The Road to a Trustworthy 6G; On the Need for a "Zero Trust 6G" Paradigm

Geir M. Køien

Department of Microsystems, University of South-Eastern Norway, Norway
E-mail: geir.koien@usn.no

Received 31 January 2023; Accepted 27 October 2023;
Publication 07 February 2024

Abstract

The high-level aspects of 6G are slowly being agreed upon. It is safe to assume that 6G will bring many enhancements to 5G, including pervasive application of Artificial Intelligence (AI) and Machine Learning (ML) services. The "softwareization" trend is likely to continue and become even more prevalent. Security and trustworthiness are recognized goals for 6G. Accordingly, we predict that Zero Trust principles will become fully integrated in the 6G architecture. While necessary, this will not be sufficient. With the "softwareization" in mind, we postulate a need for increased focus on software development and deployment practices. Thus, we propose to extend the Zero Trust paradigm to encompass software development assurance and bring about a *Zero Trust 6G* regime. This will also be in line with the current developments for improved accountability for software and services.

Keywords: 6G security, softwareization, accountability, "Zero Trust 6G" concepts.

Journal of Mobile Multimedia, Vol. 20_1, 87–110.
doi: 10.13052/jmm1550-4646.2014

1 Introduction

The high-level goals for 6G have not yet been formally defined. However, it seems certain is that 6G will have to provide better and more comprehensive security solutions. This paper investigates possible strategies for security improvements to the 6G system architecture.

1.1 Background

The International Telecommunication Union (ITU) is a United Nations agency, and it is responsible for a host of information and communication technologies recommendations, etc. The ITU-R arm, which is responsible for radiocommunications aspects, has a "Working Party' known as WP 5D. This group is responsible for the high-level system aspects of International Mobile Telecommunications (IMT) systems, comprising IMT-2000, IMT-Advanced, IMT-2020 and IMT for 2030 and beyond. These IMT visions correspond to 3G, 4G, 5G and 6G respectively.

The recommendation for "IMT for 2030 and Beyond", aka 6G, is expected to be similar in scope to the 5G vision ("IMT for 2020 and Beyond" [1]). The 5G recommendation gave us a high-level specification, including usage scenarios, etc. The "triangle vision is from the 5G recommendation.

The mobile systems vendors are now in the process of defining and proposing position statements and usage scenarios. As expected, one will find many proposals concerning adoption and integration of AI/ML-technology.

1.2 High-level Expectations for the 6G System Architecture

The ITU-R report M.2516-0 "Future technology trends of terrestrial International Mobile Telecommunications systems towards 2030 and beyond" [2] is expected to be influential for the technology foundations for 6G. We can safely assume it will contain many of the premises for the 6G requirements. The report contains four sections that are of particular interest:

4. Overview of emerging services and applications
5. Emerging technology trends and enablers
6. Technologies to enhance the radio interface
7. Technology enablers to enhance the radio network

We expect 6G to contain requirements and usage scenarios for these areas. We shall return to the M.2516-0 report in clause 2.1.

1.3 Introduction to the Zero Trust Concept

This subsection briefly introduces the Zero Trust (ZT) concept. We shall return to ZT in clause 4.1.

1.3.1 The zero trust concept

The ZT concept is about unwarranted and implicit assumptions about security. The background were a situation where one commonly assumed that the (intra)network perimeter would serve as a dividing line between untrusted and trusted actors, and that within the perimeter one tended to trust all actors and accesses. This relied on the tacit assumption that the network perimeter would be an effective *access control* barrier and that all actors and entities within the perimeter could be trusted. Of course, this presumes that one could uniquely identify and authenticate all the legitimate actors and entities. And, finally, it presumes that one has explicitly defined a security policy, which includes authorization regimes and proper access control to the various resources. In general, these assumptions are false. ZT therefore rejects the *network perimeter* security assumption. Instead, one adopts an "always verify" and "assume breach" position.

1.3.2 The zero trust architecture

Associated with the ZT concept is the Zero Trust Architecture (ZTA), which is a concrete manifestation of ZT principles and tenets. ZTA is defined in NIST Special Publication (SP) 800-207 [3]. This document includes a definition of the ZT tenets and ZTA as such. We shall not repeat those here, but suffice to say that they expand on and implement the ZT principles.

1.3.3 Operation and maintenance aspects

The ZT concept and the associated ZTA are concerned with operation and maintenance (OAM) of systems such as the mobile networks. To this end, ZTA specifies required functionality, like "Identity and Access Management" (IAM), and required operational methods. However, ZT/ZTA do not address how the systems are designed and implemented.

1.4 Requirements for Designing and Implementing Secure Systems

While the ZT/ZTA concepts are useful and necessary, they are not sufficient. There is also a strong need for designing and implementing software that is more reliable and with fewer flaws, bugs and security vulnerabilities.

1.4.1 Software vulnerabilities

The ENISA Threat Landscape (ETL) report [4], see also clause 2.6, provides an authoritative overview over the current threat landscape. The ETL reports are annual, and are based on (mandatory) reporting from large businesses, national agencies and critical-infrastructure operators all over the EU. Software flaws and bug may cause security vulnerabilities, and are the root cause of many of the incidents highlighted in the ETL report.

The "OWASP Top 10" report further illustrates the fact the many of the threats from the ENISA report stem from vulnerabilities in software [5]. To reduce the number of software vulnerabilities must therefore be a top priority.

1.5 Laws, Regulations and Compliance Aspects

The mobile systems are critical infrastructures. Societal dependency on a well-functioning mobile system is well established, and that has been recognized in legal terms too. This encompasses primary systemic aspects such as availability and privacy, but also less obvious aspect such as accountability. For EU and the European countries this is manifest in legislation such as the EU Cybersecurity Act [6], the proposed EU Cyber Resilience Act [7] and even the EU Digital Services Act [8]. We do not intend to delve any further into these aspects, but suffice to notice that they have a clear impact on how these systems must be designed, implemented and operated.

1.6 Related Research and Paper Outline

Related research, standards and reference documents are mentioned, discussed and referenced throughout the paper.

Section 1 is the *Introduction*, which briefly introduced the relevant fields. Section 2 contains *The Road to a Trustworthy 6G*. It investigates the technology basis for 6G, with elements inherited from 5G/5.5G and from the general technology trends. Section 3, *Agile Development and Cybersecurity*, outlines some high-level features of the Agile development methodologies, and explains why these are important for cybersecurity in the 6G software-based functions and services. Section 4, *Towards a Regime of "Zero Trust 6G"*, outlines the needs for an extended Zero Trust approach to 6G security. The paper is rounded off with Section 7, *Discussion and Summary*, and Section 8, *Conclusion*.

2 The Road to a Trustworthy 6G

The 6G system will inherit many features from 5G and 5.5G, and then add features and improvements of its own.

2.1 Future Technology Trends

As previously alluded to, the ITU-R Report M.2516-0 on the future technology trends [2] provides important insights into the basis for 6G.

2.1.1 New services and application trends

The M.2516-0 report highlight the following *key drivers* (section 4.2).

- Energy efficiency
- Data Rate, Latency and Jitter
- Sensing resolution and accuracy
- Connection density
- Coverage and full connectivity
- Mobility
- Spectrum utilization
- Simplified user-centric network
- Native Artificial Intelligence (AI)
- **Security/Trustworthiness**
- Dynamically controllable radio environment

All these drivers are posed to influence the 6G design. For our purpose, we shall only investigate the **Security/Trustworthiness** driver. That is, we note and concur that AI/ML will be integrated in almost all aspects of the 6G system.

2.1.2 Security/trustworthiness

Section 4.2 in the M.2516-0 report focuses on the security/trustworthiness driver [2]. We have that:

> A future network may support more advanced system resilience for reliable operation and service provision, security to provide confidentiality, integrity and availability, privacy with self-sovereign data and safety regarding the impact to the environment.
>
> *and*
>
> IMT towards 2030 and beyond should strive to support embedded end-to-end trust such that the level of information security in networks is

> significantly better than today's network. Trust modelling, trust policies and trust mechanisms need to be defined.
>
> *and*
>
> Security algorithms may use machine learning (ML) to identify attacks and respond to them. Continuous deep learning on a packet/byte level and machine learning can enforce policies, detect, contain, mitigate, and prevent threats or active attacks.

The above quotes fits well within a Zero Trust paradigm, but it still does not address software development aspects. Section 5.8 in the M.2516-0 report highlight technologies to enhance trustworthiness [2]. Here, it is noted that one

> ...depends on "trustworthiness technologies" such as security, privacy and resilience.

It is also noted that

> It is necessary to consider the extensive introduction of AI as well as the prospect of quantum computing.

Then there are subclauses that elucidates how these aspects might affect RAN privacy, how quantum technology might affect cipher solution and how physical-layer security (PLS) solutions might be

> ...used to enhance the resilience and robustness of IMT systems against active attacks at the physical layer.

All the above technology trends are of interest and may turn out to be useful and worthy additions to a 6G security architecture. They do not, however, take into account the fact that software will define and affect almost all aspects of a 6G ecosystem. There is thus a serious omission in the scope of the M.2516-0 report concerning how the IMT/6G system will be realized.

2.2 Fully Virtualized

The cybersecurity of 6G will not be dominated by traditional mobile system access security schemes. There will of course be security controls and cryptographic schemes specifically tailored to the 6G system and the mobile phones,

but, overall the security of the 6G systems will depend more on the security and reliability of the virtual network functions (software) and the management and orchestration of the system. Thus, much of the afforded security will be generic to the base technologies. As such, the 6G system will be a large and complex cloud-based web-technology system. Generic hardware (HW) will run the software, and the hardware will include accelerators for specific loads like ML-algorithms, etc.

2.3 From Cloud-ification to ML-ification

We observe that the drive towards virtualized and cloud-based network functions in 5G is all-encompassing. This drive to realize all functions as software-based will continue in 6G. However, while this was a defining aspect of 5G, for 6G this will be taken for granted. To fully benefit from a virtualized infrastructure, one needs automation and adaptability that scales.

Industrial-strength machine learning (ML) holds the promise of optimized and fast automation. This usage of ML for automation, also known as hyper-automation, is a general trend [9–11]. Fast adaptability, advanced fine-tuning of and fine-grained load-balancing will provide new opportunities for the network operators. This will enable better services at a lower energy cost, amongst others. These aspects were also clearly indicated in the ITU M.2516-0 technology report [2], and it seems evident that AI/ML technology will be a pervasive feature of 6G.

2.4 An Improved Triangle

The "IMT-2020 and Beyond" 5G triangle vision concept should be well-known [1]. The triangle is featured as Figure M.2083-02, and highlight these usage scenarios:

- Ultra-reliable and low latency communications (*urllc*)
- Massive machine type communications (*mmtc*)
- Enhanced mobile broadband (*emb*)

The 3GPP-based 5G system will realize these scenarios. However, the 5G version of the *urllc* and *mmtc* functions are far from perfect. The maturity and flexibility of these functions could no doubt be improved upon. Also, while the *emb* functionality is there, there is still room for improvements. Inevitably, one will want to reduce power consumptions even more, and to have more agility and flexibility. Finally, it would be advantageous to have more autonomous adaptability, and security needs to be pervasive, ubiquitous

and configurable. A major goal for 6G, whether stated or not, will therefore be to improve upon the 5G service triangle.

2.5 Features and Functions of 5.5G

3GPP Release 18 is known by many names, including "5G Advanced" and "5.5G". According to the 3GPP newsletter "Highlights" (November 2022) [12], the following topics are being studied (amongst others).

- **Study on System Support for AI/ML-based Services.** This work will expand on the Release 17 work on AI/ML for network automation, primarily focusing on transmission aspects.
- **Studies on Real-Time and Timing Resiliency.** These studies will focus on URLLC enhancements for synchronization and scheduling for extreme low latency support. There is also work on real-time communications for IMS.
- **Study on deterministic networking (DetNet).** This study will aim at providing real-time support (DetNet), as specified by IETF [13].
- **Various studies to include satellite access.**
- **Various studies to improve location services.**
- **Study on proximity services improvement**, etc.

There will also be continued focus on spectrum flexibility and energy savings. As 5G moves further towards ubiquity, the energy cost must come down and provisioning of broadband must become cheaper and more flexible. Many of the topics above can be seen as improvements to the service triangle, or as enabler technologies for improvements to the service triangle.

The Release 18 features will foreshadow what one can expect in the initial phase of a 6G architecture.

2.6 Cybersecurity as a Fundamental Premise

The current threat landscape points to many and highly competent adversaries the in cyber-domain. The ENISA annual Threat Landscape report highlights this [4]. As noted by ENISA, the threat actors are increasing their capabilities and geopolitical trend is towards more hybrid attacks. These trends are likely to become a norm. With this as the background, we postulate that cybersecurity must be a fundamental premise for the 6G systems. This is clearly in line with the inputs from the ITU M.2516-0 report [2]. We shall, however, take a broader perspective and address the role software development as critical component to achieving more resilient and trustworthy systems.

3 Agile Development and Cybersecurity

The Agile software model is a dominant software development methodology today. Suffice to say that the development methods used for the current 5G systems are largely based on agile methods [14, 15]. These systems are very much *service-oriented* and the software will likely feature "continuous integration and deployment" in some form. It is likely that 6G will be developed with the same basic methodologies.

3.1 Characteristics of the Agile Development Model

The agile development philosophy includes many aspects. Notably, there is the Agile manifesto that serves as high-level guideline.[1] This has been elaborated upon in several distinct methodologies, amongst them *Scrum* and *eXtreme Programming (XP)*. See [14–16] for more information. We shall not go into details here, but suffice to highlight the development cycles (sprints) and the CI/CD delivery methods.

3.1.1 Short development cycles

The development cycles are commonly called *sprints*. A sprint typically lasts about a month. The development process is highly *incremental*, and new services and functionally are added along the way.

3.1.2 Continuous integration/continuous delivery (CI/CD)

CI/CD is an automation method for software delivery. The main concepts of CI/CD are continuous integration, continuous delivery, and continuous deployment. The concept is associated with the so-called DevOps role (see clause 3.2). We note that the CI/CD automation comes with its own mix of opportunities and problems. These include regression testing, performance testing and security aspects [17,18]. Concerning the security aspects, security patches are commonly delivered by these automation methods. At its best, CI/CD provides a timely way to correct security flaws and bugs.

3.1.3 Agile methodology and security aspects

There are many aspects that influence the actual security level of software. The development practices will obviously influence the software, but it not obvious which aspects are inherent to the development method. There are numerous biases at play, and it is hard to measure whether, and to what

[1] See https://agilemanifesto.org/

extent, certain traits are affected by the development method [19,20]. We shall not assume any specific security benefits or drawback to agile development methods per se.

3.2 The DevOps and DevSecOps Concepts

The term DevOps was coined by agile developer Patrick Debois in 2009, and it denotes a methodology where one integrates software development and IT operations. Some of the aspects of DevOps are highlighted here [21, 22]. DevOps has been defined as "a set of practices intended to reduce the time between committing a change to a system and the change being placed into normal production, while ensuring high quality" [23].

DevSecOps is an extension to DevOps that includes security practices to be integrated into the DevOps approach. The *delivery team* will be empowered to set up the appropriate security controls in the software delivery. There have been concerns about observability concerning this process[2]. Security teams may also be reluctant to trust the developers to make security decisions.

4 Towards a Regime of "Zero Trust 6G"

We are proposing to include software development into our *Zero Trust 6G* concept. It will include ZT/ZTA as we currently know them, and then extend it to also cover software development (aka *Software Development Assurance*).

4.1 The Zero Trust Guiding Principles

There is no single universally agreed upon definition of the ZT principles. The industry consortium "The Open Group" has published the white-paper "Zero Trust Core Principles" [24], which gives a good overview. They also take partnerships, supply chain aspects and business models into account.

4.1.1 The zero trust architecture

The Zero Trust Architecture (ZTA) is defined in NIST Special Publication (SP) 800-207 [3]. A central characteristics of ZT and ZTA is being *explicit* and to never take assumptions for granted.

[2] Ref. https://devclass.com/2023/01/26/devsecops-report/

4.1.2 Zero trust guiding principles

The US National Security Agency (NSA) has publish a useful memo on the basic ZT security model [25]. The NSA memo advises to "Embrace Zero Trust guiding principles" (slightly amended):

- **Never trust, always verify** – Treat every user, device, application/workload, and data flow as untrusted. Authenticate and explicitly authorize the least privilege required using dynamic security policies.
- **Assume breach** – Operate and defend resources with the assumption that the system has been breached. Deny by default and scrutinize all users, devices, data flows, and requests for access. Log, inspect, and continuously monitor all configuration changes, resource accesses, and network traffic for suspicious activity.
- **Verify explicitly** – Access to resources should be conducted in a consistent and secure manner using multiple attributes (dynamic and static) to derive confidence levels for contextual access decisions to resources.

For our purpose, we want to take these OAM-related guidelines and characteristics one step further, and extend the ZT paradigm towards software development.

4.2 An Outline of the "Zero Trust 6G" Concept

We shall argue that the 6G systems will need to be designed, operated and maintained with security in mind.

4.2.1 Systematic approach

That will include a systematic approach to secure development practices. The methodology, whether or not based on agile principles, will need to include security perspectives all through the development phases. One may here include DevOps/DevSecOps practices. Security will of course also be part of the traditional OAM, and here Zero Trust principles and the Zero Trust Architecture concept comes into play. Operators will need to deploy and operate their networks according to ZTA concepts. We have not mentioned it yet, but ZTA commonly also comes with different degrees of maturity. There are several versions of this [26–28], although the big commercial software developers tend to have in-house version based on the US "Cybersecurity and Infrastructure Security Agency (CISA)" maturity model [27]. For critical infrastructures, we expect there to be requirements for an "advanced" or even "optimal" maturity level.

4.2.2 Assume flaws and bugs

Design flaws and implementation bugs[3] can be assumed to be a root cause of many vulnerabilities. There is therefore clearly a need for procedures and methods that facilitates secure software design methods and secure software implementation methods. A guiding principles, akin to the Zero Trust adage "Assume Breach", would be to "Assume Flaws and Bugs".

4.3 Secure Development Practices

For *Zero Trust 6G*, one needs to develop explicit and yet flexible practices. We shall not repeat the material covered in clause 3, suffice to say that agile methods will likely be part of *Zero Trust 6G*.

4.3.1 Security testing

Software flaws and bugs are an unwanted, but inherent, feature of software at large. However, it is still possible to reduce the amount and seriousness of the flaws and bugs, and thereby to reduce software vulnerabilities. There are three main strategies:

- Improve the design to avoid flaws
- Improve the implementation to avoid bugs
- Detection and Removal (through testing)

The design and implementation will involve tools and methods such as formal specification, formal modeling tools and various development methodologies, etc. There are heated debates about the benefits and drawbacks of the various programming languages. Proponents may claim that this or that programming language is safer, better, etc. That may be so, but one may write poor code in "safe" languages and, corollary, secure code in unsafe languages. For our purpose, we shall not delve in these matters, or for that matter software design and implementation methodologies. These are large and complex topics, and involves many disciplines and a multitude of aspects.

However, we shall propose that the developer use modern versions of whatever programming language they are using, and adhere to best current practices for that language. There exists many "secure practices" proposal, and the developers must document their choice of practices.

[3] Implementation errors in software designs are commonly call "bugs".

4.3.2 Threat modeling and security testing

Software testing is part and parcel of software development. Security testing can be viewed as included in the software testing, but we shall choose to treat it as an independent topic.

There exists frameworks for comprehensive software *security* testing regimes, such as the "OWASP Web Security Testing Guide" [29]. This framework includes various security testing methods, including penetration testing and threat modeling. Threat modeling is a design-phase companion to security testing, and these days it should be considered a mandatory requirement for critical software. Threat modeling, while technically independent of security testing, will be a useful way to focus on assets and associated threats. A decent resource on threat modeling methods and tools is found in the book "Threat Modeling: A Practical Guide for Development Teams" [30].

4.3.3 Engineering trustworthy secure systems

There have been many attempts at defining secure development practices, with different perspectives and priorities. As for engineering of trusted and secure systems, we find the NIST special publications on "Engineering Trustworthy Secure Systems" noteworthy [31,32]. These publication do not define development methods per se, but aim at providing tutorial-style guideline.

4.3.4 Software assurance

There are no shortage of software assurance and maturity models to choose from. We shall not investigate these, but suffice to say that one probably will want adherence to some such model. Unfortunately, many of the models are fairly complex and to demonstrate compliance one needs external auditors, etc. This is too expensive for many small development projects, and in practice impossible to achieve for Free and Open Source Software (FOSS). And, FOSS code is interwoven in almost all software projects these days.

A useful overview over secure software development standards can be found in [33]. There are numerous guidelines, standards, etc. for secure development practices [34–43].

4.4 AI/ML-assisted Development and Deployment

We have made a point of highlighting AI/ML as a pervasive technology in 6G. It stands to reason that AI/ML technologies will also become prevalent in software development. This will range from the mundane, as in assistance in programming styles and library usage, to more profound aspects such as correctness analysis, etc.

4.4.1 Avoiding well-known vulnerabilities and weaknesses

One area that should benefit is to let AI/ML-tools assist in analyzing the code for security vulnerabilities. This could range from known dangerous coding styles, and would certainly include analysis of known weaknesses and vulnerabilities. One can automate scanning for traits found in the Common Vulnerability and Exposure (CVE) database[4] or the in MITRE (cwe.mitre.org) Common Weakness Enumeration (CWE) database. Such tools will not entirely prevent weaknesses or vulnerabilities, but it should be feasible to avoid most known vulnerabilities. We expect this use of AI/ML to become the norm in professional software development.

4.4.2 AI/ML-assistance in design and development

To avoid vulnerabilities and flaws is all very well, but the AI/ML tools will also provide assistance in design decision by proposing appropriate design patterns. The tools will additionally propose code, or template code, for the task to carried out. The tools will be trained on large data sets and millions of lines of source code. The data sets can include meta-data, including statistics for usage and for flaws, bugs and fixes.

Some tools already exists, like Amazon's CodeWhisperer companion[5] and GitHub's CoPilot [6]. We expect these tools to evolve and become common in use. They will feature best common practices, including using safe coding patterns. As this field is relatively new, there does not yet exists many papers targeting AI-assisted development, but we are optimistic that these tools will permit developers to produce more and better code in less time [44, 45].

4.4.3 AI/ML-assistance to DevSecOps

The Gartner group have identified AIOps as top technology trend [46]. They define AIOps to combine big data and machine learning to automate IT operations processes. This will include event correlation, anomaly detection and causality determination.

4.4.4 Combating security fatigue

One has identified a condition where users security decisions deteriorates due to *security fatigue* [47, 48]. Security fatigue affects end-users, but also

[4]National Vulnerability Database (nve.nist.gov).
[5]https://aws.amazon.com/codewhisperer/
[6]https://github.com/features/copilot

software developers and IT professionals [49]. It has been suggested that AI/ML-based techniques may be effective in combatting this condition [50], and it seems likely that AI/ML-tools will be helpful in offloading the mental burden of getting all security settings and choices right.

4.4.5 AI/ML-assistance, accountability, transparency and observability

Even with effective use of AI/ML-assistance in both software development and in operations and management, there will be breaches. The original Zero Trust adages still applies. However, with so many tasks being completed by AI/ML-assistance, it will potentially be harder to analyze failures. That is, aspects such as accountability, transparency and observability will potentially suffer. The actual impact is hard to assess, but there is an urgent need for further study and research into these aspects.

4.5 Extended Security Testing

We have asserted that AI/ML-assistance will become the norm for software development. However, it does not stop there. Use of AI/ML-techniques and tools will also become the norm for software testing, including software security testing. These tools will range from tools that carry out security tests for known weaknesses and vulnerabilities, to automated fuzz testing on executable software [51] and all the way to AI/ML-guided automated penetration testing [52]. Our paper on input validation and fuzz testing [53] covers some of these aspects.

AI/ML-assisted security testing will ensure that many common types of weaknesses and vulnerabilities will be detected prior to deployment. These methods will surely be useful, but there will be no guarantees. To quote Edsger W. Dijkstra (from Jon Bentley's *Programming Pearls* [54]):

> Program testing can be used to show the presence of bugs, but never to show their absence!

Some of these methods will also make it into vulnerability scanners that will continue to scan for problems even after the software has been deployed. These advanced vulnerability scanner may be augmented and include knowledge about the source code, etc. One will then have a fair chance of detecting new and emerging threats at an early stage. These aspects will not be part of the development regime per se, but can be considered to be included in ZTA in the monitoring requirements.

5 Discussion and Summary

5.1 The Road to a Trustworthy 6G

In this paper we have examined some aspects of what it will take to make the 6G systems trustworthy. We have investigated what is currently known about 6G, and in particular we have taken the ITU-R report on future technology trends for IMT 2030 and beyond into account [2]. The report highlights drivers and technology trends, and in addition to the more obvious aspects such as enhanced mobility (in a multitude of ways), we have focused mostly on AI/ML as enabler technologies and on security and trustworthiness.

5.2 The Zero Trust Concept

The ZT concept applies primarily to operations and maintenance of large-scale networks (like a 6G network). The core of ZT is that there can be no assumed or implicit trust. The Zero Trust Architecture is a security architecture framework that takes the ZT philosophy into account. We recommend ZT/ZTA to be fully integrated into the 6G networks.

5.3 A Software-based 6G

The 5G systems features software defined networking (SDN) and network function visualization (NFV), which are software defined through and through. Technologies such as the Service Based Architecture (SBA) are also very much defined by software, Web-technologies and Cloud-based technologies. The 6G systems will continue the trend, and it will be even more dependent on software-defined functionality. The hardware platforms used in 6G will be generic and will largely be tailored to support cloud-based systems. The current trend is for software to be developed with agile development methods. The agile methods feature aspects such as *sprints* and continuous integration, and continuous delivery/deployment (CI/CD). Associated with these methods is the DevOps and DevSecOps way of handling the automation (CI/CD).

Thus, the security and trustworthiness of the 6G system will largely be decided by the software it contains. This makes software quality of utmost importance to a trustworthy 6G system.

5.4 The “Zero Trust 6G” Concept

In this paper we have argued for the need for a *Zero Trust 6G* concept for the 6G system. The concept will include *software development assurance (SDA)*

requirements. We have not fully defined what this might encompass or how it might be realized. The following features will be included in the concept:

- ZT/ZTA for operation and maintenance of 6G systems
- Software Development Assurance
- AI/ML-methods as enablers for the above

5.5 Further Study

It is obvious that the *Zero Trust 6G* concept will need to be investigated further, and that the realization of the SDA part needs further study. While this is a shortcoming of the proposal, we note that it is still early days when it comes to AI/ML-assistance methods and tools. It would there be premature to rigourously define requirements on how SDA can be achieved. Suffice to note that we expect great strides to be made during the next couple of years.

5.6 The Double-Edged Sword

Software development in the "...2030 and Beyond..." era will no doubt be heavily aided by AI/ML tools. These tools will become efficient, effective, and reliable and will potentially produce code with far fewer problems. But, of course, AI/ML-tools will also be used for offensive security purposes. These tools will scan code, binary or source-code, for vulnerabilities, and automatically select attack methods accordingly. In this sense, there is an AI/ML arms race for cybersecurity tools and methods. It is also a race that we can ill afford to lose.

6 Conclusion

In this paper we have outlined some of the drivers and developments that will shape 6G. Security and trustworthiness are highlighted as a key driver, which is no surprise considering how important the mobile systems have become. Software is a defining aspects of ICT infrastructures, and the 6G systems will be no different. The proposed *EU Cyber Resilience Act*, amongst others, addresses the reliability and security of software, with implications and repercussions for software development.

There will be many measures that must be taken to achieve a secure and full trustworthy 6G network. These range from providing (quantum-safe) robust and effective security controls, comprehensive operation- and management schemes and all the way to providing trustworthy software. That is why

we have proposed a *Zero Trust 6G* paradigm for the 6G systems. Here one will adapt and amend Zero Trust concepts and the Zero Trust Architecture to fit with the 6G system architecture. AI/ML-methods will be used throughout to enhance and automate the security defenses. The *Zero Trust 6G* concept also include requirements for *software development assurance* to achieve the security-level needed for a critical infrastructure. AI/ML-tools will be essential to achieve this goal. Currently, the AI/ML-tools are in their infancy, but we believe that the tools will improve immensely during the next few years.

References

[1] ITU-R. IMT Vision – Framework and overall objectives of the future development of IMT for 2020 and beyond. Recommendation M.2083-0, ITU-R, 09 2015.

[2] ITU-R. Future technology trends of terrestrial International Mobile Telecommunications systems towards 2030 and beyond. Report M.2516-0, ITU, 11 2022.

[3] Scott Rose, Oliver Borchert, Stu Mitchell, and Sean Connelly. Zero Trust Architecture. Special Publication 800-207, NIST, 08 2020.

[4] Ifigeneia Lella, Eleni Tsekmezoglou, Rossen Svetozarov Naydenov, Cosmin Ciobanu, Apostolos Malatras, and Marianthi Theocharidou (eds). ENISA THREAT LANDSCAPE 2022; July 2021 to July 2022. Report, ENISA, 11 2022.

[5] Andrew van der Stock, Brian Glas, Neil Smithline, and Torsten Gigler. OWASP Top 10 – 2021. https://owasp.org/Top10/, 09 2021.

[6] EU. EU Cybersecuirty Act. Regulation, EU, 04 2019.

[7] EU. EU Cyber Resilience Act. Proposal, EU, 09 2022.

[8] EU. EU Digital Services Act. Regulation, EU, 10 2022.

[9] Sukhpal Singh Gill, Minxian Xu, Carlo Ottaviani, Panos Patros, Rami Bahsoon, Arash Shaghaghi, Muhammed Golec, Vlado Stankovski, Huaming Wu, Ajith Abraham, et al. Ai for next generation computing: Emerging trends and future directions. *Internet of Things*, 19:100514, 2022.

[10] Fausto Artico, Arthur L Edge III, and Kyle Langham. The future of artificial intelligence for the biotech big data landscape. *Current Opinion in Biotechnology*, 76:102714, 2022.

[11] Abid Haleem, Mohd Javaid, Ravi Pratap Singh, Shanay Rab, and Rajiv Suman. Hyperautomation for the enhancement of automation in industries. *Sensors International*, 2:100124, 2021.

[12] 3GPP. Highlights. Newsletter Issue 05, 3GPP, 11 2022.

[13] B. Varga, J. Farkas, L. Berger, A. Malis, and S. Bryant. Deterministic Networking (DetNet) Data Plane Framework. Informational RFC 8938, IETF, 11 2020.

[14] James Taylor Faria Chaves and Sergio Antônio Andrade de Freitas. A systematic literature review for service-oriented architecture and agile development. In Sanjay Misra, Osvaldo Gervasi, Beniamino Murgante, Elena Stankova, Vladimir Korkhov, Carmelo Torre, Ana Maria A.C. Rocha, David Taniar, Bernady O. Apduhan, and Eufemia Tarantino, editors, *Computational Science and Its Applications – ICCSA 2019*, pages 120–135. Springer International Publishing, 2019.

[15] Torgeir Dingsøyr, Tore Dybå, and Nils Brede Moe. *Agile software development: current research and future directions*. Springer Science & Business Media, 2010.

[16] Henry Edison, Xiaofeng Wang, and Kieran Conboy. Comparing methods for large-scale agile software development: A systematic literature review. *IEEE Transactions on Software Engineering*, 48(8):2709–2731, 2022.

[17] Mohammad Rizky Pratama and Dana Sulistiyo Kusumo. Implementation of continuous integration and continuous delivery (ci/cd) on automatic performance testing. In *2021 9th International Conference on Information and Communication Technology (ICoICT)*, pages 230–235, 2021.

[18] Eliezio Soares, Gustavo Sizilio, Jadson Santos, Daniel Alencar da Costa, and Uirá Kulesza. The effects of continuous integration on software development: a systematic literature review. *Empirical Software Engineering*, 27(3):78, 2022.

[19] Kalle Rindell, Jukka Ruohonen, Johannes Holvitie, Sami Hyrynsalmi, and Ville Leppänen. Security in agile software development: A practitioner survey. *Information and Software Technology*, 131:106488, 2021.

[20] Maria Ilaria Lunesu, Roberto Tonelli, Lodovica Marchesi, and Michele Marchesi. Assessing the risk of software development in agile methodologies using simulation. *IEEE Access*, 9:134240–134258, 2021.

[21] Alberto Avritzer. Challenges and approaches for the assessment of micro-service architecture deployment alternatives in devops : A tutorial

presented at icsa 2020. In *2020 IEEE International Conference on Software Architecture Companion (ICSA-C)*, pages 1–2, 2020.

[22] Sébastien Mosser and Jean-Michel Bruel. Requirements engineering in the devops era. In *2021 IEEE 29th International Requirements Engineering Conference (RE)*, pages 510–511, 2021.

[23] Len Bass, Ingo Weber, and Liming Zhu. *DevOps: A software architect's perspective*. Addison-Wesley Professional, 2015.

[24] Tuhinshubhra Ghosh, Nikhil Kumar, Sai Mohan Sakuru, Patrick Shirazi, Mark Simos, Altaz Valani, Anthony Carrato, Stephen Whitlock, Jim Hietala, John Linford, and Andras Szakal. Zero Trust Core Principles. Whitepaper W210, The Open Group, 04 2021.

[25] NSA. Embracing a Zero Trust Security Model. Memo U/OO/115131-21 | PP-21-0191 | February 2021 Ver. 1.0, NSA, 02 2021.

[26] Raj Badhwar. *CISO Maturity ModelCISO maturity model (CMM)*, pages 29–37. Springer International Publishing, Cham, 2021.

[27] US Cybersecurity & Infrastructure Security Agency. Zero Trust Maturity Model. Pre-decisional draft, CISA, 06 2021.

[28] Belal Ali, Simsam Hijjawi, Leith H Campbell, Mark A Gregory, and Shuo Li. A maturity framework for zero-trust security in multiaccess edge computing. *Security and Communication Networks*, 2022, 2022.

[29] Elie Saad and Rick Mitchell. OWASP Web Security Testing Guide; Version 4.2. OWASP Webpage, 12 2020.

[30] I. Tarandach and M.J. Coles. *Threat Modeling: A Practical Guide for Development Teams*. O'Reilly Media, Incorporated, 2020.

[31] Ron Ross, Mark Winstead, and Michael McEvilley. Engineering Trustworthy Secure Systems. Special Publication 800-160v1r1, NIST, 11 2022.

[32] Ron Ross, Victoria Pillitteri, Richard Graubart, Deborah Bodeau, and Rosalie McQuuaid. Engineering Trustworthy Secure Systems. Special Publication 800-160v2r1, NIST, 12 2021.

[33] Armando Ramirez, Anthony Aiello, and Susan J Lincke. A survey and comparison of secure software development standards. In *2020 13th CMI Conference on Cybersecurity and Privacy (CMI) – Digital Transformation – Potentials and Challenges(51275)*, pages 1–6, 2020.

[34] Hala Assal and Sonia Chiasson. 'think secure from the beginning': A survey with software developers. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–13, New York, NY, USA, 2019. Association for Computing Machinery.

[35] Jangirala Srinivas, Ashok Kumar Das, and Neeraj Kumar. Government regulations in cyber security: Framework, standards and recommendations. *Future Generation Computer Systems*, 92:178–188, 2019.

[36] Mahmood Niazi, Ashraf Mohammed Saeed, Mohammad Alshayeb, Sajjad Mahmood, and Saad Zafar. A maturity model for secure requirements engineering. *Computers & Security*, 95:101852, 2020.

[37] Rafiq Ahmad Khan, Siffat Ullah Khan, Habib Ullah Khan, and Muhammad Ilyas. Systematic mapping study on security approaches in secure software engineering. *IEEE Access*, 9:19139–19160, 2021.

[38] Murugiah Souppaya, Karen Scarfone, and Donna Dodson. Secure Software Development Framework (SSDF) Version 1.1: Recommendations for Mitigating the Risk of Software Vulnerabilities. Special Publication 800-218v1r1, NIST, 02 2022.

[39] Bart De Win, Riccardo Scandariato, Koen Buyens, Johan Grégoire, and Wouter Joosen. On the secure software development process: Clasp, sdl and touchpoints compared. *Information and Software Technology*, 51(7):1152–1171, 2009. Special Section: Software Engineering for Secure Systems.

[40] Nor Shahriza Abdul Karim, Arwa Albuolayan, Tanzila Saba, and Amjad Rehman. The practice of secure software development in sdlc: an investigation through existing model and a case study. *Security and Communication Networks*, 9(18):5333–5345, 2016.

[41] Rafiq Ahmad Khan, Siffat Ullah Khan, Habib Ullah Khan, and Muhammad Ilyas. Systematic literature review on security risks and its practices in secure software development. *IEEE Access*, 10:5456–5481, 2022.

[42] Shams Al-Amin, Nirav Ajmeri, Hongying Du, Emily Z. Berglund, and Munindar P. Singh. Toward effective adoption of secure software development practices. *Simulation Modelling Practice and Theory*, 85:33–46, 2018.

[43] Lynn Futcher and Rossouw von Solms. Guidelines for secure software development. In *Proceedings of the 2008 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists on IT Research in Developing Countries: Riding the Wave of Technology*, SAICSIT '08, page 56–65, New York, NY, USA, 2008. Association for Computing Machinery.

[44] Binayak Parashar, Inderjeet Kaur, Anupama Sharma, Pratima Singh, and Deepti Mishra. Revolutionary transformations in twentieth century: making ai-assisted software development. *Computational Intelligence in Software Modeling*, 13(1), 2022.

[45] Sumit Gulwani. Ai-assisted programming: Applications, user experiences, and neuro-symbolic techniques (keynote). In *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2022, page 1, New York, NY, USA, 2022. Association for Computing Machinery.

[46] Gartner. Top strategic technology trends for 2022. eBook, 2022.

[47] Steven Furnell and Kerry-Lynn Thomson. Recognising and addressing "security fatigue". *Computer Fraud & Security*, 2009(11):7–11, 2009.

[48] Brian Stanton, Mary F. Theofanos, Sandra Spickard Prettyman, and Susanne Furman. Security fatigue. *IT Professional*, 18(5):26–32, 2016.

[49] W Alec Cram, Jeffrey G Proudfoot, and John D'Arcy. When enough is enough: Investigating the antecedents and consequences of information security fatigue. *Information Systems Journal*, 31(4):521–549, 2021.

[50] Tao Ban, Ndichu Samuel, Takeshi Takahashi, and Daisuke Inoue. Combat security alert fatigue with ai-assisted techniques. In *Cyber Security Experimentation and Test Workshop*, CSET '21, page 9–16, New York, NY, USA, 2021. Association for Computing Machinery.

[51] Dimitrios Serpanos and Konstantinos Katsigiannis. Fuzzing: Cyber-physical system testing for security and dependability. *Computer*, 54(9):86–89, 2021.

[52] László Erdődi, Åvald Åslaugson Sommervoll, and Fabio Massimo Zennaro. Simulating sql injection vulnerability exploitation using q-learning reinforcement learning agents. *Journal of Information Security and Applications*, 61:102903, 2021.

[53] Geir Køien and Lasse Øverlier. A call for mandatory input validation and fuzz testing. *Wireless Personal Communications*, *Accepted for publication*, 2023.

[54] Jon Bently. Programming pearls; bumper-stricker computer science. *Communications of the ACM*, 28(9):896–901, September 1985.

Biography

Geir M. Køien received his PhD from Aalborg University, on access security for mobile systems. He has also worked for many years in industry, including for LM Ericsson Norway and Telenor R&D. During these years he worked extensively with mobile systems, and with security and privacy. He has also worked with the Norwegian Defence Research Establishment and with Norwegian Communications Authority on various security and communications related projects. Currently, he is a professor with the University of South-Eastern Norway (USN).