This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

Privacy preserving federated machine learning and blockchaining for reduced cyber risks in a world of distributed healthcare

Project coordinator:
Professor Dr Jan Baumbach, TECHNISCHE UNIVERSITAET MUENCHEN (TUM)
info@featurecloud.eu

Deliverable D2.2
"KPIs and metrics for local execution platforms"

Work package WP2
"Cyber risk assessment and mitigation"

Disclaimer

Copyright message

Document information

Grant Agreement Number 826078			**Acronym FeatureCloud**	
Full title	Privacy preserving federated machine learning and blockchaining for reduced cyber risks in a world of distributed healthcare			
Topic	Toolkit for assessing and reducing cyber risks in hospitals and care centres to protect privacy/data/infrastructures			
Funding scheme	RIA - Research and Innovation action			
Start Date	1 January 2019	**Duration**	60 months	
Project URL	https://featurecloud.eu/			
EU Project Officer	Reza RAZAVI (CNECT/H/03)			
Project Coordinator	Jan Baumbach, TECHNISCHE UNIVERSITAET MUENCHEN (TUM)			
Deliverable	“KPIs and metrics for local execution platforms”			
Work Package	WP2 “Cyber risk assessment”			
Date of Delivery	**Contractual**	M16 (30/04/20)	**Actual**	M22 (02/10/20)
Nature	REPORT	**Dissemination Level**	PUBLIC	
Lead Beneficiary	05 SBA Research			
Responsible Author(s)	Rudolf Mayer (SBA), Jan Baumbach, Julian Matschinske , Julian Späth, Reihaneh Torkzadehmahani, Reza Nasirigerdeh (TUM)			
Keywords	Key performance indicator, Data security, Privacy leakage			

Table of Content

1 Related objectives of the Description of Action 5

2 Executive Summary 5

2.1 Methodology 5

2.2 Main results 5

2.3 Progress beyond the state-of-the-art 5

3 Deliverable report 6

3.1 Challenge 6

3.2 Methodology 7

3.3 Key Performance Indicators 7

3.3.1 No global service inside hospitals (KPI 1) 7

3.3.2 Common Vulnerability Scoring System (KPI 2) 9

Local Scenario 10

Remote Scenario 10

3.3.3 Privacy Requirements for Federated Algorithms Working on Patient Data (KPI 3) 11

3.3.4 Encryption Requirements for Patient Data and Network Traffic (KPI 4) 12

4 Conclusion 14

5 References 14

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

Acronyms and definitions

3DES	Triple Data Encryption Algorithm
A	Availability
AC	attack complexity
AES	Advanced Encryption Standard
API	application programming interface
AV	attack vector
concentris	concentris research management GmbH
CVSS	Common Vulnerability Scoring System
DDOS	distributed denial-of-service
DES	Data Encryption Standard
FIRST	Forum of Incident Response and Security Technology
GND	Gnome Design SRL
H	HIgh
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
I	Integrity
IT	information technology
KPIs	Key Performance Indicator
L	local
MUG	Medizinische Universitaet Graz
N	None
patients	In this deliverable, we use the term "patients" for all research subjects. In FeatureCloud, we will focus on patients, as this is already the most vulnerable case scenario and this is where most primary data is available to us. Admittedly, some research subjects participate in clinical trials but not as patients but as healthy individuals, usually on a voluntary basis and are therefore not dependent on the physicians who care for them. Thus to increase readability, we simply refer to them as "patients".
PR	privileges required
R	Required
RI	Research Institute AG & Co. KG
RSA	Rivest–Shamir–Adleman
SBA	SBA Research Gemeinnutzige GmbH
SDU	Syddansk Universitet
SSL	Secure Sockets Layer
TLS	Transport Layer Security
TUM	Technische Universitaet Muenchen
UI	user interaction
UM	Universiteit Maastricht
UMR	Philipps Universitaet Marburg
USB	universal serial bus
WP	work package

1 Related objectives of the Description of Action

WP2, Objective 2, Task 2: One major aspect of the approach described in this project is the federated concept, i.e. pushing the algorithms to local execution platforms inside the data holders' premises, calculating the feature vectors there and subsequently pushing them back to the overall platform. While this approach is excellent from a privacy point of view, it puts responsibility on the local execution platforms inside the partners, e.g. hospitals. Thus, in this deliverable, TUM has developed Key Performance indicators (KPIs) and metrics for measuring the security of these local analysis platforms in order to provide systematic minimal requirements for the platform and its requirements imposed on federated applications, and detecting potentially dangerous installations.

2 Executive Summary

2.1 Methodology

In FeatureCloud, the security of the local execution platform depends on the technologies used, the design of the architecture and the restrictions imposed on federated algorithms being executed by it. To a large extent, state-of-the-art approaches can be used and applied to a federated context, considering the specialties of FeatureCloud, such as extra sensitivity of patient data and additional challenges due to the federated nature of the overall system.

Therefore, we looked at comparable software systems that need to have network access and access to the internet in particular. Such systems include database systems, web servers, in-memory caches and other web-related software systems. We investigated common security problems these systems can suffer from and how they are mitigated. Applicable mitigations were then adapted to the FeatureCloud context and an acceptance criteria has been shaped out of it to create testable KPIs. All candidate KPIs have been discussed with the whole consortium in regular online meetings, and this way have been extended and adopted until application readiness became foreseeable, considering legal, technical and algorithmic points of view.

2.2 Main results

We present four metrics and requirements that cover integral aspects of the confidentiality of patient data and maintenance of system integrity, namely

- No global service inside hospitals (KPI 1)
- Common vulnerability scoring system (KPI 2)
- Privacy requirements for federated algorithms working on patient data (KPI 3)
- Encryption requirements for patient data and network traffic (KPI 4)

Apart from KPI 3, these KPIs do not just apply to federated systems, but to all systems dealing with sensitive data. However, they play an extraordinarily important role in a federated context, involving lots of communication among the participating sites and dealing with sensitive patient data.

2.3 Progress beyond the state-of-the-art

The KPIs largely follow state-of-the-art guidelines for cyber security, which is crucial for a high level of security. However, with KPI 3 we present a metric that can be tested automatically and is specific for federated algorithms.

3 Deliverable report

3.1 Challenge

One major aspect of FeatureCloud is the federated concept, i.e. pushing the algorithms to local execution platforms inside the data holders’ premises, calculating the feature vectors there and subsequently pushing them back to the overall platform, possibly in an iterative process going back and forth until convergence has been reached (Yang et al., 2019).

While the federated approach is beneficial from a privacy point of view, it puts responsibility on the local execution platforms inside the partners, e.g. hospitals. Broadly speaking, there are two aspects to be considered (Bertino et al., 2018):

- Data security
- Privacy leakage

The goal of the KPIs presented in this report is to measure how well data security is ensured and privacy leakage is mitigated.

To understand the KPIs, it is necessary to introduce the drafted system architecture of FeatureCloud (see Figure 1). In particular, the system parts running on the local machines of hospitals are crucial.

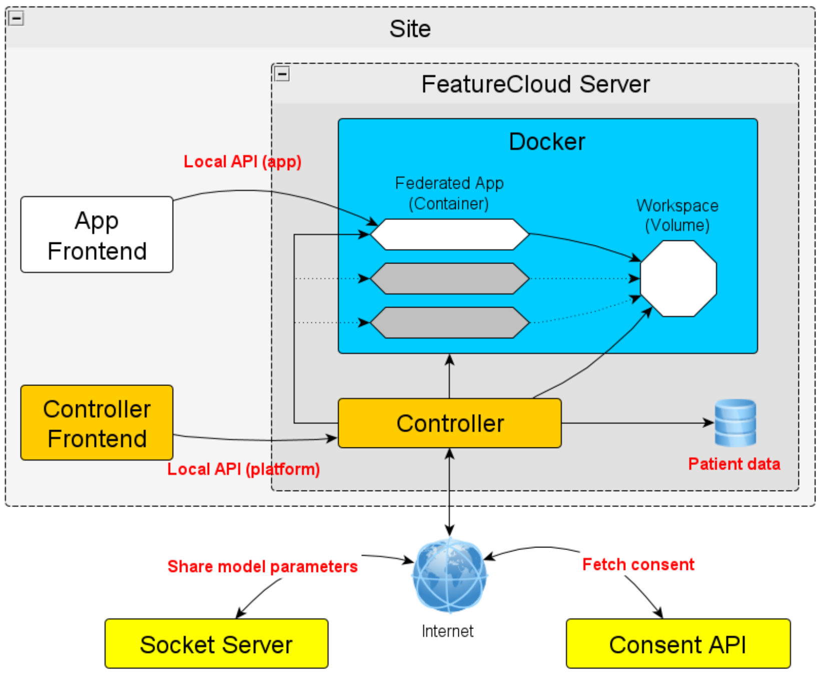

Fig. 1: *Local platform running in a hospital and its outside connections*

Inside a hospital (here called site), the local FeatureCloud platform is being run on a dedicated machine (here called FeatureCloud server). On this server, the platform uses OS-level virtualisation in the form of docker containers to run federated applications. Since patients need to give their consent in order to be included in a study, the controller needs to query a consent Application programming interface (API) for each potential patient. This consent API is either a global server or a distributed ledger such as a blockchain, storing patient consent in an anonymous manner. If patient consent is given, the respective data can be used from the patient data database. Federated applications are part of a workflow where each application can output data for the next application to be processed. This is done with docker volumes that are passed from each federated application container to the next. The whole platform can be controlled via a web application (here called Controller Frontend) involving user rights management with various permission levels.

The socket server is responsible for relaying model parameters from the participating hospitals (clients) to the coordinating hospital (server) and vice versa. Notice that the controller is running as a local service and the other local parts (e.g. App frontend and controller frontend) communicate with the controller within the local network. The local service from the controller is not reachable from the other parts of the system running outside the local machine of the hospital (e.g. socket server and consent API). That is, they cannot send a request (GET or POST) to the controller and only the controller initiates a request to consent API or socket server and they provide the controller with the data it wants in response. The aim of this communication paradigm is to minimize security risks such as vulnerability to DDoS or evasion attacks (see D2.1, section 8.3 and KPI 1). Since we cannot allow external services to connect to a server application running inside a hospital network, the socket server needs to be outside of it, so that all participants can send a query (HTTP request) to it.

3.2 Methodology

In this report, we require KPIs to be relevant to data security or privacy, exact, testable and achievable.

We distinguish between two sorts of KPIs:

- Binary KPIs
- Quantitative KPIs

The requirements for binary KPIs are either fulfilled or missed. Therefore, they always contain an acceptance criterion that specifies when the requirement is fulfilled.

Quantitative KPIs try to assess how well a certain goal is met. It can be combined with a threshold which is chosen in such a way that it contributes as much as possible to its goal while still keeping the platform operable and usable.

3.3 Key Performance Indicators

There are already privacy metrics (indicators) defined and published by different organizations and researchers (e.g. *Wagner et al., 2018*). However, those metrics are dependent on the algorithm and dataset and are not directly applicable to the FeatureCloud platform, which needs metrics that are general enough to be applied to any federated application and healthcare data leveraged in the study. To this end, we define four KPIs for the platform that are described in the subsequent sections.

3.3.1 No global service inside hospitals (KPI 1)

Running global services (e.g. HTTP service on port 80) in the local machine of a hospital is not a good security practice. By global service, we mean a service that is accessible from outside (i.e. Internet). They can reveal information about the target machine and make the site vulnerable to masquerading and denial of service attacks. Secondly, they allow other programs to actively connect to them and exploit potential vulnerabilities by running malicious code inside the machine providing the global service (also see D2.1, section 7).

For this reason, it is often not even possible to run a global service in a hospital context because the network layout does not allow for it (see fig. 2). More precisely, IT policies in hospitals often explicitly disallow running global service on ports accessible from outside as a general principle. Thus, while it serves a higher level of security (see threats/mitigation), this KPI also fulfils a strict requirement imposed by hospitals themselves and other less drastic solutions are not pondered.

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 826078.

Instead, a socket server relays the model parameter values from the participating sites to the coordinating site and vice versa (Fig. 1). Even though these parameters are much less problematic than the raw data, security and authentication mechanisms are in place to ensure only entitled parties can connect. These mechanisms include using authentication tokens distributed to the sites by the coordinator and encrypted communication.

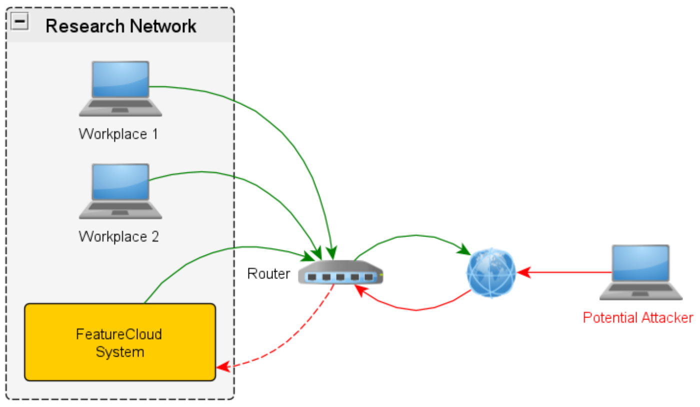

Fig. 2: Potential external attacker

Threats

Revealing information about the system (reconnaissance)

One way of gathering information about the target system can be done by performing a port scan on the target host. It tries to find out which ports are open and which kind of program is listening for connections (global service). This can reveal information that is then used in subsequent attacks, e.g to perform a (distributed) denial-of-service (DDOS) attack (also see D2.1, section 7) or, even worse, exploit vulnerabilities (security holes) to cause a data breach.

(D)DOS attacks

Denial-of-service attacks aim to take down a server (reduce the availability of a service) by bombarding it with a high number of requests (in a short period of time) too large for the server to handle. Normally, these attacks are made in order to sabotage or blackmail service providers. In a distributed denial of service attack , a network of attackers performs these requests, potentially from multiple locations so that there is no easy way of telling malicious and benign requests apart.

Exploits and data breaches

If an open port has been found, a potential attacker can try to find a security vulnerability (“security hole”) that can be exploited. Since the FeatureCloud platform is run by the hospitals and thus different local administrators, it is difficult to ensure that the platform instances are all kept up-to-date. This means that not only zero-day exploits (also see D2.1, section 4.1) pose a problem, but also known exploits that have already been addressed.

Mitigation

The most efficient way of mitigating the above-mentioned threats is to pursue a security-by-design approach; in this case, designing the system architecture so that hospital services can only run local services, which is not accessible from outside.

This entails several implications for the architecture and API design. Communication between participating hospitals needs to be established through a message broker (aka socket server, see fig. 1) to which the hospitals can send a request, rather than directly connecting to another hospital, which is not possible because it cannot run a global service for the reasons stated earlier. This complicates the architecture slightly since it makes a new system component necessary. It does not decrease the diversity of machine learning algorithms that can be developed and has therefore no further disadvantages.

Acceptance Criteria

This KPI is a binary KPI and is fulfilled, if none of the system components that run on a hospital machine provide a global service to outside. This can be verified by appropriate tools (e.g. port scan tools).

3.3.2 Common Vulnerability Scoring System (KPI 2)

The Common Vulnerability Scoring System (CVSS) is an open industry standard providing a way to capture the principal characteristics of a vulnerability and assessing the severity in a numerical score. It has been developed by the Forum of Incident Response and Security Teams (FIRST). The most recent version (04/2020) is currently 3.1 and can be found at https://www.first.org/cvss/.

Scores range from 0 (least severe) to 10 (most severe) and depend on several metrics that approximate the ease and impact of an exploit. In the end, the scores allow the prioritization of vulnerabilities according to a threat. Numerical scores can also be converted to a qualitative rating in a textual representation:

- 0.0 = None
- 0.1 - 3.9 = Low
- 4.0 - 6.9 = Medium
- 7.0 - 8.9 = High

9.0 - 10.0 = Critical

Metrics and scoring

The CVSS is composed of three different metrics groups, each including a set of metrics. To compute the score of the system, a security analyst or vendor assigns values to the metrics of the Base metric group first.

The *Base metric group* represents the intrinsic characteristics of a vulnerability that are constant over time and across user environments:

1. Exploitability metrics
   - Attack Vector (Network, Adjacent, Local, Physical)
   - Attack Complexity (Low, High)
   - Privileges Required (None, Low, High)
   - User interaction (None, Required)
2. Scope (Unchanged, Changed)
3. Impact metrics (see also D2.1 section 8.3 and 8.4)
   - Confidentiality (High, Low, None)
   - Integrity (High, Low, None)
   - Availability (High, Low, None)

In the next optional steps, the analysis can then refine the base score by adding temporal and environmental metrics.

The *Temporal metric group* reflects the characteristics of a vulnerability that change over time:

1. Exploit Code Maturity (Not Defined, High, Functional, Proof-of-Concept, Unproven)
2. Remediation Level (Not Defined, Unavailable, Workaround, Temporary Fix, Official Fix)
3. Report Confidence (Not Defined, Confirmed, Reasonable, Unknown)

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

The *Environmental metric group represents* the characteristics of a vulnerability that are relevant and unique to a particular user's environment:

1. Security Requirements (Not Defined, High, Medium, Low)
2. Modified Base Metric

The assignment of values to the metrics results in a CVSS v3.1 vector string, which is used to record or transfer the metric information in a concise form.

CVSS v3.1 Equations

After assigning the metric values, the CVSS v3.1 score can be computed using the CVSS 3.1 equations. As an example, the base metric score is computed as follows:

Impact =

$6.42 \times ISS$, if Scope is Unchanged

$7.52 \times (ISS - 0.029) - 3.25 \times (ISS - 0.02)^{15}$, if Scope is Changed

Exploitability = $8.22 \times$ AttackVector $\times$ AttackComplexity $\times$ PrivilegesRequired $\times$ UserInteraction

BaseScore =

0, if Impact <= 0

Roundup (Minimum [(Impact + Exploitability), 10]), if Scope is Unchanged

Roundup (Minimum [$1.08 \times$ (Impact + Exploitability), 10]), if Scope is Changed

Example Scenarios

Local Scenario

In FeatureCloud, a medical doctor has access to the user frontend. In this example, it would be possible for an attacker to steal information via a USB stick, that a doctor sticks into a local machine and the attacker stealing sensitive data (e.g., raw patient data). This scenario would result in the following Base metric values:

- Attack Vector (AV): Local (L) - 0.55
- Attack Complexity(AC): High (H) - 0.44
- Privileges Required (PR): High (H) - 0.27
- User Interaction (UI): Required (R) - 0.62
- Scope (S): Unchanged (U)
- Confidentiality (C): High (H) - 0.56
- Integrity (I): None (N) - 0
- Availability (A): None (N) - 0

The corresponding vector string would be CVSS:3.1/AV:L/AC:H/PR:H/UI:R/S:U/C:H/I:N/A:N. This scenario would conform to a CVSS score of 4.0 (medium).

Remote Scenario

In this example, an attacker can disturb the communication between platforms by sending a lot of requests to one or some of the platforms (DDos attack according to D2.1, section 7). The scenario would result in the following base metric values:

- Attack Vector (AV): Adjacent (A) - 0.62
- Attack Complexity(AC): Low (L) - 0.77
- Privileges Required (PR): None (N) - 0.85
- User Interaction (UI): None (N) - 0.85

- Scope (S): Unchanged (U)
- Confidentiality (C): High (H) - 0
- Integrity (I): None (N) - 0
- Availability (A): None (H) - 0.56

This scenario would amount to a CVSS score of 6.5 (medium).

Severity Rating Threshold

Since the CVSS score is a quantitative KPI, we need to specify the goal threshold to meet. For the FeatureCloud platform, we generally aim to rule out all vulnerabilities with a CVSS score larger than to 3.9, which corresponds to a None or Low severity in the qualitative rating.

Furthermore, we add one exception to the general threshold. In the case of confidentiality, which means that there is a potential security flaw leaking patient data, we do not accept any vulnerabilities except if physical access is involved.

Summary

Using the CVSS as a KPI for FeatureCloud enables a consistent measuring of a local site's vulnerability and makes it comparable between different sites. A big advantage of the CVSS is the various metrics it takes into account. It supports both a numerical and a qualitative score to score the vulnerability of systems. By defining a secure enough upper threshold, e.g. 3.9 (Low), FeatureCloud can ensure that enough reasonable security actions have been taken to protect the sensitive data from attackers. The exact threshold needs to be evaluated in future, but in general, it should be as low as possible. Employing CVSS for FeatureCloud can help to get a first impression of the vulnerability of the platform in a measurable form. However, even with a low score, it cannot guarantee that all vulnerabilities have been found. Additional measures and concepts will still be necessary.

3.3.3 Privacy Requirements for Federated Algorithms Working on Patient Data (KPI 3)

The only information being shared between hospital platforms are model parameters. This, however, does not guarantee that no private information can be retrieved from this data (Li et al.,2019), as has been detailed also in Deliverable D2.1, Risk Assessment Methodology. The application developers need to ensure that the information they send around is not sensitive (containing patients’ private data) and this is also being checked again manually by a trained personnel of a certification authority before the application becomes available in the app store.

Yet, it is worth looking at how the data being sent around relates to the private raw data. If the amount being sent around is as large as the amount of private data that is used for the learning of the model parameters itself, this might point at a problem.

O-Notation

Therefore, a different measurement is used as a KPI here, which resembles the big-O notation known from computational complexity theory (Papadimitriou et al., 2003). Instead of simply comparing the amount of patient data with the amount of exchanged data, we investigate how the amount of exchanged data depends on the amount of patient data.

An application whose amount of shared data grows proportionally or even over-proportionally to the amount of patient data most probably does not use a valid form of aggregation that complies with privacy requirements. By definition, an aggregation reduces a set of values to a single data object, independently of the number of elements in the value set. For this reason, in big O notation, federated

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 826078.

applications must not have a shared-to-raw-data dependency larger or equal to O(p), with p being the amount of private data. Ideally, they have a dependency of O(1).

This behaviour can be tested automatically. To do that, one would need to create at least three chunks of artificial patient data of different sizes s1, s2 and s3, with s1 < s2 < s3. Then, one needs to measure the sizes of the respective network traffic t1, t2 and t3. If the data exchange quota is sub-linear, the point (s3, t3) needs to lie below the line specified by (s1,t2) and (s2,t2) (see fig. 3).

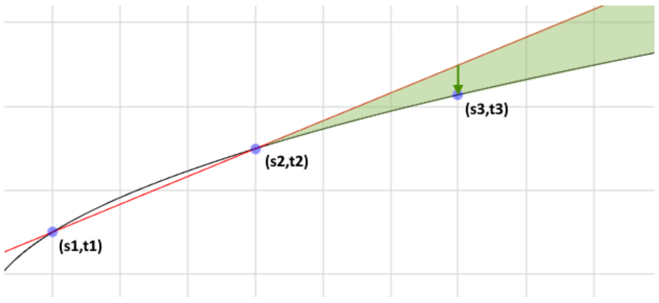

Fig. 3: *Automatic measuring of sub-linear exchange quota*

Stricter requirements could be imposed depending on the type of machine learning algorithm.

Acceptance Criteria

To pass the acceptance criteria of this binary KPI, the app should meeting one of the following requirements:

1. The data amount of the federated application being run on the execution platform needs to be independent of and less than the patient data size.
2. If the above is not the case, the app must employ additional mechanisms such as differential privacy, homomorphic encryption, or secure multi-party computation to exchange the model parameters.

Discussion

It needs to be stated that passing this KPI does not imply that no privacy leakage occurs. However, it helps to rule out the most problematic cases, even before manual verification takes place and can be assessed automatically and integrated into the platform. It is extremely difficult, if not impossible, to automatically determine privacy leakage by looking at the binary data, unless the semantics of the data is given. Since we are dealing with generic algorithms, we do not know the semantics of the data and therefore can only look at this from a semantic-agnostic, high-level perspective.

3.3.4 Encryption Requirements for Patient Data and Network Traffic (KPI 4)

Encryption is crucial to ensure that any stolen data stored on servers or logged network traffic does not reveal anything about patients, even in the case of a data breach.

In FeatureCloud, we apply encryption to

- Internet traffic containing model parameters, and
- Patient data stored in the medical centers.

Encryption of Communication Data

FeatureCloud forces the server and clients to leverage the HTTPS protocol to exchange the model parameters to make sure that communication between them is secure. The HTTPS protocol uses Transport Layer Security (TLS) (Dierks et al., 2014), the successor of Secure Sockets Layer (SSL)

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 826078.

(Freier et al., 2011) to secure the communication between two endpoints. By secure we mean that the protocol enforces user authentication (authenticating the server and a client to each other with the help of certificates), data integrity (data is not changed during communication) and data confidentiality (by encrypting data exchanged between the server and clients). This way, an outsider cannot hijack the communication (man-in-the-middle attack (Callegati et al., 2009), D2.1, section 7) or change the value of model parameters, and the value of model parameters is not exposed to the outsider. In this KPI, we require that all data traffic is encrypted in this manner.

This mainly affects the following connections (identified by the connected components, see also Fig. 1):

1. Controller and Socket Server (Internet traffic)
2. Controller and Consent API (Internet traffic)
3. Global API and Frontend (Internet traffic)
4. Controller and Frontend (Local network traffic)
5. App Container and App Frontend (Local network traffic)

Encryption of Patient Data

Although the patient data is never revealed to a third party, including the server, it is still a good security practice to encrypt patients’ data located in each medical center. There are two alternatives to this end: Asymmetric (public/private key) encryption or symmetric encryption. In the former, there are two different keys: private key and public key. The data is encrypted using a public key and can be decrypted by the corresponding private key. In the latter, there is only a single key, which is employed to encrypt or decrypt the data. RSA (Kaliski et al., 1998) is an example of asymmetric encryption algorithm while DES, 3DES, and AES are examples of symmetric encryption algorithms. Asymmetric encryption algorithms incur a remarkable computational overhead and are not recommended to encrypt a large amount of data, which is the case in medical centers with large patients’ datasets. On the other hand, symmetric algorithms are well-suited for encryption of huge datasets, especially AES, which is a very popular algorithm to this end.

Acceptance Criteria

For all network traffic, we require a minimum key size of 256 bits for the symmetric encryption by the cipher and a minimum key size of 2048 bits for public keys used during the asymmetric key exchange. We also require all patient data stored in FeatureCloud installations in medical centers to use AES with a minimum key size of 256 bits. Moreover, the authentication between the server and clients needs to be performed via certificates.

The minimum key sizes (for AES and all network traffics) and authentication by certificates are the acceptance thresholds for these binary KPIs.

Discussion

Whether a key size can be considered secure enough depends on the technology available. The above stated values are currently being used by applications requiring a high level of security (e.g., bank applications). However, with advancing technologies, this can change and require a revision.

Therefore, additionally to the fixed values stated above as acceptance criteria need to be revised regularly to always match a level considered to be secure (e.g. by following Lenstra's equation, Lenstra, 2006). Furthermore, several national recommendations exist, which are updated regularly (BSI, 2020) which are taken into account to update these values as well. Apart from key sizes, the

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 826078.

encryption technology itself (e.g. AES) is important, too. In case a weakness or an exploit is reported, the platform needs to be updated accordingly.

4 Conclusion

In this deliverable, we presented requirements for the initial architecture of the FeatureCloud platform and described the functionality of the platform’s components and its communication paradigm. We introduced four KPIs that are directly applicable to the platform and are independent of the algorithms or applications that are going to run inside the platform. These KPIs point at crucial security measures and privacy aspects and aim to increase awareness during system design and implementation. They ought to ensure that security and privacy deliberations are given precedence over the other considerations. Importantly though, they cannot guarantee a completely secure and privacy-preserving platform even when fulfilled but only reduce the risks on a general level.

5 References

[Bertino et al., 2018] Bertino, E., & Ferrari, E. (2018). Big data security and privacy. In A Comprehensive Guide Through the Italian Database Research Over the Last 25 Years (pp. 425-439). Springer, Cham.

[BSI, 2020] Federal Office for Information Security, Germany. Cryptographic Mechanisms: Recommendations and Key Lengths. https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TG02102/BSI-TR-02102-1.pdf

[Callegati et al., 2009] Callegati, F., Cerroni, W., & Ramilli, M. (2009). Man-in-the-Middle Attack to the HTTPS Protocol. IEEE Security & Privacy, 7(1), 78-81.

[Dierks et al., 2014] Dierks, T., & Rescorla, E. (2008). The transport layer security (TLS) protocol version 1.2.

[Freier et al., 2011] Freier, A., Karlton, P., & Kocher, P. (2011). The secure sockets layer (SSL) protocol version 3.0. IETF, 3, 1-67.

[Kaliski et al., 1998] Kaliski, B. (1998). PKCS# 1: RSA encryption version 1.5. RFC 2313, March.

[Lenstra, 2006] Lenstra, A. K. (2006). Handbook of Information Security, Volume 1: Key Concepts, Infrastructure, Standards and Protocols. John Wiley and Sons Ltd..

[Li et al., 2019] Li, T., Sahu, A. K., Talwalkar, A., & Smith, V. (2019). Federated learning: Challenges, methods, and future directions. arXiv preprint arXiv:1908.07873.

[Papadimitriou et al., 2003] Papadimitriou, C. H. (2003). Computational complexity (pp. 260-265). John Wiley and Sons Ltd..

[Yang et al., 2019] Yang, Q., Liu, Y., Chen, T., & Tong, Y. (2019). Federated machine learning: Concept and applications. ACM Transactions on Intelligent Systems and Technology (TIST), 10(2), 1-19.

[Wagner et al., 2018] Wagner, Isabel, and David Eckhoff. "Technical privacy metrics: a systematic survey." ACM Computing Surveys (CSUR) 51.3 (2018): 1-38.

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 826078.