O-RAN next Generation Research Group (nGRG)

Contributed Research Report

Platform Security in Next-Gen Mobile Networks

Report ID: RR-2024-12

Contributors:

Dell

Nokia

Qualcomm

Rakuten

CICT

Release date: 2025.02

Authors

Alex Reznik, Dell (Editor-in-Chief)

Michael Rash, Dell

Clifton Fernandes, Nokia

Ian Oliver, Nokia

SooBum Lee, Qualcomm

Antoynette Ngye, Rakuten

Wiktor Sedkowski, Nokia

Shihan Bao, CICT

Reviewers

Vikas Dixit, Reliance Jio

Mohammed Soliman, Iowa State University

Scott Poretsky, Ericsson

Krishna Pramod Adharapurapu, Rakuten

Pekka Kuure, Nokia

Executive summary

The industry trend towards open, disaggregated systems, as embodied in the work in O-RAN ALLIANCE, is to gain significant advantages over traditional mobile infrastructure, such as flexibility to create highly customized solutions with best-of-breed components; efficiencies driven by the economies of scales; increased competition; – and innovation driven by lower barriers to entry. Open, disaggregated systems are foundational to the mobile network clouds which Mobile Network Operators increasingly deploy to address the highly heterogeneous use cases and requirements that we commonly understand under the umbrella of 5G. They are key in enabling the economic feasibility of distributed "edge computing" systems that drive many "5G applications." It is not unreasonable to expect that 6G networks will be built almost exclusively upon such systems, because of resource optimization needs related to the growing demand for computation and communication resources consumption. However, as mobile networks and mobile cloud systems become ever more open and disaggregated and are sourced from an increasingly larger set of suppliers, security vulnerabilities associated with other similar systems – notably modern IT infrastructure – are likely to become an increasingly significant threat.

As highlighted in the O-RAN White Paper on Zero-Trust [1], mobile networks require a significantly stronger security posture than is typically present. Techniques necessary to enable such a security posture for network infrastructure are generally not new – much of what has been summarized in this research report has been known to the security community. Considering the expected pervasive reliance on open, disaggregated systems in 6G networks, the importance of security is significant and therefore warrants a focused discussion.

This report examines threats to 6G networks and recommends controls and mitigations to make 6G networks secure from existing and emerging threats.

Table of Contents

List of abbreviations

3GPP	3rd Generation Partnership Project
AI	Artificial Intelligence
APT	Advanced Persistent Threats
BOM	Bill of Materials
CaaS	Containers-as-a-Service
CNF	Containerized Network Function
CVE	Common Vulnerabilities and Exposures
DICE	Device Identifier Composition Engine
DRTM	Dynamic RTM
ML	Machine Learning
IETF	Internet Engineering Task Force
IoT	Internet of Things
IT	Information Technology [referring to Enterprise]
KVM	Kernel-based Virtual Machine
LTE	Long Term Evolution
MANO	Management and Orchestration
MBOM	Model BOM [referring to AI models]
MNC	Mobile Network Cloud
MNO	Mobile Network Operator
NIST	National Institute for Standards and Technology [US-based]
O-CU	O-RAN Central Unit
O-DU	O-RAN Distributed Unit
O-RAN	Open Radio Access Network
O-RU	O-RAN Radio Unit
ODS	Open Disaggregated System
OEM	Original Equipment Manufacturer
OSS	Open Source Software
RAN	Radio Access Network
RATS	Remote ATtestation procedureS
RCE	Remote Code Execution
RIC	RAN Intelligent Controller
RTM	Root of Trust for Measurement
SBOM	SW BOM
SI	System Integrator
SMO	Service Management and Orchestration
SRTM	Static RTM
SSDF	Secure Software Development Framework
SW	Software
TCB	Trusted Computing Base
TCG	Trusted Computing Group
TPM	Trusted Platform Module
UE	User Equipment
UEFI	Unified Extensible Firmware Interface
VNF	Virtualized Network Function

List of figures

List of tables

1 Introduction

The industry trend towards Open Disaggregated Systems (ODS), as embodied in the work in O-RAN ALLIANCE, is grounded in the significant advantages that such systems can offer: the flexibility to create highly customized solutions and systems by bringing together the right mix of best-of-breed components, efficiencies driven by the economies of scales, and innovation driven by lower barriers to entry and thus increased competition. Such systems are foundational to the emerging Mobile Network Cloud (MNC), which, for the purposes of this report, we define as any cloud infrastructure (public, private, or hybrid) whose primary purpose is to support mobile network functions and applications. Our definition of mobile network cloud includes infrastructure hardware, the physical components (e.g. compute, storage, networking), and infrastructure software, the virtualization and Containers-as-a-Service (CaaS) platform, and associated management and automation.

Mobile Network Operators (MNOs) are increasingly deploying MNCs to address the highly heterogeneous sets of use cases and requirements that we commonly understand under the umbrella of 5G. Addressing these use cases is key in enabling the economic feasibility of distributed “edge computing” systems that drive many “5G applications.” It is not unreasonable to expect that 6G networks will be built almost exclusively using such systems. However, as mobile network and MNC components become more open and disaggregated and are sourced from an increasingly large set of suppliers, security threats and vulnerabilities associated with other similar systems – notably modern IT infrastructure – are likely to expand the attack surface.

Protecting the mobile network from these threats requires an adoption of a Zero-Trust approach towards the entire MNC. O-RAN’s recent White Paper on Zero-Trust Architecture (ZTA) [1] lays out the foundational principles of how to do this. Security Work Items in O-RAN’s WG11 are addressing security across the O-RAN architecture to achieve a ZTA through stages following a process of asset identification, threat analysis, risk assessment, and normative security specifications to mitigate risk in O-RAN from the evolving threats. This Research Report picks up from [1] and provides an overview of some of the key techniques that the industry should consider as we move towards building a ZTA for 6G networks.

The rest of this report is structured as follows. In Section 2 we provide a definition of Open Disaggregated System followed by an overview of some key threats to such systems. in Section 3. Section 4 addresses the notion of how a trusted platform plays a key role in platform security. Sections 5 and 6 summarize other key techniques for achieving platform security and addresses the threats highlighted in Section 3. We summarize and conclude this report in Section 7.

2 Open Disaggregated Systems in Emerging Communication Systems

An ODS is a system or application composed of many components from various sources. To achieve disaggregation, each component may be obtained from multiple supply sources with minimal impact to the other components. ’Open’ requires the use of well-defined standardized interfaces between the components. Many of the hardware and software components are based on publicly available, e.g. open-source, reference designs (thus open systems) – a key aspect in lowering costs, removing barriers to entry for small players and increasing competition. Critically, in such ODS systems, many components tend to become common and widespread – as witnessed by the widespread adoption of Linux and KVM-based virtualization first across IT and now Telecom. Here again, building on a small set of common “foundations” allows widespread interoperability while reducing the effort required to be relevant in the market.

The industry has done an excellent job securing mobile systems. Mobile network protocols as defined by 3GPP are often considered to be some of the most secure standardized protocols defined. Each generation of mobile technology has been more secure than its predecessors. 5G is the most secure generation of mobile technology to date and the same is expected as the industry evolves into 6G. With the emergence of open and virtualized platforms supporting 5G implementations including O-RAN, platform security has been given significant attention. The impact of attacks against the platform or underlying infrastructure can be significant – allowing the systems to be brought down or, even worse, taken over and controlled by malicious entities. With 5G systems already increasingly supporting critical infrastructure and 6G systems expected to play an even more central role in critical infrastructure, the impact of an attack could be high. Security will need to continue to evolve with the increased attack surface due to ODS and underlying cloud infrastructure,

3 Threats to Open Disaggregated Systems

In this section, we examine the threat surface to consider when securing an ODS.

Open Source Software

While the quality of open-source software (OSS) has increased, the depth of dependencies continues to present a challenge. Additionally, some open-source software is not designed to be used in production deployments and may not be properly secured. Modern projects include many dependencies (69 according to [11]). Due to their complexity, there could be the cases where some of those dependencies are not well-managed even in large projects. Thus, the likelihood of poorly maintained open-source code being used in production is non-trivial. For example, it has been reported that an average application has about 5 critical vulnerabilities and an average patch time of 98 days [11]. Log4j is an example of a critical vulnerability (NVD - cve-2021-44228) in OSS with high impact. While some of these issues are likely addressed by improved supply chain visibility (for example SBOM), given the scale of use of Open Source, the problem with OSS security is likely to persist. It is necessary that vendors implement secure software development practices to ensure secure consumption of OSS.

Complex Supply Chains

As the number of components in a disaggregated system increases, so does the size and complexity of the supply chain of such a system. Moreover, each component is increasingly likely to be put together from a number of sub-components, some (often many) based on open source. Indeed, the economics of competition in a market where open source is available makes it almost impossible to not take advantage of it. System Integrators (SIs) often have both limited visibility into the supply chains of their component providers and limited control over them. With each link in a supply chain potentially vulnerable, supply chain complexity associated with disaggregated system tends to significantly increase the vulnerability of the overall system. The US NSA ESF report on Open RAN security acknowledges this increased risk in open systems [37].

The use of opensource software is not, in itself, a problem. The challenge is establishing the provenance of all components that could have possibly been used by multiple participants in the value chain and ensuring that these are securely consumed using secure software development practices consistent with NIST's SSDF [18].

The emergence of SBOM should, in the short term, provide a new level of visibility into SW components used and the related potential vulnerabilities across the board. However, it remains to be seen whether SBOM data will enable SW consumers to put such pressure on suppliers to provide patches and to enable processes to apply patches easily on deployed systems. The dynamic and complex nature of advanced 5G and 6G networks, characterized by rapid deployment cycles and a diverse array of components from multiple vendors, creates significant challenges for traditional Software Bill of Materials (SBOM) solutions [23]. Additionally, there are several known issues with SBOMs as applied to MNCs, that need to be addressed. SBOM is discussed further in section 6 of this research report.

Remote Code Execution (RCE) exploits

A Remote Code Execution (RCE) attack takes advantage of vulnerabilities in the implementation of a system component (the ability to overflow heaps and stacks is a common example) to load and execute malicious code *at runtime* and with privileges that may allow it to take over system control. Since the attack happens at run-time and can be launched from a remote terminal, traditional platform security techniques (e.g. secure boot) are not effective against it. APIs implemented without recommended security best practices can enable exploitation of vulnerabilities [40]. The Cloud Security Alliance (CSA) has ranked "Insecure interfaces and APIs" the number 3 threat to cloud computing in 2024 [41]. The most well-known RCE vulnerability impacting the telecom industry is the Log4j critical vulnerability (NVD - cve-2021-44228).

Advanced Persistent Threats

ODS, due to their inherent nature of incorporating components from diverse sources, presents a unique and expanded attack surface for Advanced Persistent Threats (APTs). APTs are sophisticated threat “actors”, that are able to exploit a known, unpatched vulnerability to establish a beachhead and move laterally internal to the network to perform reconnaissance, data exfiltration, unauthorized control, or disruption/damage. With an APT, the external threat actor transitions to an internal threat actor. The US NSA’s report on Open Radio Access Network Security Considerations [42] states:

> The use of cloud infrastructure introduces security considerations that must be addressed to protect against internal threats and advanced persistent threats that can move laterally through a cloud deployment.

The Zero Trust Architecture (ZTA) defined by US NIST provides a security framework for protecting digital systems from external and internal threats [43]. The ATIS report on Enhanced Zero Trust and 5G recommends that nextG networks implement continuous monitoring, anomalous behavior detection, dynamic security policy management, and threat intelligence to be able to detect and protect against APTs [44].

Zero-Day Exploits

Zero-Day Exploit is a type of cyber-attack that takes advantage of previously unknown vulnerability in software, hardware, or firmware. Because the vulnerability is unknown, there are no patches or defences available at the time of exploit. If such code is inserted into a component and accepted as valid, traditional platform security techniques are not effective against it. Open-source projects are particularly vulnerable to such contribution of benign-looking code, and while most widely used open-source projects have significant processes in place to protect against such instances, the likelihood of zero-day exploit insertion remains non-zero. The complex ODS supply chains then amplify the small risk of zero-day exploits associated with any single component. Secure software development best practices, such as the NIST SSDF, should be followed to minimize the introduction of zero day vulnerabilities [18].

Gaps in measurement and attestation of states of complex multi-purpose systems

In the context of a mobile network, an end-entity may be physical cloud infrastructure hardware (server) and cloud infrastructure software (OS, hypervisor). Authentication and authorization of an end entity does not necessarily assure that the end entity can be trusted. No single criteria can be used to appraise or evaluate the trustworthiness of all end-entities, universally. Trustworthiness may depend on the composition and other factors that need to be evaluated by the service (such as the relying party) and service-specific requirements. Compromise of any system software/firmware loaded in the boot process may result in a compromise of all software loaded on that computing system in a later time. Control flow attacks and Return-Oriented Programming (ROP) attacks aimed at subverting Root of Trust for Measurement (RTM) are also possible [30].

Launch-time measurements of the computing platform and evaluating the platform integrity based on these reported measurements have been developed for the past decades [20], [21], [22]. However, ever-increasing complexity and extensibility of the computing platform grows with the size of the Trust Computing Base (TCB) upon which the trust chain of the computing platform is established. Complex and large TCB entails larger attack surface compared to the smaller ones.

4 Leveraging attestation for trustworthiness in 6G

4.1 What is trusted computing

A trustworthy system is one that can prove its identity and integrity. What this means is that at least during boot/run-time for machines, load/placement/run-time for containers and Virtual Network Functions (VNFs), one can ensure that the element in question has not been tampered with by any malicious actor. This can be applied to the supply chain to ensure that a Chain-of-Trust (or cross-referenceable Chains-of-Trust) can be established. The relevance here is that the expected integrity values need to be supplied by the source, e.g.: manufacturer, OEM etc., and be communicated to the verifier such that these can be validated against the computed measurements. For example, see the Linux Firmware Service LVFS as to how manufacturers like Dell, HP, Lenovo can distribute firmware and the necessary integrity values that can be attested via the TPM [20]. The increasing use of SBOM and similar identity/integrity measurements is being mandated by certain legislation, for example in EU NIS2 Directive.

Any VNF and/or software component needs to be able to provide traceable evidence to prove who it is, what it is, and ideally its provenance (supply chain). If all these claims can be attested and verified, then a decision can be made that the system can be trusted.

A trusted system means that it is in a known good, verified state – knowing this allows detection of any kind of change to system properties. For example, when an O-RU is deployed the need is to know how it is booted up, configured, on what physical environment it is running on, and where it is connected. The supply chain is important, and it must be validated from where the physical machine (e.g.: O-RU) and supporting software (operating systems, configuration, VNFs) have come from: e.g. manufacturer, firmware configuration. This further builds the chain of trust from the manufacturer to the end-users, ensuring that every item in that chain can be trusted

Trusted Computing based on a root of trust, typically a hardware Root-of-Trust such as a TPM, is able to provide a cryptographic identity and capabilities for recording the integrity of the system. Research indicates that remote attestation [12], if used as part of the network management, operations, and supply-chain, can provide verification and integrate with the management functions of a system. For example, a future definition of the O-RAN SMO or NFV MANO could include remote attestation.

4.2 Trusted-computing-based security techniques

If we have attestation capabilities, the need is to give measurements to the other systems on top of the network management hierarchy. Remote attestation is based on the idea of measurement. For example, cryptographic hashes are calculated during the boot-time of a device or as part of file system integrity monitoring.

Further this involves collecting those measurements which are known as claims and verifying them in a remote attestation server against known good values. Once this is complete, a decision is made whether to trust the device. This gives four actions: measure, attest, verify, and decide as expressed in Figure 1 utilizing the formalisms here.

$$\begin{tikzcd}
Element \arrow[r, "measure"] \arrow[rr, dashed, bend right, "attestable"'] \arrow[rrr, dashed, bend right=50, "trustworthy"'] & Measure \arrow[r, "attest"] & Claim \arrow[r, "verify"] & Result \arrow[r, "decide"] & Decision
\end{tikzcd}$$

Figure 1 Definition of trust [12]

If measurements - cryptographic hashes, boot logs, identity structures, TPM, SGX (CPU enclave) quote structures, etc. - can be used and the element has some interface providing access to those measurements then the element is attestable, and a claim may be produced. If the correctness of the obtained claim can be verified, then the element can be denoted as being “trusted” at that point in time. Trusted elements are those elements that map to trusted decisions.

4.3 Making this specific to O-RAN

A trusted O-RAN implementation means that its architectural elements providing O-RAN functionalities are trusted according to the requirements set for that O-RAN deployment, as established through remote attestation. The underlying O-Cloud must also be trusted, and trust here could be established through remote attestation provided within the NFV MANO. Furthermore, any functionalities realized through the applications deployed on it and being utilized would also need to demonstrate their trust in some manner. Remote attestation facilities in the SMO and NFV MANO would need to communicate and a mechanism for communicating the current trusted status of the elements under their respective jurisdictions is thereby required to be put in place.

Trust in an O-RAN system is provided from a core Root-of-Trust using hardware Root-of-Trust modules such as the TPM or some other mechanism found in CPU-enclaving systems such as Intel SGX, AMD SEV-SNP, and Arm TrustZone [12].and Figure 2 below describes these elements.

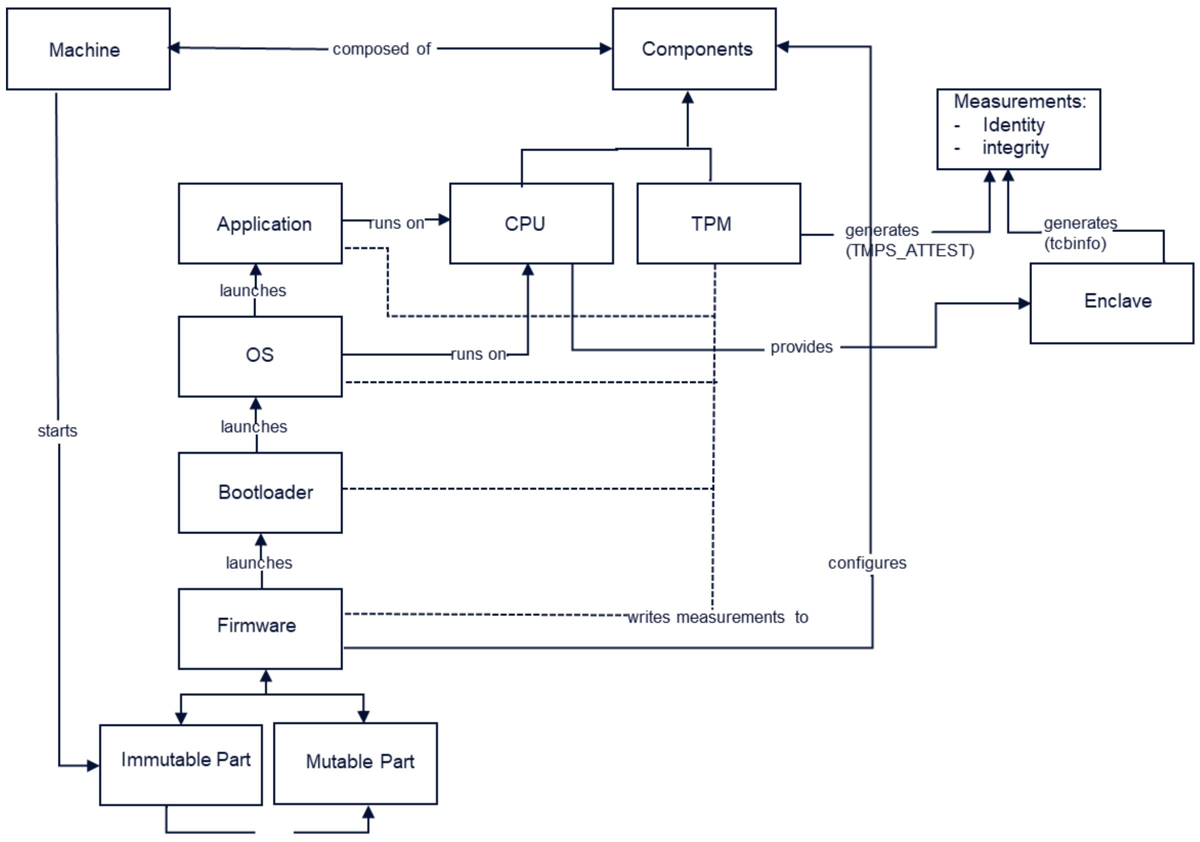

Figure 2 Trust elements [12]

A system consists of several components which are capable of running network applications. The measured boot process is carried out from one stage to the next one. For example, the Unified Extensible Firmware Interface (UEFI) measured boot on x86 devices and Dynamic Root-of-Trust found in Intel TXT-equipped systems is measured. This starts with an explicitly trusted piece of code [15], which then cryptographically measures (hashes) the next piece of code to run which is the firmware and then written to a TPM. This process continues during boot until run-time forming a Chain-of-Trust right from the firmware until the applications on top of the stack.

Figure 3 below shows the structures used to establish the identity and integrity of a system utilising a TPM. The quote structures (TPMS_ATTEST) provide cryptographically signed evidence or claims about the machine's identity and integrity.

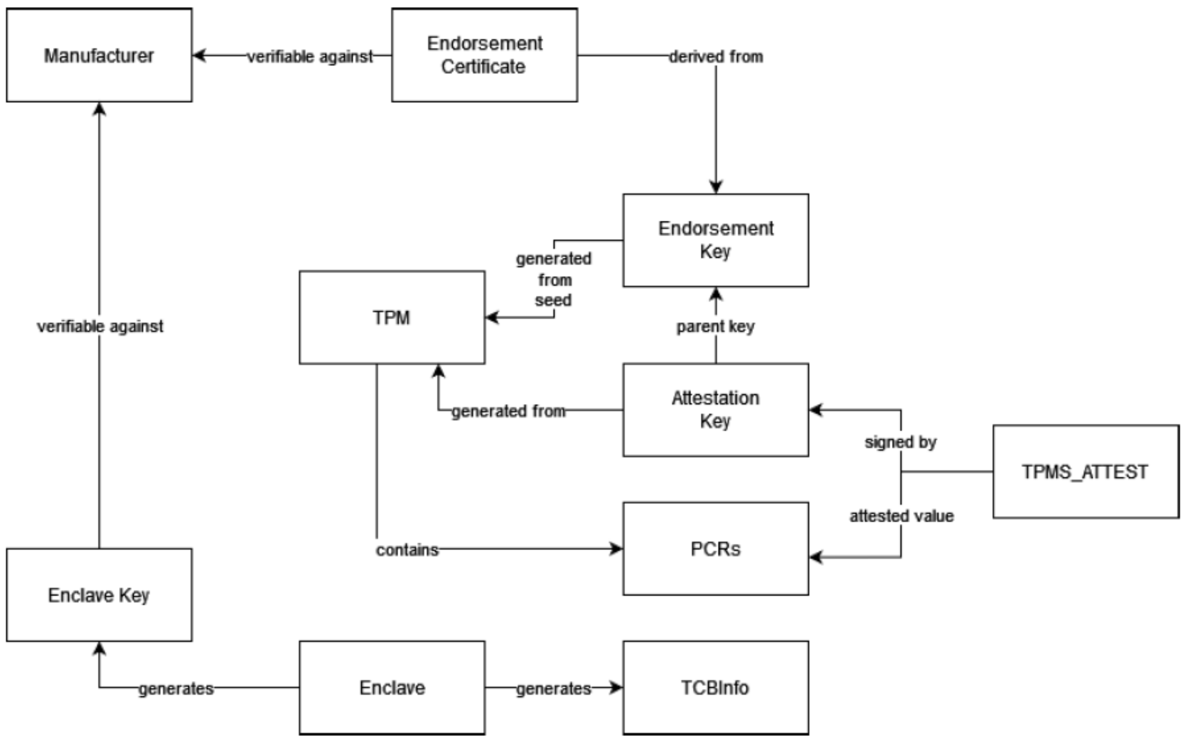

Figure 3 Quotes and their derivations and relationships [12]

- It is important to note the cross-referencing, such as the TPM's Endorsement Certificate being verifiable against a manufacturer's certificate authority or the UEFI eventlog's contents used to calculate a value against those in the Platform Configuration Registers (PCRs). The TPM's quote (TPMS ATTEST) structure itself is signed by an attestation key (AK) derived from the endorsement key (EK) and contains a hash of the hashed public parts of the AK and the EK. These are the 'measurements' from any given element. An AK is derived from an EK to establish a certificate chain to the TPM's (and thus the device's) identity. Technically any signing key can be utilised, but a specific EK-AK relationship allows further attestation using TPM identity attestation commands.

In Figure 4 below we have a PC's (example of a system) firmware and firmware configuration measurements which becomes a claim through the process of attestation.

$$\begin{tikzcd}
 & t : TPM \arrow[r, "measure"] & ak : Key \arrow[r, dotted] & q : TPMQuote \\
e : PC \arrow[ur, dotted] \arrow[r, dotted] \arrow[dr, dotted] & f : FW \arrow[r, "measure"] & pcr0 : Hash \arrow[r, dotted] & pcr0||pcr1 : Hash \arrow[u, dotted] \\
 & c : Config \arrow[r, "measure"] & pcr1 : Hash \arrow[ur, dotted] &
\end{tikzcd}$$

Figure 4 Elements and measures [12]

With composition above being linked together, this shows the structure of a PC composed of a TPM, firmware, and configuration, each of these with its own measurement(s). A TPM quote is a measurement formed by the composition of these measures. The composition of PCRs means an ordered concatenation and hashing (Merkel tree) and the composition of a key means signing by that key. The fields of the TPM's quote or TPMS ATTEST structure as described in Table 1. Similar structures for attestation claim from other components, e.g.: SGX [16], can be found in relevant specifications from the manufacturer.

Field	**Description**
pcrDigest	hash of the selected PCR entries
pcrSelect	list of the selected PCR fields
magic, type	Fixed values denoting the type of the structure (TCG defined)
firmwareVersion	Firwmare version of the TPM 2.0 device
clockInfo::clock	Value of the TPM's internal clock
clockInfo::resetCount	Counters showing the number of power cycles
clockInfo::restartCount	Counters showing the number of sleep (S3) cycles
clockInfo::safe	Flag showing whether the TPM was powered off explicitly
extraData	Any user supplied data, eg: a nonce for replay attack prevention
qualifiedSigner	a hash of the public parts of the signing key's hierachy

Table 1 - TPM 2.0 Quote (TPMS_ATTEST) fields

It follows that the term 'integrity of a system' is just the composition of all possible measures for that system. Evidence of, such as in Table 1 above, is provided by each element in the system and collated and verified through remote attestation. Through the verification of this decision about the trust and trustworthiness of elements, sections of the system and the system as a whole can be attested.

4.4 Expanding to dynamic and complex system

A dynamic/flexible framework is necessary to evaluate the security posture of an entire system comprising various types of (composite) devices including peripherals and other co-processors (e.g., cryptographic accelerators and specialized processing units) and their firmware/software. In particular, the following capabilities and policies would further enhance evaluating a system's security posture.

- Service-specific dynamic appraisal policy to evaluate an entity.
- Up-to-date reference data is used for evaluating an entity.
- Dynamic Root of Trust for Measurement (DRTM) in addition to Static Root of Trust for Measurement (SRTM)
- An extensible set of claims/quotes
- Boot time, Launch time, and runtime measurements
- Detailed information about the claims/quotes

- Standardized claim/quote structure and encoding formats
- On-demand or periodic remote attestation

As critical services, such as telecom services, become enabled in the cloud or even hosted by cloud service providers, it is necessary for those service owners to be ensured of the trustworthiness of the cloud infrastructure hosting their services to protect not only their services, but also confidential data associated with services. This essentially requires verifying the integrity of the cloud infrastructure much like verifying a single device. The O-RAN ALLIANCE is addressing this in its O-Cloud security work item led by its WG11 for security [45].

Different services may have different evaluation criteria and/or proof of the device state. For example, a service that processes confidential data wants to verify whether the remote platform supports confidential computing and further a required feature for the service (e.g., full memory encryption) is enabled on the platform. Other services may want to verify the last boot time or the time that the current measurement was performed. Such evaluation criteria are defined as an appraisal policy and determine the trustworthiness of the platform for each service. Especially for a complex system consisting of various services and devices, support of service-specific dynamic appraisal policy would help manage the overall trustworthiness of the system.

Standard-based, extensible attestation architectures and protocols [20] [21] have been developed for the past years. These architectural reference models would help build a unified attestation framework in mobile networks. Both IETF Remote ATtestation procedureS (RATS) [20] and Trusted Computing Group Device Identifier Composition Engine (DICE) [22] architectures share a similar architectural model with some differences in the reporting structure and formats.

The RATS specification lists potential use case scenarios, which include [20]

- Network Endpoint Assessment
- Confidential Machine Learning Model Protection
- Confidential Data Protection
- Critical Infrastructure Control
- Trusted Execution Environment Provisioning
- Hardware Watchdog
- FIDO Biometric Authentication

In particular, network endpoint assessment is most relevant to mobile networks, yet other use cases may equally benefit depending on deployment and/or device capabilities. For example, if xApp and/or rApp confidentiality is deemed necessary even in runtime, Radio Access Network Intelligent Controller (RIC) platform may architecturally support runtime protection of machine learning models, which needs to be attested to xApp/rApp owners before being deployed. As another example, network function may need to verify management system states and possibly authentication state, e.g., based on Multi-Factor Authentication (MFA), before being configured with sensitive and/or critical information such as a trust bundle, e.g., trusted CA certificates.

Open Radio Access Network (O-RAN) comprises various network functions such as O-CU, O-DU, O-RU, and RIC where such network functions can be realized in different forms, e.g., physical network function, virtualized network function, cloud-native network functions on respective computing architectures/platforms. Those computing platforms and the network functions built on them have different layering structures [21] (See Figure 5 Examples of system layering [21]) and accordingly have different attesting environments (see Figure 6 [20]) A certain network function may require a combination of attestation environments, each of which has a unique layering structure.

	Complex OS	Virtualization	Enclave	Embedded System
Hardware	Processor Boot ROM	Dynamic RTM	TEE	Processor Boot ROM
Layer 0	Boot Loader	Hypervisor Loader	Enclave Runtime	Runtime Boot Loader
Layer 1	HLOS	Hypervisor	Enclave Applet	Co-processor runtime
Layer 2	Container Runtime	VM		Embedded Function
Layer 3	Container	Container Runtime		
Layer …		Container		

Figure 5 Examples of system layering [21]

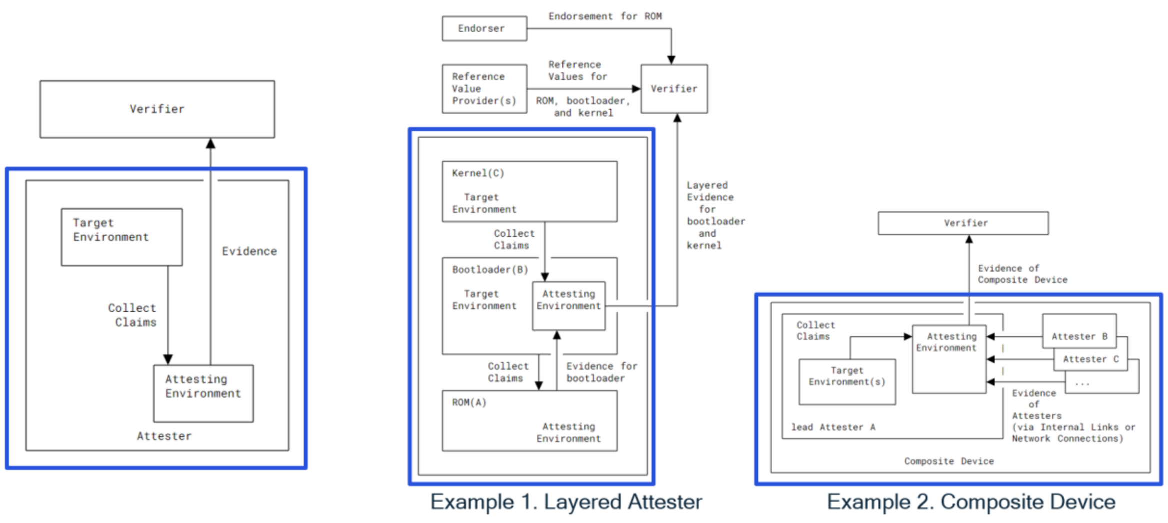

Figure 6 Layered Attestation Environments [20]

A service architecture would determine an appropriate attestation topological pattern among possible patterns [20], [22]. Among different patterns, the combined model (or any variant of it) may be appropriate for the mobile networks especially when it comes to mutual attestation between network functions. In O-RAN, Service Management and Orchestration (SMO) or an equivalent management service, which can take the role of Verifier, thereby collecting attestation information of individual network functions periodically or on-demand and sharing such information with network functions when requested. It should be noted that in such a scenario, SMO should be synchronized with the up-to-date reference values of the attesting network function in coordination with its vendor. Additionally, SMO can configure each network function with the service-specific appraisal policies with which the network function evaluates the attestation results of other network functions and determines service access.

5 Memory Safe Software Development

There is no denying the incredible legacy that the C and C++ programming languages have had in the computing industry. These programming languages have been used to build everything from applications to operating systems to networking stacks and everything in-between, and as such have been fantastically successful. However, from a security perspective, these languages are not built to shield developers from making certain mistakes that could lead to security breaches - particularly in the area of memory handling. While bugs can be introduced in any programming language, memory handling bugs where the C and C++ compilers compile code that contains a stack or heap overflow deserve a special note in the annals of computer insecurity. Successful exploitation of stack or heap overflow vulnerabilities can sometimes result in remote code execution (RCE) privileges being exploited by the attacker, and this commonly results in the total compromise of the target.

Fortunately, over the past 15 years or so, there has been a lot of research performed in terms of language design to provide memory safety as a fundamental aspect of the languages themselves. For example, programming languages like Goand Rustboth provide memory safety as a core security capability, and this is having a significant positive impact on the vulnerability landscape presented by applications and system components written in these languages [38], [39]. There is even a serious effort led by Google to put Rust directly into the Linux kernel [5], and this would be a major win for the security of the resulting compiled code [15]. In essence, Rust enforces memory safety through a restricted ownership model and accordingly realizes automatic memory management, which greatly reduces the burden from programmers to manually handle memory, improving development efficiency and code correctness [31]. Consequently, Rust is becoming known as a capable systems programming language that brings memory safety to areas that would otherwise likely be developed in C or C++.

Now, it should be acknowledged that the massively successful legacy of C and C++ will be with us for the foreseeable future. The Linux kernel itself is primarily developed in C, and many vendors playing in the O-RAN space have standardized the usage of Linux. However, providing “memory safety” is still required even for this code. That is, whenever a stack or heap overflow pattern is introduced into, say, the Linux kernel, this represents a security vulnerability and hence must be fixed. Detecting the existence of such a vulnerable code pattern may be achieved with various techniques such as static analysis, dynamic analysis, and fuzzing. Further, a lot of research has gone into the compilers themselves to provide protections against such vulnerable code patterns. For example, in the GCC compiler, there are over 30 command line switches to enable technologies such as stack protection, read-only relocations, position-independent executables, and more. These protection mechanisms should be used within vendor-build pipelines as binaries are compiled. An excellent and authoritative reference that provides in-depth information on this topic entitled "*Compiler Options Hardening Guide for C and C++*" published through the OpenSSF [19]. This guide discusses everything from compiler versions to runtime performance implications of compiler security options and will likely become a standard reference for the industry over time on how to leverage the strongest compilation options to achieve some measure of memory safety for C and C++.

6 Addressing the challenges associated with Software Bill of Material (SBOM)

According to the O-RAN ALLIANCE and NTIA guidelines, the necessity for generating a new SBOM for each build is critical to maintain accuracy [2], [25]. Traditional SBOMs are often static and generated at specific points in time, which makes them unsuitable for environments where software components are frequently updated and changed. This can result in SBOMs becoming obsolete or insufficient, thereby failing to accurately reflect the current state of the software supply chain. The integration of advanced technologies such as AI, IoT, and quantum computing introduces new complexities and potential vulnerabilities [33]. As discussed in Section 3, the present state of SBOMs needs further evolution to provide the increased level of security that next-generation mobile systems are likely to demand

Another critical challenge concerning SBOM integrity is that it is susceptible to unauthorized alterations or tampering, which can mislead stakeholders about the true nature of the software components [24]. This manipulation can result in inaccurate or incomplete information about the software's composition, hindering vulnerability assessment, compliance efforts, and overall security posture. Consequently, this can compromise the ability to identify and mitigate risks, thereby affecting the security and trustworthiness of the entire software supply chain. The current O-RAN ALLIANCE WG11 security requirements and controls document specifies requirements and controls for SBOMs [2]. While these measures provide a strong foundation for security, the recipient of the SBOMs could take additional measures to ensure comprehensive protection against unauthorized alterations or tampering.

The AI model bill of materials (AI MBOM), like SBOM for AI model, is critical for supply chain security as it offers transparency into the components and algorithms used in AI models, aiding in vulnerability management and compliance. However, challenges such as the lack of standardized practices hinder the comprehensive documentation and analysis of AI MBOMs, posing risks to security and integrity. Currently, O-RAN ALLIANCE WG11 is conducting risk analysis for AI/ML models and their lifecycle within O-RAN architecture elements. It is also specifying normative requirements and security controls to mitigate AI supply chain attacks.

Ensuring data privacy in the context of SBOMs is increasingly important, especially with the integration of IoT and AI technologies that handle vast amounts of sensitive data. Traditional SBOMs may not include sufficient measures to protect data confidentiality and integrity. Inadequate access controls can result in business and compliance risks.

SBOMs are essential for ensuring transparency and security in the software supply chain, particularly within the Open Radio Access Network (O-RAN) for advanced 5G and 6G networks. O-RAN promotes interoperability and flexibility by using open standards and interfaces, necessitating a clear understanding of all software components involved. Currently, SBOMs in O-RAN for 5G are being implemented to track and manage software components. This can aid in identifying vulnerabilities, including zero-day vulnerabilities, and ensuring compliance with security standards [33]. As we transition to 6G, the complexity and scale of networks will demand more detailed and comprehensive SBOMs to manage the increased number of software components and dependencies. The integration of advanced technologies such as AI and IoT will require robust SBOM practices to ensure security, reliability, and privacy [33].

Addressing the highly dynamic nature of next-generation mobile systems requires building SBOMs capable of real-time updates [23] to accurately reflect changes in the network's software components. This can be achieved by integrating SBOM generation and updating processes into the CI/CD development pipeline. The process should be automated so that the SBOM is automatically updated whenever new software is integrated or deployed. Implementing runtime SBOM in production ensures continuous visibility and accuracy of software components, enabling effective security management and compliance in dynamic cloud-native environments.

The O-RAN ALLIANCE already emphasizes the importance of SBOM integrity, recommending cryptographic signing, robust access controls, and regular security assessments to prevent unauthorized modifications and ensure trustworthiness [2].

Building on these recommendations, advanced techniques such as SBOM fingerprinting and blockchain technologies can further make SBOMs tamper-proof. For instance, using blockchain to record each SBOM update transaction allows for tracing and auditing unauthorized changes, thereby providing a transparent and tamper-proof record that enhances the security of the software supply chain.

Having a transparent view of the supply chain helps verify the authenticity of components and quickly identify the source of any compromised components. One way to do so is to have trusted network providers use a blockchain-based ledger. Each vendor in the supply chain ledgers their contribution to the final product on the blockchain, creating a verifiable and immutable supply chain record. Decentralized identity management further enhances this process by logging each vendor’s credentials on the blockchain. This process ensures that only genuine components are deployed in the network, protecting against the threat of counterfeit products.

Artificial intelligence (AI) is increasingly being utilized as a defensive tool against AI-driven security threats across various applications. This strategic deployment of AI is crucial in enhancing cybersecurity measures. Similarly, for Software Bill of Materials (SBOMs), leveraging a proactive AI-based strategy can significantly improve security oversight. By using AI to predict potential changes and monitor for anomalies, organizations can proactively detect unauthorized modifications or emerging security threats. For instance, deploying machine learning models to analyze update patterns in software components can enable the early identification of potential vulnerabilities, allowing organizations to take preemptive security measures both before and after deployment. Additionally, standardizing a Machine Learning Model Bill of Materials (MBOM) can further secure AI models within the supply chain, ensuring a comprehensive defense against potential security breaches.

Advanced privacy-preserving techniques, including data anonymization are required to address the various privacy challenges that the use of SBOMs can bring. Furthermore, identification of relevant privacy regulations, standards, and building compliance from the onset and maintaining it through the operational lifecycle of the software should be a best practice for creation and maintenance of SBOMs.

Development and utilization of standardized APIs to facilitate efficient communication of SBOMs data across different tools and platforms should address the challenges of interoperability and the various SBOM standards. Such APIs should follow modern best practices for secure and robust API development. Furthermore, API gateways should be used to manage and secure API access and ensure real-time processing of SBOM updates. O-RAN ALLIANCE currently mandates adopting standard SBOM formats to ensure consistency and interoperability.

7 Conclusions

Mobile networks are widely recognized as critical infrastructures whose availability and operational integrity are frequently treated as a matter of national security. As our world becomes ever more connected, mobile networks are assuming an ever more critical role. Attacks on the mobile network, such as APTs, can result in compromise and be disruptive. Security solutions need to be implemented for the network to protect against external and internal threats using the principles of a ZTA, as highlighted in the O-RAN White Paper on Zero Trust Architecture [1].

In this research report, we provide an overview of some of the key techniques that mobile network architects need to be aware of as they evolve the MNC. While the overview is far from complete, we believe it provides a good starting point from which to approach the design of secure future mobile networks.

References

[1] O-RAN ALLIANCE, “Zero-Trust Architecture for Secure O-RAN,” (last accessed Sep. 18, 2024) https://mediastorage.o-ran.org/white-papers/O-RAN.WG11.ZTA%20for%20Secure%20O-RAN%20White%20Paper-2024-05.pdf

[2] O-RAN ALLIANCE, " O-RAN Security Requirements and Controls Specification," v. 9.0, (last accessed Sep. 18, 2024) https://specifications.o-ran.org/download?id=705

[3] Roger Piqueras Jover, “5G Protocol Vulnerabilities and Exploits”, ShmooCon 2020, (last accessed Sep. 18, 2024) http://rogerpiquerasjover.net/5G_ShmooCon_FINAL.pdf

[4] Google’s Address Sanitizer (ASAN) extension to the gcc compiler: (last accessed Sep. 18, 2024) https://github.com/google/sanitizers

[5] Google Security Blog: “Rust in the Linux Kernel”, (last accessed Sep. 18, 2024) https://security.googleblog.com/2021/04/rust-in-linux-kernel.html

[6] Zero Day Initiative, “CVE-2022-23088: Exploiting a Heap Overflow in the FreeBSD WiFi Stack”, (last accessed Sep. 18, 2024) https://www.zerodayinitiative.com/blog/2022/6/15/cve-2022-23088-exploiting-a-heap-overflow-in-the-freebsd-wi-fi-stack

[7] Zero Day Initiative, CVE-2022-23088: Full Exploit Code, (last accessed Sep. 18, 2024) https://github.com/thezdi/PoC/tree/master/CVE-2022-23088

[8] “Various Linux Kernel WLAN security issues (RCE/DOS) found”, (last accessed Sep. 18, 2024) https://seclists.org/oss-sec/2022/q4/23

[9] Yael Grauer, “Future of Memory Safety Challenges and Recommendations,” Consumer Reports, Jan. 2023, (last accessed Sep. 18, 2024) https://advocacy.consumerreports.org/wp-content/uploads/2023/01/Memory-Safety-Convening-Report.pdf

[10] Viavi, State of the Network, (last accessed Sep. 18, 2024) https://comms.viavisolutions.com/State-of-the-Network-2023-vs10912

[11] The Linux Foundation Research, “Addressing Cybersecurity Challenges in Open Source Software,” Annual Report, (last accessed Sep. 18, 2024) https://www.linuxfoundation.org/research/addressing-cybersecurity-challenges-in-open-source-software

[12] Oliver I, Kuure P, Sedkowski, W., Sommer, T., “OntologizingTrustworthy in the Telecommunications Domain,” (last accessed Sep. 18, 2024) https://arxiv.org/pdf/2311.15839.

[13] Tan CJ, Mohamad-Saleh J, Zain KAM, Aziz ZAA. “Review on firmware.” In: Proceedings of the International Conference on Imaging, Signal Processing and Communication, 2017. p. 186-90.

[14] Zaidenberg NJ. “Hardware rooted security in industry 4.0 systems,” Cyber Defence in Industry 40 Systems and Related Logistics and IT Infrastructures. 2018;51(135).

[15] Trusted Comptuing Group (TCG), EFI Protocol Specification Level 00, Revision 00.13.; 2016.

[16] Intel Corporation, Intel Software Guard Extensions (Intel SGX) Data Center Attestation Primitives: ECDSA Quote Library API; 2020.

[17] Tomlinson A. Introduction to the TPM. Smart Cards, Tokens, Security and Applications. 2017:173-91.

[18] NIST, Secure Software Development Framework (SSDF), (last accessed Sep. 18, 2024) https://csrc.nist.gov/Projects/ssdf

[19] OpenSSF, "Compiler Options Hardening Guide for C and C++", (last accessed Sep. 18, 2024) https://best.openssf.org/Compiler-Hardening-Guides/Compiler-Options-Hardening-Guide-for-C-and-C++

[20] H. Birkholz, D. Thaler, M. Richardson, N. Smith, W. Pan, "RFC 9334 Remote ATtestation procedureS (RATS) Architecture", January 2023

[21] Trusted Computing Group, DICE Layering Architecture, (last accessed Sep. 17, 2024) https://trustedcomputinggroup.org/resource/dice-layering-architecture/

[22] Trusted Computing Group, DICE Attestation Architecture, (last accessed Sep. 17, 2024) https://trustedcomputinggroup.org/resource/dice-attestation-architecture/

[23] K. David and H. Berndt, "6G Vision and Requirements: Is There Any Need for Beyond 5G?," in *IEEE Vehicular Technology Magazine*, vol. 13, no. 3, pp. 72-80, Sept. 2018, doi: 10.1109/MVT.2018.2848498.

[24] Japanese Ministry of Economy, Trade and Industry Commerce and Information Policy Bureau Cybersecurity Division (METI), "Guidance on Introduction of Software Bill of Materials (SBOM) for Software Management", July 2023, (last accessed Sep. 18, 2024) https://www.meti.go.jp/policy/netsecurity/wg1/sbom_tebiki_en.pdf

[25] The United States Department of Commerce & NTIA "The Minimum Elements For a Software Bill of Materials (SBOM)", July 2021, (last accessed Sep. 18, 2024) https://www.ntia.doc.gov/files/ntia/publications/sbom_minimum_elements_report.pdf

[26] The Linux Foundation "Software Package Data Exchange (SPDX): SPDX Specification", 2021, (last accessed Sep. 18, 2024) https://spdx.dev/wp-content/uploads/sites/31/2023/09/spdx-1.1.pdf "

[27] OWASP CycloneDX, "CycloneDX Specification," v 1.6., (last accessed Sep. 18, 2024) https://cyclonedx.org/specification/overview/

[28] NIST Computer Security Resource Center (CSRC), (date of last checked) "Software Identification (SWID) Tagging," (last accessed Sep. 18, 2024) https://csrc.nist.gov/projects/Software-Identification-SWID

[29] Ministry of Foreign Affairs of Japan, "Open RAN Security Report," (last accessed Sep. 18, 2024) https://www.mofa.go.jp/mofaj/files/100509243.pdf

[30] Stefan Hristozov, Moritz Wettermann, Manuel Huber, "A TOCTOU Attack on DICE Attestation", in Proceedings of the Twelveth ACM Conference on Data and Application Security and Privacy (CODASPY 2022). Association for Computing Machinery, New York, NY, USA, 226–235. https://doi.org/10.1145/3508398.3511507

[31] Abhiram Balasubramanian, Marek S. Baranowski, Anton Burtsev, Aurojit Panda, Zvonimir Rakamari, and Leonid Ryzhyk. 2017. “System Programming in Rust: Beyond Safety,” SIGOPS Oper. Syst. Rev. 51, 1 (August 2017), 94–99. (last accessed Sep. 18, 2024) https://doi.org/10.1145/3139645.3139660

[32] Synopsis, “Coverity Scan Report Finds Open Source Software Quality Outpaces Proprietary Code for the First Time,”, (last accessed Sep. 18, 2024) https://news.synopsys.com/2014-04-15-Coverity-Scan-Report-Finds-Open-Source-Software-Quality-Outpaces-Proprietary-Code-for-the-First-Time

[33] NIST SP 800-53, "Security and Privacy Controls for Information Systems and Organizations", (last accessed Sep. 18, 2024) https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r5.pdf

[34] Wikipedia, Linus’ Law, (last accessed Sep. 18, 2024) https://en.wikipedia.org/wiki/Linus%27s_law.

[35] Scott Poretsky, Robert Byrne, Tuomo Tanskkaren, “Open source software security in an ICT context – benefits, risks, and safeguards,” Ericsson Blog, 2021, (last access Nov. 26, 2024) https://www.ericsson.com/en/blog/2021/1/open-source-security-software

[36] The Linux Foundation, “Report on the 2020 FOSS Contributor Survey,” (last accessed Nov. 26, 2024) https://8112310.fs1.hubspotusercontent-na1.net/hubfs/8112310/2020FOSSContributorSurveyReport_121020.pdf

[37] US NSA, US DHS CISA, “Open Radio Access Network Security Considerations,” (last accessed Nov. 26, 2024) https://www.cisa.gov/sites/default/files/publications/open-radio-access-network-security-considerations_508.pdf

[38] The Go Programming Language, (last accessed Nov. 27, 2024), https://go.dev

[39] Rust Programming Language, (last accessed Nov. 27, 2024), https://rust-lang.org

[40] Top 10 API Security Risks, OWASP, 2023, https://owasp.org/API-Security/editions/2023/en/0x10-api-security-risks/

[41] Top Threats to Cloud Computing 2024, Cloud Security Alliance, August 2024, https://cloudsecurityalliance.org/artifacts/top-threats-to-cloud-computing-2024.

[42] Open Radio Access Network Security Considerations, NSA ESF, September 2022, https://www.nsa.gov/Press-Room/News-Highlights/Article/Article/3159347/esf-members-nsa-and-cisa-publish-open-radio-access-network-security-considerati/ .

[43] Zero Trust Architecture (ZTA), US NIST, NIST SP 800-207, August 2020.

[44] Enhanced Zero Trust and 5G, ATIS, July 2023, https://atis.org/resources/enhanced-zero-trust-and-5g/ .

[45] O-RAN ALLIANCE, “Study on Security for O-Cloud”, v7.0, (last accessed October 2024)