously evolves, further increasing the challenge of creating holistic security monitoring.

At Rakuten, we needed monitoring from day 1 and hence started implementing a continuous improvement process that allowed adaptive learning even before our first mobile network was launched in Japan. The continuous learning approach is built around improving hacking threat detection on two fronts in parallel through simulated hacking activity, expressed in two scores:

1. **Signal Score.** First, data about potential hacking activity must be available on our analysis platform. In a complex and growing network like our 5G environment, it is not trivial to know whether all required information is successfully collected. To create visibility about the Signal availability, we regularly simulate hacking attempts on all services exposed to any of our networks.

   Take for example servers that expose SSH to the network. We run a series of attempted logins with a unique usernames per server. The logins attempts should be logged and forwarded for analysis. We check which of the unique usernames are found in our central log storage. For any server that did not correctly report the suspicious activity, we investigate where the log forwarding is broken and apply fixes accordingly.

   We have implemented similar signal heuristics for many of our services across more than 500,000 active IP addresses in our network. Regular testing leads to continuous signal improvements. Three months after implementing these tests, the signal availability had already increased by more than 40%.

2. **Analytics Score.** The collected signal needs to be processed to create alerts. We implement a range of hacking tests for a large and growing number of hacking techniques to check whether a technique is successfully flagged as suspicious.

   Unlike the signal tests, analytics tests only need to be run on a small subset of IP addresses to check whether our analytics engine works correctly. We typically test two IPs per environment acknowledging that each environment can provide individual challenges to our analytics stack.

We are simulating over 30 hacking techniques so far, with a focus on the technologies hackers are most interested in. For example, APIs are a likely hacking target in 5G environments. API hacks can lead to information leakage, identity theft, and service unavailability; hence we prioritize them when implementing analytics tests.

The Analytics Score tests drive continuous improvement of our Security Operations Center (SOC) performance. During the first three months of running the tests, our analytics coverage doubled.

Signal and Analytics Scores are each measured on a scale of 0-100%. The overall Security Monitoring Performance Score is simply:

Security Monitoring Performance Score = Signal Score x Analytics Score

Additional end-to-end assurance. The signal and analytics measurements continuously increase the detection baseline bottom-up. They do not however capture Response capabilities. To quality assure the Security Monitoring process end-to-end we periodically conduct Red Team simulations in which ethical hackers break into our network to provide a real-world opportunity to respond to their activities.

Threat Response. Our analysts work 24/7 and are supported through modern technology to increase the effectiveness of their response. For instance, our security orchestration, automation, and response (SOAR) tools invoke automation scripts, for example to automatically collect forensics evidence and rebuild suspicious network elements. These processes aim to continuously increase the level of automation in threat detection and response over time.

5 | Conclusion

Venturing into new technologies creates opportunity and risks. 5G and Open RAN are no exception: They provide faster, lower-latency, and smarter networks, but introduce an unprecedented level of technology scale and complexity.

Fortunately, other industries have paved the way towards securely deploying and operating cloud-scale infrastructures. At Rakuten Mobile and Rakuten Symphony, we leverage best practice knowledge in five ways:

1. Automate workflows to reduce human error and tightly manage access

2. Isolate components as much as possible to reduce the impact of hacks

3. Engage technology teams in continuously reducing security effects

4. Provide immediate feedback on such defects through pipeline checks

5. Continuously improve our detection capability through simulated hacking

We train every day because the race is never over and the hackers never slow down. The first five steps you take in this race will be your most important. We hope this guide brings clarity to the track ahead.

“The Rakuten telco environment, typical for a cloud-native 5G network, comes at unprecedented scale: We run thousands of containers in hundreds of clusters spread across the country.”