fiskaly GmbH
Stutterheimstraße 16-18/2/20e
A - 1150 Wien

Umgebungsschutz der fiskaly SIGN Cloud-TSE

Sehr geehrte Damen und Herren,

bevor Sie in das angehängte Umgebungsschutzkonzept der fiskaly SIGN Cloud-TSE (technische Sicherheitseinrichtung) einsteigen, möchten wir hierzu ein paar Worte verlieren:

Um ein EAS (*elektronisches Aufzeichnungssystem*) in Deutschland vertreiben oder betreiben zu dürfen, muss eine TSE als integraler Bestandteil des EAS integriert sein. Eine entsprechende technische Integration sowie Dokumentation in der Verfahrensdokumentation[1] ist ihrem EAS-Produkt hinzuzufügen.

Im Zertifizierungsprozess wurde klargestellt, dass Steuerpflichtige und Kassenhersteller (*Integrator*) als *„Angreifer"* (*aus der Sicht der TSE*) gesehen werden. Deshalb muss eine TSE in sich selbst einen entsprechenden Schutz gegenüber möglicher Attacken der *„Angreifer"* mit sich bringen. Genau dieser Schutz wird mit der Umsetzung des Umgebungsschutzkonzepts gewährleistet.

In der Entwicklung und Zertifizierung einer TSE ist das Umgebungsschutzkonzept ein integraler Bestandteil. Der Umgebungsschutz muss eingehalten werden, andernfalls handelt es sich um keine zertifizierte TSE und folglich um Nicht-Erfüllung der gesetzlichen Vorgaben.
Wichtig: In der Umsetzung des Umgebungsschutzes, kann der TSE-Hersteller dies durch Auflagen an den Integrator und/oder Steuerpflichtigen oder durch Umsetzung im Produkt und im Rahmen des Betriebes erfüllen.

Als fiskaly haben wir uns für eine Umsetzung durch den TSE-Hersteller entschieden. In den nachfolgenden Seiten wird beschrieben, mit welchen technischen und organisatorischen Schritten die fiskaly SIGN Cloud-TSE, **ohne Zutun** des Steuerpflichtigen oder Integrators, die entsprechenden Schutzanforderungen erfüllt werden.

fiskaly stellt Ihnen gerne Abnahmebestätigungen über die Erfüllung des Umgebungsschutzes aus.

Alle nachfolgenden Themen sind rein durch fiskaly umzusetzen!

Sollten Sie Fragen haben, wenden Sie sich bitte an sales@fiskaly.com.

Beste Grüße,

Johannes Ferner (CEO), Patrick Gaubatz (CTO), Simon Tragatschnig (COO)

[1] DFKA-Muster-Verfahrensdokumentation zur ordnungsmäßigen Kassenführung

fiskaly GmbH
Stutterheimstraße 16-18/2/20e
A-1150 Vienna
Austria

Tel.: +43 1 9974459
Fax.: +43 1 9974459 99
E-Mail: office@fiskaly.com
Web: https://fiskaly.com

Handelsgericht Wien
HR-Nr.: FN 504484p
USt.-ID: ATU73948115
Steuer-Nr.: 03-634/8639

Erste Bank, BIC: GIBAATWWXXX
AT77 2011 1828 1714 6300
Geschäftsführung: Johannes Ferner,
Simon Tragatschnig, Patrick Gaubatz

Umgebungsschutzkonzept

fiskaly Security Module Application for Electronic Record-keeping Systems

TOE Version 1.0.5

Document Version 1.3.5

2021-04-21

fiskaly GmbH

Contents

1 Introduction **3**

1.1 Document and TOE Reference 3

2 Roles **4**

2.1 Tax Payer 4

2.2 ERS Manufacturer 5

2.3 SMAERS Administrator 5

2.4 Cloud Operator 5

2.5 SMAERS Manufacturer 5

3 Requirements **6**

4 Implementation Scenarios of the Requirements **8**

4.1 Cloud Realization 8

4.1.1 Amazon AWS 8

4.1.2 Google Cloud Platform 10

4.1.3 Microsoft Azure 11

4.1.4 General Considerations for Cloud Scenarios 12

4.2 Hardware Realization 14

4.2.1 On-Premises installation 14

4.2.2 Hardware Platform 15

4.2.3 Software 16

4.3 Special Case Scenarios 19

5 Update-Concept **20**

5.1 Introduction 20

5.2 Context of the rollout 20

5.2.1 Maintaining updated versions 21

Bibliography **22**

Chapter 1

Introduction

This document is the Umgebungsschutzkonzept for the fiskaly Security Module Application for Electronic Record-keeping Systems (fiskaly SMAERS). This document contains a list of valid scenarios how the operational environment of fiskaly SMAERS has to look like.

The document starts by defining the involved roles and stating the requirements, which are given by the Protection Profile [3], the Security Target [8], and the Security Architecture [7]. Then it lists possible scenarios, which allow to fulfill the stated requirements and thereby make it possible to install, execute, and use the TOE in a secure and legal fashion.

1.1 Document and TOE Reference

Document Type:	Umgebungsschutzkonzept
Document Version:	1.3.5
Document built from commit:	1481a077d7c6137ff154ab572b2957641a0c0eca
Date:	2021-04-21
Author:	fiskaly GmbH
Certification-ID:	BSI-DSZ-CC-1130
TOE Identification:	fiskaly Security Module Application for Electronic Record-keeping Systems
TOE Version:	1.0.5
CC Version:	3.1 Revision 5
Assurance Level:	EAL2 augmented with ALC_LCD.1 and ALC_CMS.3
PP Conformance:	BSI-CC-PP-0105-V2-2020 including the package *Trusted Channel between TOE and CSP* from BSI-CC-PP-0105-V2-2020

Chapter 2

Roles

The following roles are important to understand, judge, and realize the following scenarios for operational environments for the fiskaly SMAERS. Note that the role of the *integrator* is intentionally omitted. At the current moment it is unclear, if a manufacturer of an ERS might be an integrator or not. To prevent misunderstandings, we do not use the term integrator for now. Instead, this document defines and uses the following roles:

- *Tax Payer* (or persons being dependent on the *Tax Payer*)
- *ERS Manufacturer*
- *(SMAERS) Administrator*
- *SMAERS Manufacturer*
- *Cloud Operator*

Note that *Tax Payer* can be neither *(SMAERS) Administrator* nor *Cloud Operator*. The same is true for *ERS Manufacturer*, i.e. *ERS Manufacturer* can be neither *(SMAERS) Administrator* nor *Cloud Operator*. On the other hand, *Tax Payer* might also be *ERS Manufacturer*. In addition *(SMAERS) Administrator* might be *Cloud Operator*.

2.1 Tax Payer

The *Tax Payer* is the end user of the TOE and TSE. Although it is the *Tax Payer*'s duty to correctly use the TOE and protect it, the *Tax Payer* has to be considered as an attacker for certain kinds of attacks and can not be trusted completely.

2.2 ERS Manufacturer

The *ERS Manufacturer* builds the ERS system and develops the software of the ERS. He (or a third party) integrates the TOE / TSE into the ERS software. In case a third party does the integration, this party is also assumed to have the role *ERS Manufacturer*.

At the current moment it is unclear, how trustworthy the *ERS Manufacturer* is. Therefore this document does not delegate requirements of the TOE protection to the *ERS Manufacturer* and considers him to be a possible attacker, exactly as the *Tax Payer*. This might change in future version depending on the outcome of discussion between fiskaly GmbH, BSI, and BMF.

2.3 SMAERS Administrator

The *(SMAERS) Administrator* is considered to be trustworthy. Persons incorporating this role must not be *Tax Payer*, *ERS Manufacturer*, or persons, being dependent on *Tax Payer* or *ERS Manufacturer*.

For now, *(SMAERS) Administrator* are employees of fiskaly GmbH or commissioned by fiskaly GmbH to be *(SMAERS) Administrator*. In case of a commissioned *(SMAERS) Administrator* the commissioned person has to be independent of *Tax Payer* and *ERS Manufacturer*.

2.4 Cloud Operator

The *Cloud Operator* operates a cloud and allows third parties to use it for cloud computing services. Other roles of this document are able to use the cloud such that it executes (parts of) the ERS and the TOE / TSE.

The *Cloud Operator* is considered trustworthy to the extent that he has no intention to interfere with the systems he executes. Especially we assume, that a TOE being hosted by *(SMAERS) Administrator* in the cloud is not made accessible to *Tax Payer* or *ERS Manufacturer* to any extent exceeding the configuration by *(SMAERS) Administrator*.

2.5 SMAERS Manufacturer

The *SMAERS Manufacturer* (i.e. fiskaly GmbH) develops the SMAERS Software, i.e. the TOE and makes it available to *(SMAERS) Administrator*. It is provided in the form of a signed Docker image.

Chapter 3

Requirements

As detailed in the Security Architecture Document of the TOE [7], the TOE relies on its operational environment to protect it. The TOE has the following operational requirements:

- Separation of *(SMAERS) Administrator* and *Tax Payer* (ERS end user)
- Integrity and confidentiality protection of the *Secure Storage* (including protection against the *Tax Payer*)
- Integrity protection of the Docker image hosting the TOE (including protection against the *Tax Payer*)
- Protection of the running TOE instance (including protection against the *Tax Payer*)
- Protection of the Docker daemon, such that no potential attacker is able to control the daemon

Remember that the TOE needs a *Secure Storage* to store the PACE Password, the transaction count and additional internal configuration data. The *Secure Storage* is accessible to the TOE and to *(SMAERS) Administrator* only. It does not need to be protected against *Cloud Operator*, but *ERS Manufacturer* and *Tax Payer* must not have access to the content of *Secure Storage*.

The *Storage* contains signed transaction logs. These are exported data from the perspective of the Target of Evaluation with respect to the Common Criteria Certification. Still, this data is within the TSE and are not yet exported from the perspective of the TR-Certification. This data has to be protected, but the requirement of protection to these data is not as high as of the *Secure Storage*. The data are already integrity protected by the signatures, which the CSP created and by the continuity of transaction- and signature counter. Still, the availability of this data has to be ensured.

The Docker image, which contains the TOE software, is immutable and digitally signed. The Docker Daemon verifies the images's signature and starts it only, if the verification succeeds. To do so, it uses a preconfigured public key, to which only fiskaly GmbH has the private key.

These requirements come from the Protection Profile's Security Objectives for the Operational Environment, cf. section 4.2 of [3]. They are:

OE.SMAERSPlatform Secure platform storage

The platform that executes the TOE has to ensure the integrity of the TOE itself and to provide secure storage which protects the integrity and confidentiality of stored security relevant objects as required (cf. Chapter 1.1.2 'TOE Type'). The platform verifies and installs the UCP.

OE.SecOEnv Secure Operational Environment

The operational environment shall protect the integrity of the communication between the electronic record-keeping system and the TOE. The administrator shall act in a trustworthy way and is assumed to be the manufacturer or integrator. The administrator must be independent of the taxpayer.

OE.SecUCP Secure download and authorized use of Update Code Package

The platform role shall verify the authenticity of received update code packages and install only authentic update code packages.

For each of the implementation scenarios, this document contains a rationale, why the requirements are fulfilled by the scenario.

Chapter 4

Implementation Scenarios of the Requirements

The following scenarios allow to fulfill the requirements of the TOE to its operational environment.

4.1 Cloud Realization

If the ERS system, especially the "Aufzeichnungssystem" is not executed on the premises of the *Tax Payer* or *ERS Manufacturer*, but instead in the cloud, i.e. in a virtualized hosting environment, which is owned by *Cloud Operator* providing the hardware to rent, the *(SMAERS) Administrator* hosts the TOE in the same environment, which also hosts the ERS. Note that both the host of the TOE and the host of the ERS are operated within the same geographical data center region in order to meet requirement OE.SecOEnv.

The ERS is controlled by the *Tax Payer*, while *(SMAERS) Administrator* controls the TOE's host. Both communicate protected via HTTPS (TLS1.2+) and a suitable way to authenticate (i.e. server authentication and API tokens or client certificates). In addition, the configuration of the SMAERS and ERS ensure that both are executed in the same environment. This setup is shown in Figure 4.1.

This scenario assumes that the *Cloud Operator* has no intention to attack the TOE on its own or give the *Tax Payer* access to the TOE against the will of *(SMAERS) Administrator*. The *Cloud Operator* is seen as trustworthy in that respect.

4.1.1 Amazon AWS

In order to install SMAERS in this scenario, the AWS cloud is used. More precisely, a new EC2 instance (that is only to be used for the purpose described in this scenario) is created, which is launched in the AWS-Region Frankfurt. Note that it is required, that the Aufzeichnungssystem

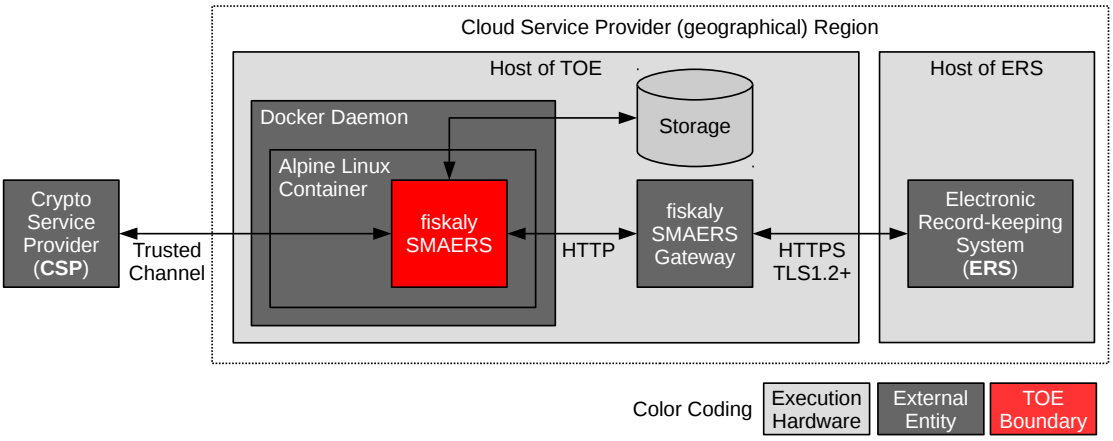

Figure 4.1: Overview of the CTSS components in a Cloud Context

component of the ERS is launched in the same facility, i.e. the Frankfurt region of AWS.

Setting up the EC2 instance

(SMAERS) Administrator is responsible to set up the EC2 instances. When doing so, *(SMAERS) Administrator* has to proceed as follows:

(1) The instance is created using the image Ubuntu Server 20.04 LTS, 64 bit.

(2) Storage is attached and configured to ensure that the storage is encrypted (by using the standard EBS encryption of AWS). This storage is configured to grow on demand basis accordingly. Also the root drive (which is attached by default) shall be encrypted

(3) The next steps are performed under consideration of the guide *Best practices for Amazon EC2*[1] from Amazon:

  (1) A dedicated security group is created and the security group is associated with the VM.

  (2) By connecting to the VM via the web UI, the following steps are performed

    (1) A new user admin is created

    (2) The new admin user is assigned a secure password and sudo privileges

    (3) The standard user is removed

    (4) configure EC2 Instance Connect[2] to connect to the system per SSH

    (5) update and upgrade all packages

  (3) Follow the steps of section 4.1.4 to harden the linux image

(4) Docker is installed on the machine

[1] cf. `https://docs.amazonaws.cn/en_us/AWSEC2/latest/UserGuide/ec2-best-practices.html`

[2] cf. `https://aws.amazon.com/de/blogs/compute/new-using-amazon-ec2-instance-connect-for-ssh-access-to-your-ec2-instanc`

(5) The Docker daemon is configured such that it only runs images that have been signed by fiskaly GmbH using *Docker Content Trust*[3].

(6) For each SMAERS instance to be run in this EC2 instance follow section 3.1.1 of AGD [6] to install SMAERS and make it available on port x.

(7) Install NGINX as a reverse proxy for TLS traffic. Obtain a certificate from `https://letsencrypt.org/` according to the instructions mentioned in `https://www.nginx.com/blog/using-free-ssltls-certificates-from-lets-encrypt-with-nginx/` Also, ensure that only TLS 1.2 and TLS 1.3 with cipher suites that are recommended by BSI TR-2102[4] are accepted. All unencrypted traffic is denied.

(8) The dedicated security group is configured to allow only incoming requests originating from the same AWS region to the machine. Also, the dedicated security group is configured to prevent access to any other ports, except those that are exposed (i.e. HTTPS) by SMAERS and used for SSH.

(9) In the complete setup, it shall be ensured that - beside the components described before - no component is existing between the Tax Payer and SMAERS that actively processes the data sent by the Tax Payer. With other words: only components that transparently forward the traffic are allowed.

After this setup of the environment of the SMAERS instance is completed, the steps of the setup procedure form the Guidance Document [5] are required to make the SMAERS software operational.

4.1.2 Google Cloud Platform

This section discusses the installation of SMAERS instances in the Google Cloud Platform (GCP). In this scenario, a new instance of the Google Compute Engine (that is only to be used for the purpose described in this scenario) is created, which is located in the Frankfurt region (europe-west3). Note that the Aufzeichnungssystem component of the ERS must be located in the same operational environment, i.e. the Frankfurt region of GCP.

Setting up the Compute Engine instance

(SMAERS) Administrator is responsible for the setup of the Compute Engine instance as follows:

(1) Setup the instance with Ubuntu Server 20.04 LTS

(2) The instance is configured to use Standard Storage of the Cloud Storage provided by same region. The storage is encrypted by default and grows as needed.

(3) Enable *OS Login* as recommended by GCP.[5] Particularly, set up *OS Login* with 2-step verification.[6]

[3] cf. `https://docs.docker.com/engine/security/trust/`
[4] `https://www.bsi.bund.de/DE/Publikationen/TechnischeRichtlinien/tr02102/tr02102_node.html`
[5] `https://cloud.google.com/compute/docs/instances/connecting-advanced`
[6] `https://cloud.google.com/compute/docs/oslogin/setup-two-factor-authentication`

(4) (Optional Step) Setup SSH access.[7] If SSH is enabled, *(SMAERS) Administrator* must perform hardening in accordance with Section 4.1.4.

(5) Docker is installed on the machine

(6) The Docker daemon is configured such that it only runs images that have been signed by fiskaly GmbH using *Docker Content Trust*[8].

(7) For each SMAERS instance to be run in this EC2 instance follow section 3.1.1 of AGD [6] to install SMAERS and make it available on port x.

(8) Install NGINX as a reverse proxy for TLS traffic. Obtain a certificate from `https://letsencrypt.org/` according to the instructions mentioned in `https://www.nginx.com/blog/using-free-ssltls-certificates-from-lets-encrypt-with-nginx/` Also, ensure that only TLS 1.2 and TLS 1.3 with cipher suites that are recommended by BSI TR-2102[9] are accepted. All unencrypted traffic is denied.

(9) A VPC (Virtual Private Cloud) firewall is configured to allow only incoming requests originating from the same GCP region to the machine. Also, the VPC firewall is configured to prevent access to any other ports, except those that are exposed (i.e. HTTPS) by SMAERS and used for SSH.

(10) In the complete setup, it shall be ensured that - beside the components described before - no component is existing between the Tax Payer and SMAERS that actively processes the data sent by the Tax Payer. With other words: only components that transparently forward the traffic are allowed.

After this setup of the environment of the SMAERS instance is completed, the steps of the setup procedure form the Guidance Document [5] are required to make the SMAERS software operational.

4.1.3 Microsoft Azure

This section discusses the installation of SMAERS instances in Microsoft Azure. In this scenario, a new Azure Virtual Machine (that is only to be used for the purpose described in this scenario) is created, which is located in the Frankfurt region (Germany West Central). Note that the Aufzeichnungssystem component of the ERS must be located in the same operational environment, i.e. the Frankfurt region of Azure.

Setting up the Azure Virtual Machine

(SMAERS) Administrator is responsible for the setup of the Azure Virtual Machine (VM) as follows:

(1) Create the VM using Ubuntu Server 20.04 LTS image.[10]

[7] `https://cloud.google.com/compute/docs/instances/managing-instance-access#add_oslogin_keys`
[8] cf. `https://docs.docker.com/engine/security/trust/`
[9] `https://www.bsi.bund.de/DE/Publikationen/TechnischeRichtlinien/tr02102/tr02102_node.html`
[10] `https://docs.microsoft.com/en-us/azure/virtual-machines/linux/`

(2) The VM is configured to use Azure managed disks.[11] Data in Azure managed disks is encrypted using 256-bit AES encryption.[12] The storage encryption must be set up accordingly.[13]

(3) **The following feature must only be used if the feature is no longer in the public preview state (and meant for testing purposes only):** Configure the instance to use Azure Active Directory authentication.[14]

(4) (Optional once Azure Active Directory can be used in production:) Use the SSH option in the setup of the Azure VM.[15] If SSH is enabled, *(SMAERS) Administrator* must perform hardening in accordance with Section 4.1.4.

(5) Docker is installed on the machine

(6) The Docker daemon is configured such that it only runs images that have been signed by fiskaly GmbH using *Docker Content Trust*[16].

(7) For each SMAERS instance to be run in this EC2 instance follow section 3.1.1 of AGD [6] to install SMAERS and make it available on port x.

(8) Install NGINX as a reverse proxy for TLS traffic. Obtain a certificate from `https://letsencrypt.org/` according to the instructions mentioned in `https://www.nginx.com/blog/using-free-ssltls-certificates-from-lets-encrypt-with-nginx/` Also, ensure that only TLS 1.2 and TLS 1.3 with cipher suites that are recommended by BSI TR-2102[17] are accepted. All unencrypted traffic is denied.

(9) A network security group is configured to allow only incoming requests originating from the same Azure region to the machine. Also, the network security group is configured to prevent access to any other ports, except those that are exposed (i.e. HTTPS) by SMAERS and used for SSH.

(10) In the complete setup, it shall be ensured that - beside the components described before - no component is existing between the Tax Payer and SMAERS that actively processes the data sent by the Tax Payer. With other words: only components that transparently forward the traffic are allowed.

After this setup of the environment of the SMAERS instance is completed, the steps of the setup procedure form the Guidance Document [5] are required to make the SMAERS software operational.

4.1.4 General Considerations for Cloud Scenarios

Hardening

(SMAERS) Administrator is responsible for hardening of the instance. The following steps must be carried out by the *(SMAERS) Administrator* in order to harden the linux system to a

[11] `https://docs.microsoft.com/en-us/azure/virtual-machines/managed-disks-overview`
[12] `https://docs.microsoft.com/en-us/azure/virtual-machines/disk-encryption`
[13] `https://docs.microsoft.com/en-us/azure/virtual-machines/linux/disk-encryption-portal-quickstart`
[14] `https://docs.microsoft.com/en-us/azure/virtual-machines/linux/login-using-aad`
[15] `https://docs.microsoft.com/en-us/azure/virtual-machines/linux/mac-create-ssh-keys`
[16] cf. `https://docs.docker.com/engine/security/trust/`
[17] `https://www.bsi.bund.de/DE/Publikationen/TechnischeRichtlinien/tr02102/tr02102_node.html`

reasonable extent:

- change the SSH default port
- disable SSH login for the root account
- configure SSH to be key-based only, i.e. prevent SSH logins based on passwords.
- configure SSH to disable protocol version 1
- configure SSH to only accept strong ciphers suites (as recommended by BSI TR-2102)
- use the UFW firewall to drop/deny all incoming connections except SSH and the ports, which are bound to SMAERS.

Maintenance

(SMAERS) Administrator is responsible for the maintenance of the instance. The *(SMAERS) Administrator* must proceed as follows:

- On a weekly basis install security updates (after testing them in a staging environment)
- On a monthly basis consider to install functional updates (after testing them in a staging environment)
- On a monthly basis check log files (or set up an automated system to check them)
- Watch CVEs of Ubuntu and of the packages in use and subscribe to the corresponding mailing lists

Backup Mechanism

Due to the crucial necessity of the *Tax Payer* to keep a copy of the transaction data, it is required to have a backup mechanism to store these data outside of the described cloud scenario. It is the duty of the *Tax Payer* to export and store the transaction logs frequently in a safe memory. This scenario does not include a backup structure.

Rationale / Mapping of Requirements

The cloud scenario sets up the instance in such a way, *Tax Payer* has access to the TOE interface through a TLS channel and can use the TOE's functionality this way. Here TLS protects the communication between ERS and TOE. By this and the separation of roles, OE.SecOEnv is met.

The integrity of the TOE is ensured by *Docker Content Trust* and the usage of signed docker images. Together with the reduction of the access to the TOE, docker installation and data in *Secure Storage* and *Storage* to *(SMAERS) Administrator* (and *Cloud Operator*), the requirements of OE.SMAERSPlatform are realized.

To implement the requirements of OE.SecUCP, again signed docker images and *Docker Content Trust* are used. This way, only images which were signed by fiskaly GmbH can be executed. In addition, the TOE itself detects up- and downgrades reliably, cf. [7].

4.2 Hardware Realization

This section describes, how the requirements for the environment will be fulfilled if the SMAERS is installed on a system within the premises of the *Tax Payer*. This situation is special because also in this scenario, the *Tax Payer* must be independent of the *(SMAERS) Administrator*. Specific measure have to be taken in order to ensure that no authorized access is possible.

This scenario bases on the following principles:

(1) for the primary installation, a SMAERS is installed on a dedicated machine. The installation is performed by *(SMAERS) Administrator* and the whole machine is shipped to the *Tax Payer*

(2) Administration is still performed by *(SMAERS) Administrator* and the *Tax Payer* will not have administrative access

The following two sections introduce the specifics of the hardware and the installation

4.2.1 On-Premises installation

For the installation in this scenario a Standard PC Hardware is used. The hardware characteristics depend on the work load to be expected. The installation is prepared by *(SMAERS) Administrator* and shipped to the site of the *Tax Payer* or the *ERS Manufacturer*. Note that it can also be shipped and then installed, as long as it is ensured, that the installation is under control of *(SMAERS) Administrator*. This can be realized by having trusted personal on the site of *Tax Payer* and combining local and remote setup steps. The following aspects will be fulfilled in any case

(1) The machine shall support UEFI Secure Boot. By installing a hardware protected platform key this allows for configuring the platform in such a way that only code (i.e. Kernel + initial Ramdisk) that is authorized by the *(SMAERS) Administrator* can be booted.

(2) The machine uses a TPM2 for storing a keys used for full disk encryption. Hereby, keys are only unsealed if the hardware platform's firmware, configuration as well as the Kernel and initial Ramdisk is unmodified.

(3) The used storage of the machine shall have a MTBF of at least x In cases, where the MTBF cannot be guaranteed by the specification of the device, a software or hardware RAID with redundancy is used to utilize the availability of the stored data

4.2.2 Hardware Platform

(SMAERS) Administrator is responsible for the setup of the hardware platform as follows:

(1) TPM2 shall be enabled.

(2) Secure Boot shall be enabled.

(3) All Secure Boot Keys shall be cleared.

(4) Secure Boot shall be put into “Setup Mode”.

(5) The UEFI Platform Key provided by *(SMAERS) Administrator* shall be installed on the platform.

Once this is done, the hardware platform will refuse booting any code that has not been signed by *(SMAERS) Administrator*. Note that the initial installation of the base Operating System may require using factory keys to allow booting the Operating System installation environment, if the latter has not been signed by *(SMAERS) Administrator*.

As an example, the following script can be used to generate a UEFI Platform Key and install it on the platform:

```
openssl req \
  -nodes \
  -new \
  -x509 \
  -newkey rsa:2048 \
  -subj "/CN=UEFI Platform Key/" \
  -keyout signing.key \
  -outform PEM \
  -out cert.pem \
  -days 3650 \
  -sha256

UUID=‘uuidgen -r‘

cert-to-efi-sig-list \
  -g ${UUID} \
  cert.pem \
  cert.esl

for KEY in PK KEK db; do
  sign-efi-sig-list \
    -k signing.key \
    -c cert.pem \
    ${KEY} \
    cert.esl \
    ${KEY}.auth

  # install key on platform:
  efi-updatevar \
    -f ${KEY}.auth \
    ${KEY}
done
```

4.2.3 Software

(SMAERS) Administrator is responsible for the setup of the software as follows

(1) Ubuntu 20.04 (20.04.02.0 to be more precise) LTS image will be used as the base Operating System.

(2) Follow the steps of section 4.1.4 to harden the base Operating System.

(3) During setup, full disk encryption is used. The passphrase used acts as a recovery key and therefore needs to be memorized by the *(SMAERS) Administrator*.

(4) The UFW firewall is enabled.

(5) Docker is installed on the machine.

(6) The Docker daemon is configured such that it only runs images that have been signed by fiskaly GmbH using *Docker Content Trust*[18].

(7) For each SMAERS instance to be run in this EC2 instance follow section 3.1.1 of AGD [6] to install SMAERS and make it available on port x.

(8) Install NGINX as a reverse proxy for TLS traffic. Obtain a certificate from `https://letsencrypt.org/` according to the instructions mentioned in `https://www.nginx.com/blog/using-free-ssltls-certificates-from-lets-encrypt-with-nginx/` Also, ensure that only TLS 1.2 and TLS 1.3 with cipher suites that are recommended by BSI TR-2102[19] are accepted. All unencrypted traffic is denied.

(9) UFW is configured to allow only incoming requests originating from the same local network. Also, UFW is configured to prevent access to any other ports, except those that are exposed (i.e. HTTPS) by SMAERS and used for SSH. Please note that this configuration assumes that the SSH traffic that is required for administration of SMAERS will be routed to the machine via a local component in the network (i.e. a router providing NAT). If this should not be the case, traffic for the SSH port might need to be allowed from everywhere.

(10) In the complete setup, it shall be ensured that - beside the components described before - no component is existing between the Tax Payer and SMAERS that actively processes the data sent by the Tax Payer. With other words: only components that transparently forward the traffic are allowed.

After the initial setup – as described above – a EFI Unified Kernel Image[20] has to be generated, signed and installed. In addition, a key for Full Disk Encryption has to be stored in the TPM2 chip [21]. To achieve this, the following steps shall be executed:

The following commands can be used for generating and signing a a EFI Unified Kernel Image:

```
# generate a EFI Unified Kernel Image:
objcopy \
  --add-section .osrel=/etc/os-release \
  --add-section .cmdline=/proc/cmdline \
```

[18] `https://docs.docker.com/engine/security/trust/`
[19] `https://www.bsi.bund.de/DE/Publikationen/TechnischeRichtlinien/tr02102/tr02102_node.html`
[20] `https://systemd.io/BOOT_LOADER_SPECIFICATION#type-2-efi-unified-kernel-images`
[21] `https://tpm2-software.github.io/2020/04/13/Disk-Encryption.html`

```
  --add-section .linux=/boot/vmlinuz \
  --add-section .initrd=/boot/initrd.img \
  --change-section-vma .osrel=0x20000 \
  --change-section-vma .cmdline=0x30000 \
  --change-section-vma .linux=0x40000 \
  --change-section-vma .initrd=0x3000000 \
  /usr/lib/systemd/boot/efi/linuxx64.efi.stub unsgined.efi

# sign image with the UEFI Platorm Key:
sbsign \
  --key  /uefi/signing.key \
  --cert /uefi/cert.pem \
  --output /boot/efi/EFI/BOOT/BOOTX64.EFI \
  unsigned.efi

# remove unsigned image:
rm unsgined.efi
```

After that, the system needs to be rebooted. This will load the previously generated and signed EFI Unified Kernel Image: `/boot/efi/EFI/BOOT/BOOTX64.EFI`. During booting the EFI Unified Kernel Image the platform will make sure that each component of the boot process is "measured" (i.e., hashed) in a register in the TPM2 built into the system. A TPM2 has several of such registers which are used for different purposes – for instance, PCR0 contains measurements of system firmware components and PCR4 contains information about the partition table and the bootloader.

Rebooting the system makes sure that the state of the TPM2 PCRs are in a known, expected state. This is important, because the next steps concern storing a storage encryption key that is sealed to the state of selected PCRs. Hence, the encryption keys can only be loaded from the TPM2 if the system is in a known state. Any tampering with the system's configuration will prevent the TPM2 from unsealing the encryption keys.

The following steps require the `tpm2-tools`[22] to be installed on the system. First, a PCR policy (i.e. `pcr.policy`) needs to be created. This policy will be used to seal the encryption keys to the state of selected PCRs: PCR0, PCR2, PCR4, PCR7 and PCR14. The following commands create the required `pcr.policy` file:

```
export TPM2TOOLS_TCTI=device:/dev/tpmrm0

# create a policy that measures PCR0, PCR2, PCR4, PCR7 and PCR14:
tpm2_startauthsession -S session.ctx
tpm2_policypcr -Q -S session.ctx -l sha256:0,2,4,7,14 -L pcr.policy
tpm2_flushcontext session.ctx

# cleanup:
rm *.ctx
```

Next, a random secret key is generated and added as a second LUKS key (the first ony is the recovery key that is known by the *(SMAERS) Administrator*) used for decrypting the platform's storage (assuming that `/dev/nvme0n1p3` is the encrypted block storage device):

```
# generate a random secret key:
dd if=/dev/random bs=1 count=32 status=none > secret.key

# adding secret key as second LUKS key:
cryptsetup luksAddKey /dev/nvme0n1p3 secret.key
```

[22] `https://github.com/tpm2-software/tpm2-tools`

Now, the `secret.key` is loaded into the TPM2 and bound to the `pcr.policy` using the following commands:

```
export TPM2TOOLS_TCTI=device:/dev/tpmrm0

# load the secret.key into TPM2:
tpm2_createprimary -Q -C o -c prim.ctx
cat secret.key | tpm2_create -Q -g sha256 -u seal.pub -r seal.priv -i- -C prim.ctx
     -L pcr.policy

# destroy secret.key as it is no longer needed:
shred -u secret.key

# make the key persistent:
tpm2_evictcontrol -C o -c 0x81010001
tpm2_load -Q -C prim.ctx -u seal.pub -r seal.priv -n seal.name -c seal.ctx
tpm2_evictcontrol -c seal.ctx 0x81010001 -C o

# cleanup:
rm *.ctx *.policy *.name *.priv *.pub
```

After that, the TPM2 will have stored the secret key and can be loaded automatically when the system is booted using the following commands:

```
export TPM2TOOLS_TCTI=device:/dev/tpmrm0

# start a new session:
tpm2_startauthsession --policy-session -S session.ctx
tpm2_policypcr -Q -S session.ctx -l sha256:0,2,4,7,14

# unseal key and pipe into cryptsetup:
tpm2_unseal -p session:session.ctx -c 0x81010001 | cryptsetup luksOpen --key-file
    =- /dev/nvme0n1p3
tpm2_flushcontext session.ctx

# cleanup:
rm *.ctx

# extend PCR14 with random data to prevent further unsealing
tpm2_pcrextend 14:sha256="$(tpm2_getrandom 32 | xxd -p -c 36)"
```

The script will only unseal the secret key if the listed PCRs have exactly the same values as they had when the key has been sealed. Now it should be clear that a new encryption key has to be generated and sealed whenever any of the relevant PCRs change. Examples for such situations include: system firmware updates, system configuration changes, kernel updates, etc.

To finish the installation, the *(SMAERS) Administrator* will get in contact with the *Tax Payer* to agree on a method for remote administration. In the standard case this will mean that the *Tax Payer* will open a port to the network where the SMAERS will be deployed for SSH access. After this setup of the environment of the SMAERS instance is completed, the steps of the setup procedure form the Guidance Document [5] are required to make the SMAERS software operational.

Rationale / Mapping of Requirements

The on-premise scenario sets up the instance in such a way, *Tax Payer* has access to the TOE interface through a TLS channel and can use the TOE's functionality this way. Here TLS protects

the communication between ERS and TOE. By this and the separation of roles, OE.SecOEnv is met.

The integrity of the TOE is ensured by *Docker Content Trust* and the usage of signed docker images. Together with the reduction of the access to the TOE, and the encrypted disks, docker installation and data in *Secure Storage* and *Storage* to *(SMAERS) Administrator* (and *Cloud Operator*), the requirements of OE.SMAERSPlatform are realized.

To implement the requirements of OE.SecUCP, again signed docker images and *Docker Content Trust* are used. This way, only images which were signed by fiskaly GmbH can be executed. In addition, the TOE itself detects up- and downgrades reliably, cf. [7].

4.3 Special Case Scenarios

If none of the previous scenarios fit a *Tax Payer*'s infrastructure, fiskaly GmbH will analyze the needs and possible solutions and discuss them with BSI to find realization to host the TOE securely. If BSI agrees to the proposed solution, such realizations are possible and covered by the scope of this certification.

Chapter 5

Update-Concept

5.1 Introduction

The Protection Profile for SMAERS ([3]) requires that the developer provides the BSI with a description of *The update procedures to allow for recovery from security incidents including the procedures for creating, distributing, and enforcing installation of update code packages for the TOE and the CSP,*. This description is contained in this section.

5.2 Context of the rollout

The TOE (i.e. SMAERS) and the corresponding CSP-L are both managed by fiskaly GmbH or a trustworthy third party on behalf of fiskaly GmbH. This fact also forms the basis of this update concept. It has the following advantages:

- fiskaly GmbH is aware of every installation of a CSP-L or SMAERS and maintains a list of all installed instances
- fiskaly GmbH has constant access to all installations for management purposes (which includes the process of updates)

SMAERS and the CSP-L can be updated separately. While the CSP-L provides a dedicated function for update (which has also been part of the Common Criteria evaluation), in the case of SMAERS, the update is mostly performed via its environment. More specifically: SMAERS is operated within a docker container and for an update the complete docker container is replaced. Please note that replacing the docker container will not destroy the database of SMAERS. After an update, the updated SMAERS will use the same database and perform any required update of the database schema (if required). This also means that the ID of the SMAERS will persist over updates.

5.2.1 Maintaining updated versions

Any updates for SMAERS or the CSP-L are developed in accordance with the fiskaly GmbH development lifecycle ([4]). As soon as a new version of SMAERS or the CSP-L has been released (and certified), fiskaly GmbH follows the following procedure for updates:

(1) SMAERS and the CSP-L can be updated separately. New releases shall remain backwards compatible for 6-12 months, if compatible with future certification requirements. Each version of SMAERS identifies the major versions of the CSP-L that it can be operated with

(2) If an update should contain critical (i.e. security relevant) aspects, it will be rolled out as soon as possible. fiskaly GmbH is committed to update all operational instances within 1 month

(3) In all other cases, the updates will be performed within planned maintenance windows. Customers can view the planned windows under status.fiskaly.com. fiskaly GmbH is committed to update all operational instances within 6 months

(4) It is very well possible that updates of an instance lead to problems within certain cases. This can for example occur if a SMAERS instance that is operated on premise is not accessible. If such problems should occur, fiskaly GmbH will allow an additional month for security relevant updates and an additional 6 month for all other updates.

(5) If after 2 months (for security relevant updates) or 12 months (for all other updates) an instance could not be updated, fiskaly GmbH will render the instance inoperative. Depending on the specific situation, this can be achieved in many different ways. Typical procedures include that *Tax Payer* will call the function of SMAERS to render the instance inoperative (DisableSecureElement) or fiskaly GmbH or *Tax Payer* terminates the contract and fiskaly GmbH blocks access to the CSP-L for the SMAERS instance.

Bibliography

[1] Common Criteria. Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and General Model, CCMB-2017-04-001, Version 3.1, Revision 5, April 2017. https://www.commoncriteriaportal.org/files/ccfiles/CCPART1V3.1R5.pdf.

[2] Federal Office for Information Security. Protection Profile Cryptographic Service Provider Light (CSPL) Version 1.0, BSI-CC-PP-0111-2019. https://www.bsi.bund.de/SharedDocs/Zertifikate_CC/PP/aktuell/PP_0111.html.

[3] Federal Office for Information Security. Security Module Application for Electronic Record-keeping Systems (SMAERS) Version 1.0, BSI-CC-PP-0105-V2-2020. https://www.bsi.bund.de/SharedDocs/Zertifikate_CC/PP/aktuell/PP_0105_0105_V2.html.

[4] fiskaly GmbH. Life-Cycle Support Documentation fiskaly Cloud Crypto Service Provider (fiskaly CSPL) and fiskaly Security Module Application for Electronic Record-keeping Systems (fiskaly SMAERS) 1.2.8, 2021.

[5] fiskaly GmbH. Preparative Procedures & Operational User Guidance Documentation – fiskaly Security Module Application for Electronic Record-Keeping Systems, Version 1.1.4, 2021.

[6] fiskaly GmbH. Preparative Procedures & Operational User Guidance Documentation fiskaly Cloud Crypto Service Provider Version 1.3.0, 2021.

[7] fiskaly GmbH. Security Architecture – Security Module Application for Electronic Record-Keeping Systems, Version 1.1.4, 2021.

[8] fiskaly GmbH. Security Target, – fiskaly Security Module Application for Electronic Record-Keeping Systems, Version 1.1.4, 2021.

Keywords

Term	Description
authentication reference data	data used by the TOE to verify the authentication attempt of a user
authentication verification data	data used by the user to authenticate themselves to the TOE
authenticity	the property that ensures that the identity of a subject or resource is the one claimed (cf. ISO/IEC 7498-2:1989)
cluster	a system of TOE samples initialized by an administrator and communication through trusted channels in order to manage known users and to share the cryptographic keys
cryptographic key	a variable parameter which is used in a cryptographic algorithm or protocol
data integrity	the property that data has not been altered or destroyed in an unauthorized manner (cf. ISO/IEC 7498-2:1989)
firmware	executable code that is stored in hardware and cannot be dynamically written or modified during execution while operating on a non-modifiable or limited execution platform, cf. ISO/IEC 19790
hardware	physical equipment or comprises the physical components used to process programs and data or to protect physically the processing components, cf. ISO/IEC 19790
Issuer of update code package	Trusted authority issuing an update code package (UCP) and holding the signature private key for signing the UCP and corresponding to the public key implemented in the TOE for verification of the UCP. The issuer is typically the TOE manufacturer. The issuer of an UCP is identified by the security attribute Issuer of the UCP.
Platform guidance	All documentation provided by the hardware manufacturer, or software platform manufacturer, that provides information on how to securely implement functionality.

private key	confidential key used for asymmetric cryptographic mechanisms like decryption of cipher text, signature-creation or authentication proof, where it is difficult for the adversary to derive the confidential private key from the known public key
public key	public known used for asymmetric cryptographic mechanisms like encryption of cipher text, signature-verification or authentication verification, where it is difficult for the adversary to derive the confidential private key from the known public key
secret key	key of symmetric cryptographic mechanisms, using two identical keys with the same secret value or two different values, where one may be easy calculated from the other one, for complementary operations like encryption / decryption, signature-creation / signature-verification, or authentication proof / authentication verification
secure channel	a trusted channel which is physically protected and logical separated communication channel between the TOE and the user, or is protected by means of cryptographic mechanisms
software	executable code that is stored on erasable media which can be dynamically written and modified during execution while operating on a modifiable execution platform, cf. ISO/IEC 19790
trusted channel	a means by which a TSF and another trusted IT product can communicate with necessary confidence (cf. CC part 1 [1], paragraph 97)
update code package	code if implemented changing the TOE implementation at the end of the TOE life time

Table 5.1: Glossary (Table 8 in Base-PP [2])

Abbreviations

Acronym	**Term**
A.xxx	Assumption
CC	Common Criteria
CSP	Cryptographic Service Provider
CSPLight	Cryptographic Service Provider Light
ECC	Elliptic curve cryptography
HMAC	Keyed-Hash Message Authentication Code
KDF	Key derivation function
MAC	Message Authentication Code
n. a.	Not applicable
O.xxx	Security objective for the TOE
OE.xxx	Security objective for the TOE environment
OSP.xxx	Organisational security policy
PACE	Password Authenticated Connection Establishment
PKI	Public key infrastructure
PP	Protection profile
SAR	Security assurance requirements
SFR	Security functional requirement
T.xxx	Threat
TOE	Target of Evaluation
TSF	TOE security functionality
UCP	update code package

Table 5.2: Abbreviations (Table 9 in Base-PP [2])