SS7 Attacker Heaven turns into Riot: How to make Nation-State and Intelligence Attackers' lives much harder on mobile networks

SigFW
Open Source SS7/Diameter firewall
for Antisniff, Antispoof & Threat Hunt

Martin Káčer
Philippe Langlois
P1 Security

BLACK HAT USA 2017

Content

1 Abstract

The SS7 mobile vulnerabilities affect the security of all mobile users worldwide. The SS7 is signalisation between Mobile Operators Core Network about where your mobile phone is located and where to send media, so the secured end-device does not help here, as it is only a consequence of having legitimate SS7 traffic. To protect against SS7 vulnerabilities, you need to play at operator-level. And this was not really the kind of thing you could do up till now.

Let's change this. In this talk we propose methods that allow any operator in the world - not only the rich ones - to protect themselves and send the attackers' tricks back to the sender. What if SS7 became a much more difficult and problematic playground for the attacker?

In this talk, we will discuss the current status, possible solutions, and outline advanced SS7 attacks and defenses using open-source SS7 firewall which we will publish after the talk. The signaling firewall is new, so we will not only use it to reduce the vulnerabilities in the SS7 networks, but we also show how to trick and abuse the attackers to make the work much harder for attackers, and give them a hard time interpreting the results. Intelligence agencies love SS7 for the wrong reasons. We will show examples and how we can make eavesdropping and geolocation a nightmare for these nation-state attackers.

The adoption of such signaling firewall could help to reduce the exposure for both active and passive attacks on a larger scale. We will present the capabilities of this solution including the encryption of signaling, report the attacks to central threat intelligence and forward the attackers to honeypot. So what about to find where these SS7 attacks are coming and to start protecting the networks?

2 Introduction

2.1 Problem Statement

The international SS7 network has been standardized and built in past as trusted network with only trusted partners. The network itself and by design does not authenticate and authorize the peers in the network and also does not encrypt the signalling communication. The exposure of these networks comes from the design and the architecture requirement of roaming architecture in past architecture releases.

Additionally we should not expect that the SS7 network will be phased out soon. The voice could be replaced by VoLTE (4G) with IMS home routed architecture, but such deployment requires VoLTE capable devices and VoLTE networks with the similar radio coverage compared to 2G, 3G. So before some operator decide to shut-down both 2G and 3G network, all the home subscribers should be VoLTE enabled. And the operator should consider also inbound-roamers.

In the LTE the Diameter protocol has replaced the SS7 signalling. However the similar issues are still present. Lack of authentication and no encryption of the signalling communication.

2.2 Related work

Several companies are offering commercial signalling firewalls and also there has been significant work on GSMA level. However we still think the problem is not fully covered. These commercial firewall solutions are reducing the risk up to some level mainly with focus on HPLMN protection, but are not so widely adopted and still there are several ways how the protection could be bypassed. These technical corner cases comes mainly from possibility of spoofing of the SCCP and Diameter messages and lack of protection of subscribers while being in roaming. Here we provide novel approach to fixing this thanks to open source approach and new signing and encryption approach.

3 The Approach

In this work we will outline some advanced SS7 attacks, including spoofing of messages, targeting roaming subscribers, some possible attacks done by MITM and passive attacks which are not addressed much by the industry today.

We will describe the open source SS7 and Diameter firewall (SigFW) using open source SS7 and Diameter stack which could be used to help to address the signalling vulnerabilities and the advanced attacks.

The open-source SigFW should be considered as **reference implementation** and **research project** but **without any warranty** and it is not a carrier grade solution.

3.1 SS7 firewall - Technical capabilities

- Open SS7 TCAP encryption and signing of the SS7 messages, including auto encryption setup
- SS7 SCCP blacklists (Category 0)
- SS7 TCAP blacklists (Category 1)
- SS7 MAP firewall rules (Category 2)
- Signalling IDS integration (for Category 3 and advanced detection)
- SS7 Filtering and honeypoting
- Centralized threat reporting with mThreat integration
- Collaboration with other SS7 and signaling security systems
- Management through open APIs
- Passive run (re-run traffic from pcap or passive interface to test the firewall)
- LUA programmable firewall rules
- Scalable/Decentralized solution

3.2 Diameter firewall - Technical capabilities

- Open Diameter encryption and signing of the Diameter messages, including auto encryption setup
- Diameter host and realms blacklists (Category 0)
- Diameter Command Code blacklists and Realm whitelist (Category 1)
- Diameter firewall rules (Category 2)
- Signalling IDS integration (for Category 3 and advanced detection)
- Diameter Filtering and honeypoting
- Centralized threat reporting with mThreat integration
- Collaboration with other Diameter and signaling security systems
- Management through open APIs
- Passive run (re-run traffic from pcap or passive interface to test the firewall)
- LUA programmable firewall rules

- Scalable/Decentralized solution

Additionally we will outline also the contribution which could be used for network monitoring and could be used in this domain but also in other domains.

- Tshark to Elasticsearch export and security monitoring with Kibana

4 The Current Status

In the following chapter we would be briefly outline the current possible approach regarding the message filtering and screening on the network boundaries.

4.1 SS7 / Sigtran stack overview

On the following figure is illustrated SS7/Sigtran protocol stack. This is important to understand for decoding and filtering reasons.

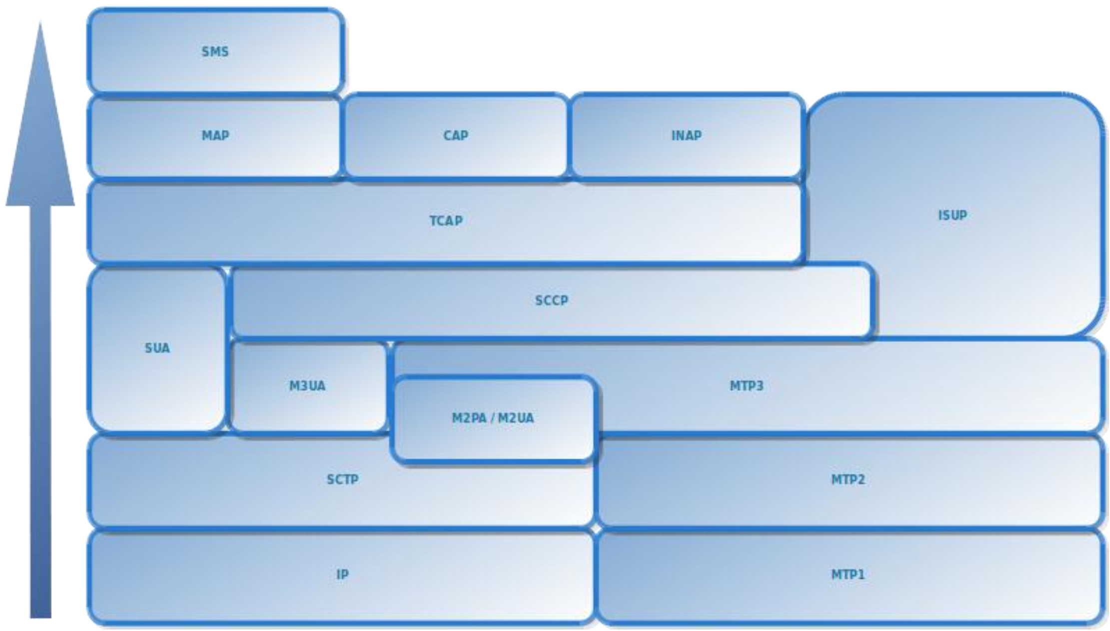

Figure 4.1 - SS7 and Sigtran stack

4.2 Perimeters of SS7 overview

The active filtering and the protection could be efficiently performed on the network boundaries and on the perimeters of the home network (HPLMN). We can consider mainly the following perimeters:

INAT 0: International interconnects (higher risk)

NAT 1: National interconnects (possibly lower risk)

There could exist the different security filtering for these perimeters. International interconnects are used mainly for inbound and outbound roaming subscribers. The national interconnects are commonly used for SMS delivery, roaming if the national roaming is allowed and forwarding signalling messages in case of number portability.

For overall security we should consider also other interfaces and interconnects e.g. with MVNOs or API towards SMSC and with 3rd party SMS aggregators.

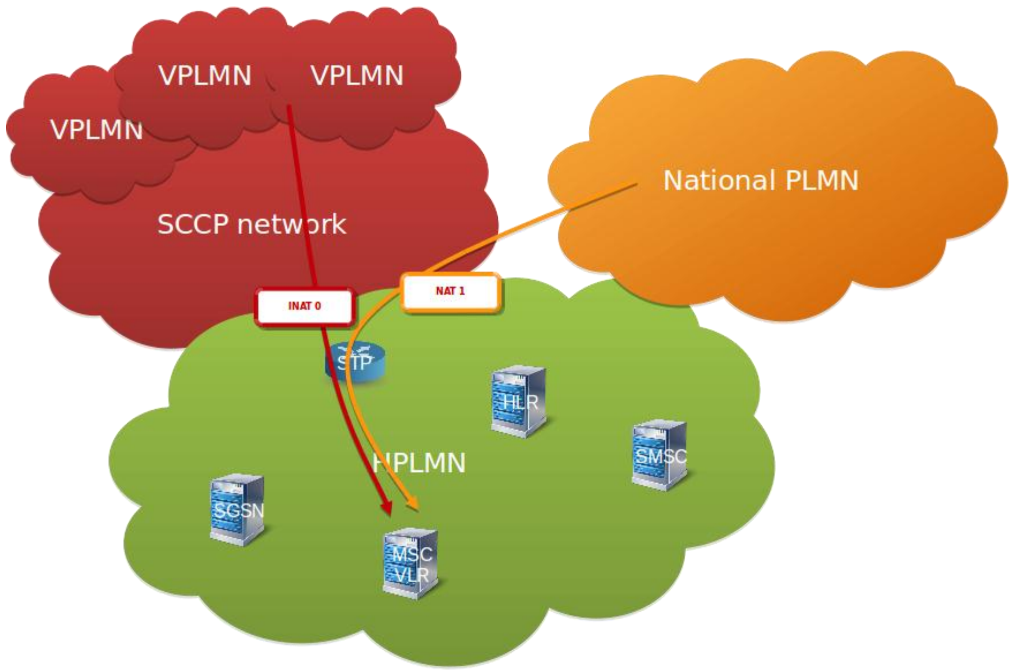

Figure 4.2 - SS7 perimeters

4.3 SS7 message categories

Category is just naming indicating the group of the similar messages. For messages in same category the same protection logic could be implemented. Mainly the message direction is important to decide into which category the message belongs. The normal call flows and normal use of the message is well described in 3GPP specifications.

MAP Cat1 messages are messages which should not be allowed towards HPLMN.
MAP Cat2 messages are messages which should be allowed towards HPLMN only if foreign network is targeting own subscribers (inbound-roamers).
MAP Cat3 messages are messages which should be allowed towards HPLMN from own subscribers in roaming (outbound-roamers) only if location condition matches.
SMS Cat: SMS messages which requires to decode SMS layer.

CAP Category 2 messages are Camel messages which should be allowed for inbound-roamers from HPLMN towards foreign network (inbound-roamers).
CAP Category 3 messages are Camel messages which should be allowed for outbound-roamers from VPLMN towards HPLMN.

From the above approach the messages could be classified into message categories and could be created protocol matrixes for SS7 but also for Diameter and GTP protocol. Then the protection could be implemented in the Signalling Firewall or in the Network Elements.

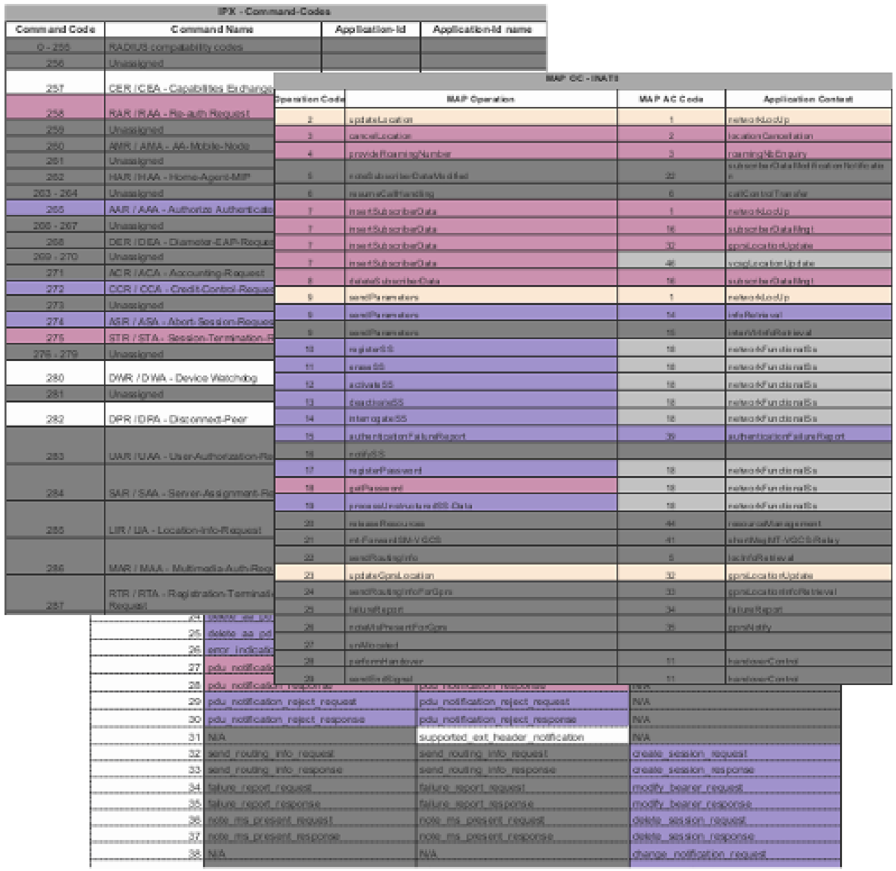

Figure 4.3 - Protocol matrixes with the message categories

4.4 SS7 screening categories grouped by protocol layers

The logic for message filtering could be grouped into screening categories blocks. The figure below illustrate this approach by defining groups with the same detection and filtering logic.

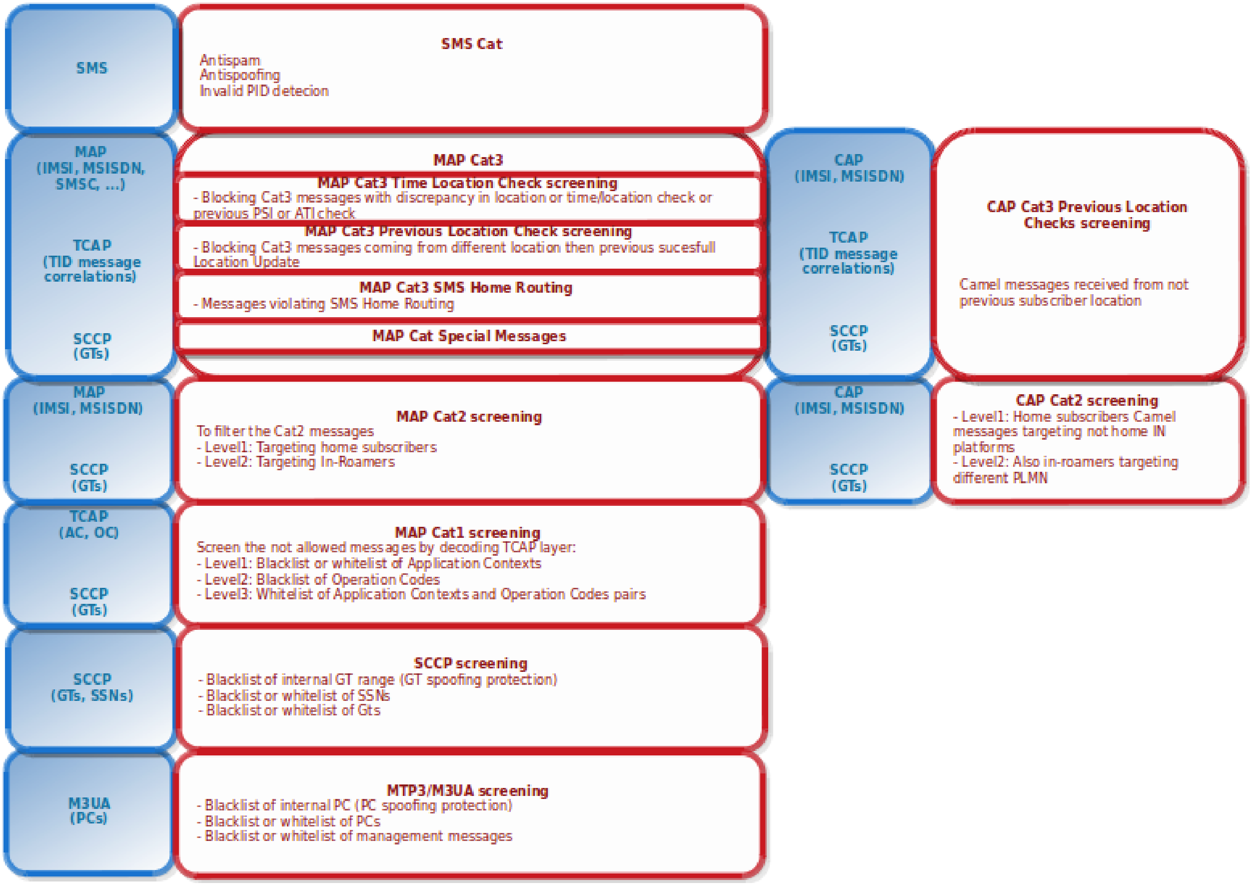

Figure 4.4 - SS7 screening categories with protocol layers

4.5 Possible SS7 filtering by existing infrastructure without FW

The filtering is possible also inside the infrastructure without having external firewall, but there are several disadvantages in this approach. (e.g. no perimeter defense, no centralized control)

Also in this approach it is hard to manage the confidentiality and integrity protection of signalling messages.

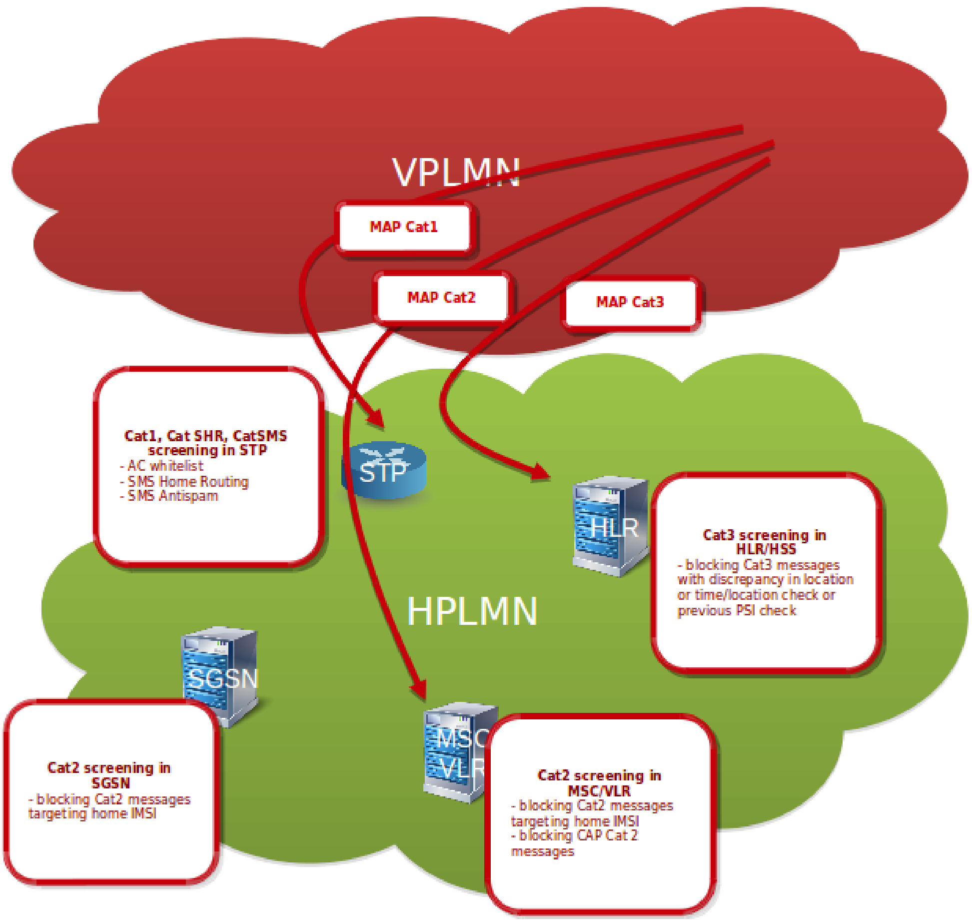

Figure 4.5 - SS7 network protected by existing infrastructure

4.6 Current status conclusion and acknowledgement

In this chapter was briefly outlined the message filtering approach on the network boundaries.

The above figures illustrates the internal research/approach but the work is inline and evolves the current GSMA recommendations. Additionally we are contributing in this direction to GSMA.

For further details of the GSMA collaborative work it could be referred to FS.11, FS.19 and FS.20 GSMA documents.

5 Advanced SS7 Attacks

In the following chapter are highlighted some attacks as examples to demonstrate the message categories. Then this is followed by examples how the protection could be bypassed while the subscriber is in roaming.

5.1 Category 2 attack example - VLR profile manipulation

Category 2 example - VLR profile manipulation. The attacker could manipulation the profile of the subscriber in the VLR.

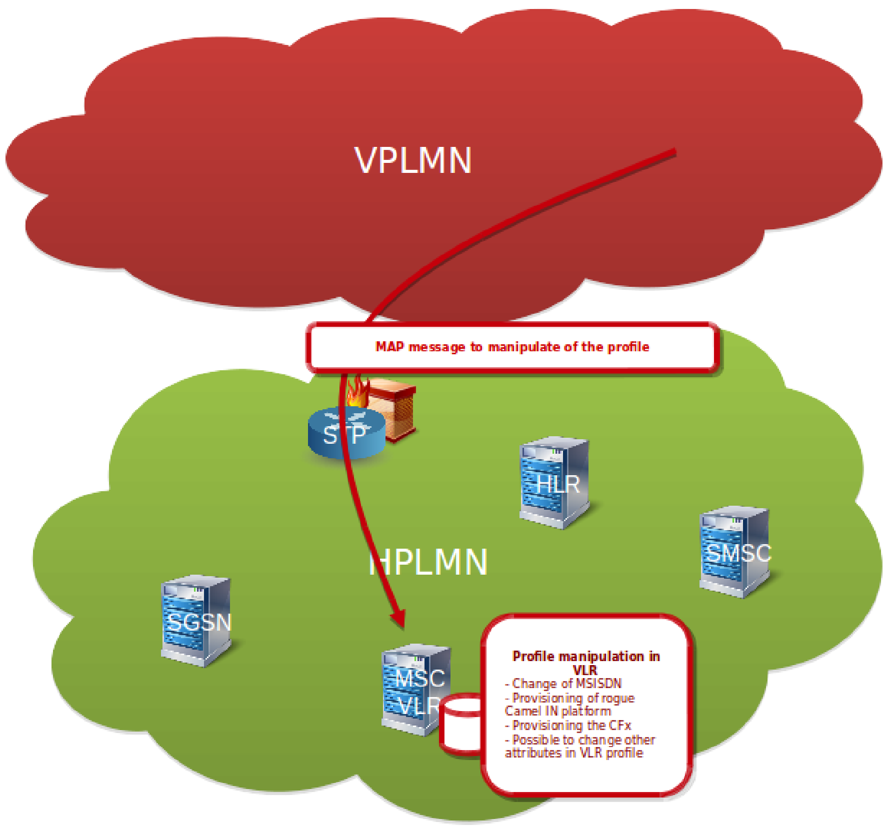

Figure 5.1 - VLR profile manipulation

Description: The figure illustrate that the attacker can craft the MAP ISD message and target the MSC/VLR which is currently serving the subscriber. It there is no protection against Category 2 attacks the attacker is able to alter the VLR profile from the attacker's GT. If in the HPLMN is Signalling FW or the protection against Category 2 attacks, the attack would fail because the attacker's GTs will belongs to different country as the HLR of the targeted subscriber.

Impact: The attacker can manipulate the whole VLR profile which could lead in the modification of MSISDN, tele/bearer services, supplementary services, baring, camel flags and the provisioned IN platform. The possible impact is the call and SMS interception, persistent location tracking, frauds or targeted DoS of the subscriber.

5.2 Category 2 attack example - GPRS/LTE profile manipulation

Category 2 example - GPRS/LTE profile manipulation. The attacker could manipulation the profile of the subscriber in the SGSN/MME.

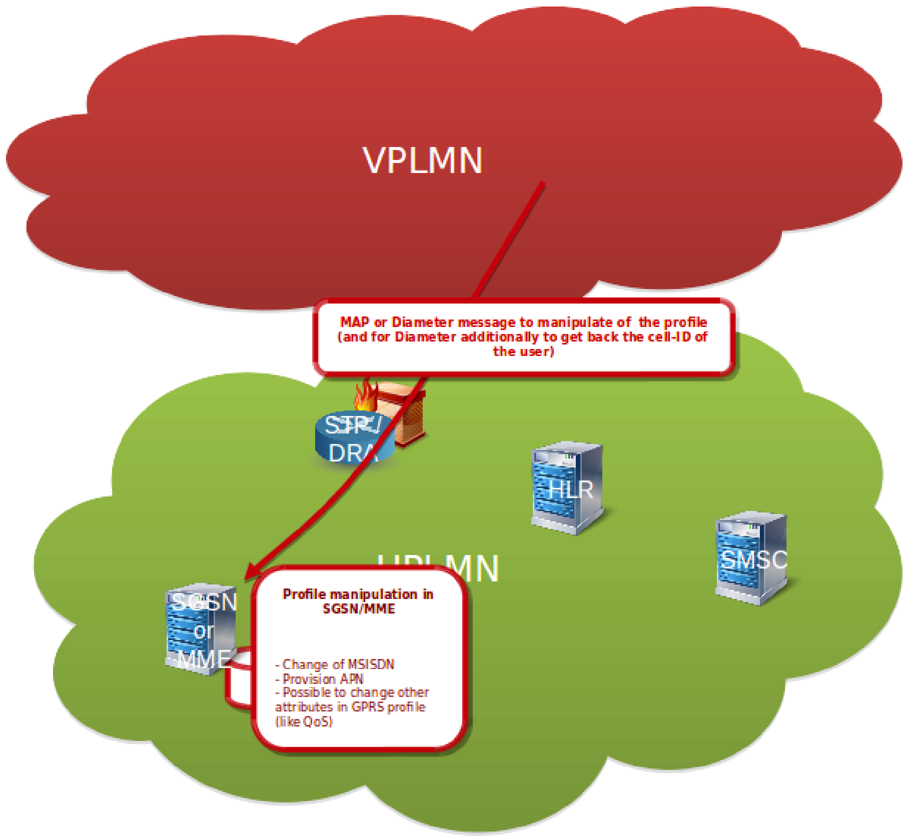

Figure 5.2 - SGSN/MME profile manipulation

Description: The figure illustrate that the attacker can craft the MAP ISD or Diameter IDR message and target the SGSN or MME which is currently serving the subscriber. It there is no protection against Category 2 attacks the attacker is able to alter the SGSN/MME profile from the attacker's GT (or Diameter Origin-Host/Realm. If in the HPLMN is Signalling FW or the protection against Category 2 attacks, the attack would fail because the attacker's GTs (or Diameter Origin-Host/Realm) will belongs to different country as the HLR/HSS of the targeted subscriber.

Impact: The attacker can manipulate the whole GPRS/LTE profile which could lead in the modification of MSISDN, APNs, QoS, camel flags and the provisioned IN platform. The possible impact is the bypass of MSISDN authentication (if HTTP enrichment and latter MSISDN authentication is used), access to private APNs and the possibly the data interception if the latter Camel is enabled in the Packet Core.

5.3 Category 3 attack example - Hostile Location Update

Category 3 example - Hostile Location Update. The attacker could change location in the HLR/HSS.

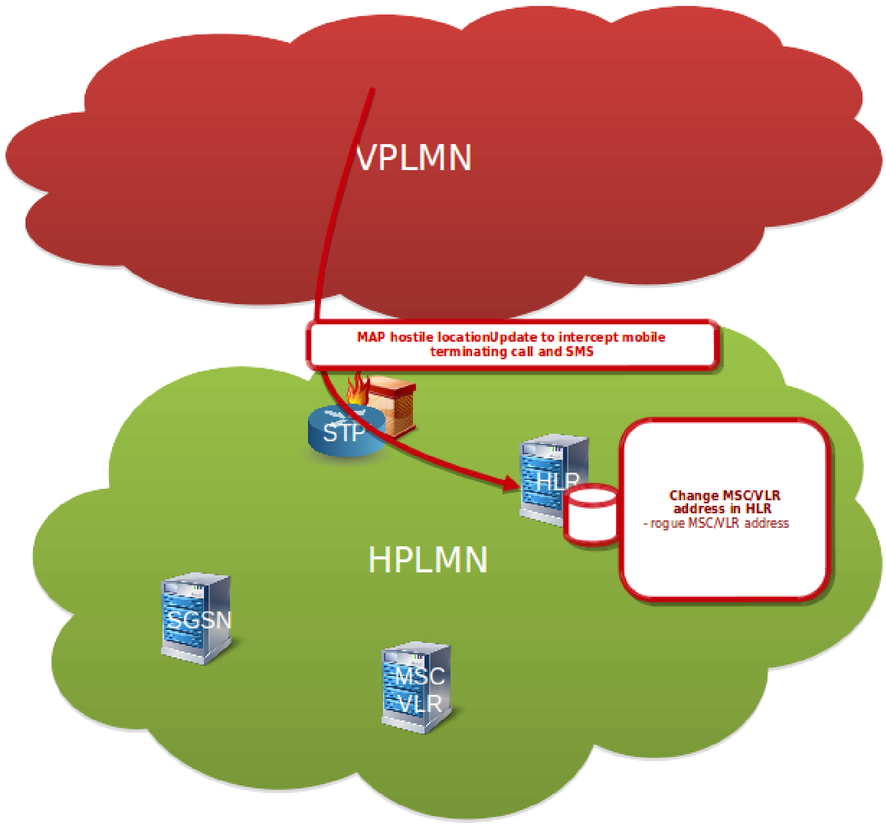

Figure 5.3 - Hostile Location Update

Description: The figure illustrate that the attacker can craft the MAP LU message towards the HLR/HSS and change the location of the subscriber to own GT. If in the HPLMN is Signalling FW or the protection against Category 3 attacks, the attack would fail because the Location Update would be interpreted as suspicious if coming from too different location compared to current location of the subscriber.

Impact: The attacker can change the subscriber GT in HLR/HSS. This could lead into MT-SMS interception, possibly MT-Call interception if the attacker can also connect the original B-party after or targeted DoS of the subscriber. Additionally could be used also as precondition for latter Category 3 attacks.

5.4 Category 3 attack example - Register/Activate SS

Category 3 example - Register/Activate SS. The attacker could manipulate the supplementary services in HLR/HSS.

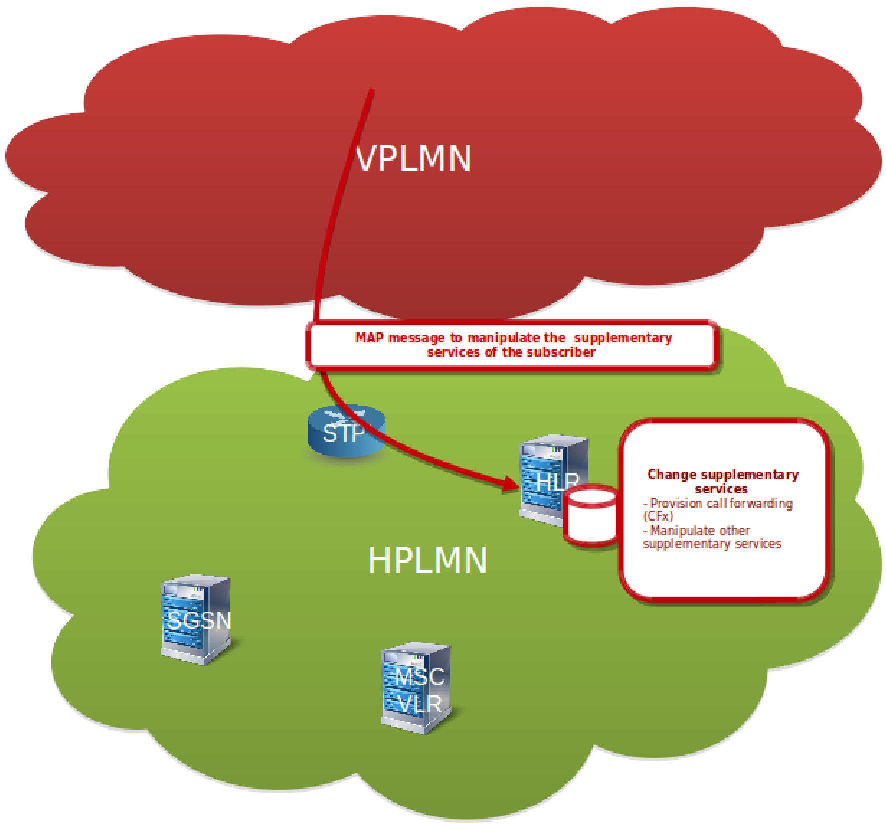

Figure 5.4 - Register/Activate SS

Description: The figure illustrate that the attacker can craft the Register/Activate SS message and target the HLR/HSS. It there is no protection against Category 3 attacks the attacker is able to alter the SS services in HLR. If in the HPLMN is Signalling FW or the protection against Category 3 attacks, the attack would fail because the attacker's GTs will not match with the current subscriber location.

Impact: The attacker can manipulate the whole SS service in HLR/HSS, which could lead on activation of call/SMS forwarding and other SS manipulation.

5.5 Category 2 protection bypass

Outbound-roamer in VPLMN: Attack targeting outbound-roamers with Cat2 messages with spoofed calling GT.

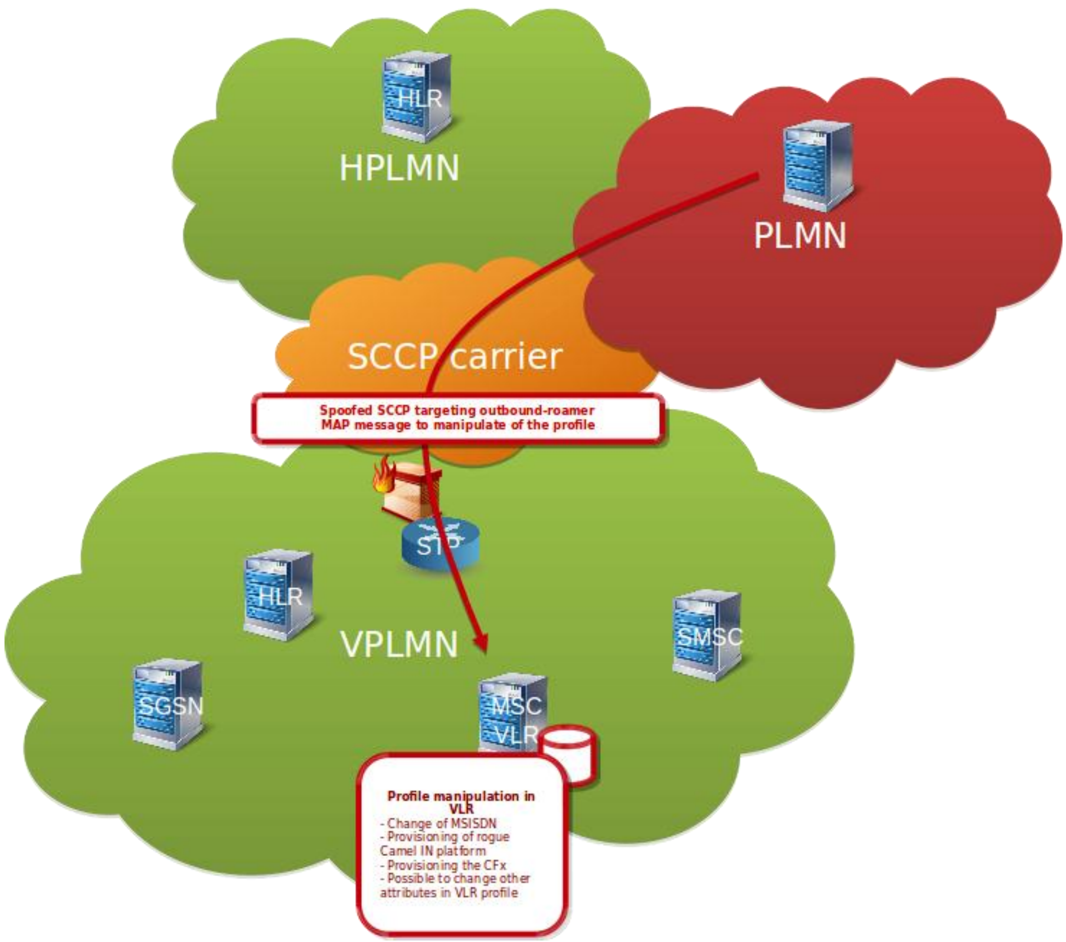

Figure 5.5 - Category 2 protection bypass

Description: The figure illustrate that when the subscriber is located in roaming network (VPLMN) and if the attacker knows his VLR/SGSN address (e.g. discovered by other SS7 messages, like SRI-SM or by passive sniffing), the attacker can send spoofed Cat2 SS7 messages and impersonate subscriber HLR from HPLMN. For such attack the signalling firewall in VPLMN would be not able to discard the message and differentiate it from legitimate signalling, because the message is spoofed with the correct Calling SCCP Address.

Impact: Subscriber in roaming could not be easily protected against spoofed SCCP attacks or could be difficult if the Calling SCCP Address is from same country as the legitimate one. This results into possible VLR and SGSN profile manipulation, which could lead into setting call forwarding, removing services, provisioning Camel services and other. (DoS, tracking, interception). For spoofed messages for SS7 the attacker would not get result message but for Diameter would, because of the Route-Record AVP.

5.6 Category 3 protection bypass

Outbound-roamer in VPLMN: Attacker first performing hostile LocationUpdate (if not working could use spoofed Cancel Location first). After performing Cat3 messages.

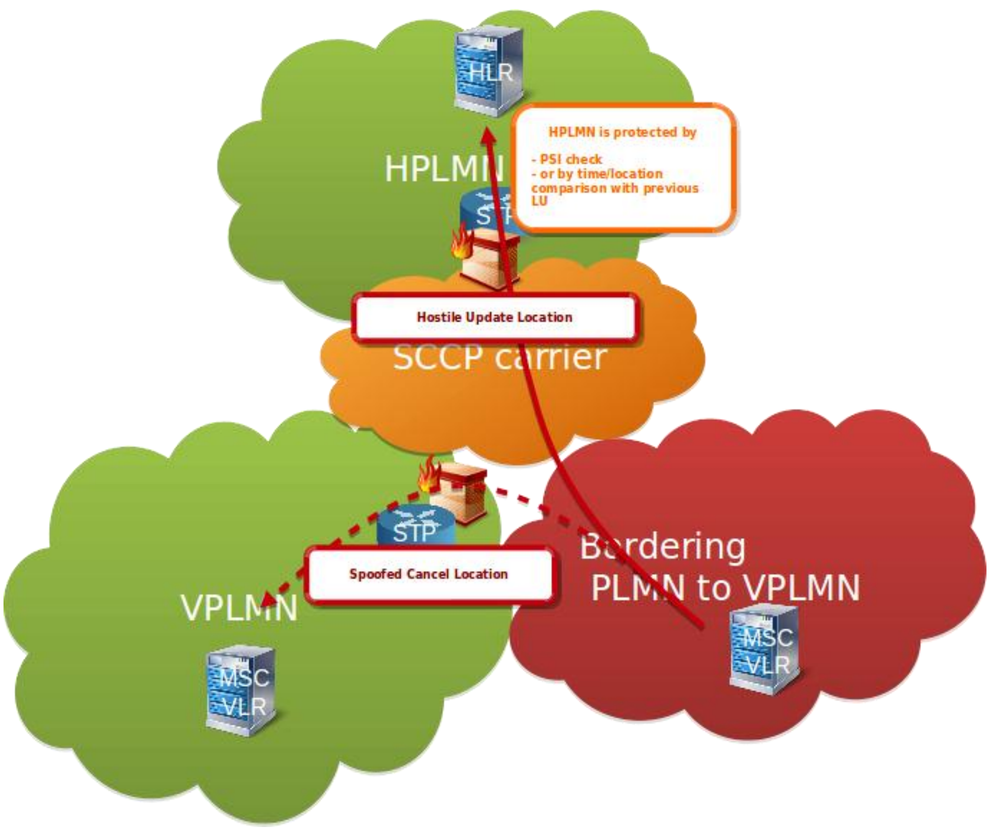

Figure 5.6 - Category 3 protection bypass

Description: The hostile Location Update sent by attacker will try change the VLR/SGSN address in the HLR first before sending later Category 3 messages. The reason for this is that in HPLMN is implemented Signalling Firewall or other protection against Category 3 messages with the following behaviour.

Option 1. - the protection in HPLMN for Cat3 messages is implemented by sending PSI message to previously known subscriber location, to verify that the subscriber is not located anymore there. This protection is possible to bypass by the hostile location update first.

Option 2. - the protection in HPLMN is implemented by time/distance analysis of previous and current location updates. This is possible to bypass by sending the hostile location update from not suspicious location (e.g. bordering country).

If the Hostile Location Update is not successful, the attacker can try to first send the spoofed Cancel Location to the current MSC/VLR to bypass any PSI checks and then try send again LU or any other Cat3 messages.

Additionally attacker can also spoof directly the calling GT of latter Category 3 messages if knows the current subscriber location.

Impact: Hostile Location Update could lead directly to DoS, SMS interception and call interception (in case the attacker is capable of receiving media and connect back the B-party). Further also enables to attacker to send later the Cat3 messages (e.g. supplementary services activation, mobile originating SMS, USSD and other) because the protection by comparing the previous subscriber location with origin of the message would be bypassed.

5.7 MITM

Description: Not encrypted SCTP protocol used for Sigtran and Diameter is vulnerable to man-in-the-middle attacks. See below extract from RFC.

```
SCTP (RFC 3257)
5.3 Security Issues with both TCP and SCTP
   It is important to note that neither TCP nor SCTP protect itself from
   man-in-the-middle attacks where an established session might be
   hijacked (assuming the attacker can see the traffic from and inject
   its own packets to either endpoints).
```

Impact: Attacker could get access into SS7 network by MITM in SCTP without being configured or provisioned on SS7 network. By having such capability motivated attacker with physical access to links could inject traffic into signalling network. This means not only attacker having SCCP address and connectivity with STP or with other network element could get access into SS7 network. Additionally in MITM scenario further attacks are possible, like ISD/profile modification, authentication vectors modification (RES, IK, CK, AUTN), modification and integrity changes also of SS7 Result messages.

5.8 Passive Attacks

Description: SS7 signalisation is not confidentiality protected.

Impact: This could be used for mass collection of signalling data includes mainly:

- SMS content with A-party, B-party information
- Locations (MAP, CAP, Diameter)
- From SS7 MAP possible to get CK, IK
- Get TCAP TID which could be used for latter attacks

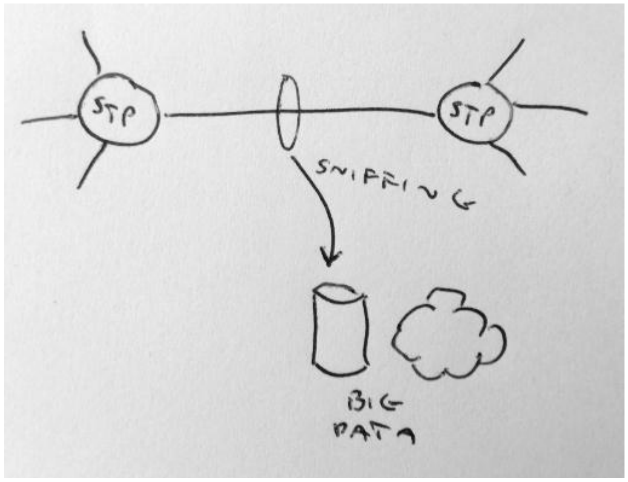

Figure 5.8 - SS7 passive attack

5.9 Combining Passive and Active Attacks

Description: By knowing the TCAP TID in real time and exact user location it could lead to more sophisticated attacks. And if the attacker is able to capture the result messages answered to spoofed messages this will also increase the capabilities.

Impact:

- Injection of messages into TCAP dialog, possibly hijacking the state machine in network elements and other effects
- Camel manipulation towards the IN platforms
- Better targeted spoofing of the SCCP messages
- Capturing the result messages to spoofed messages

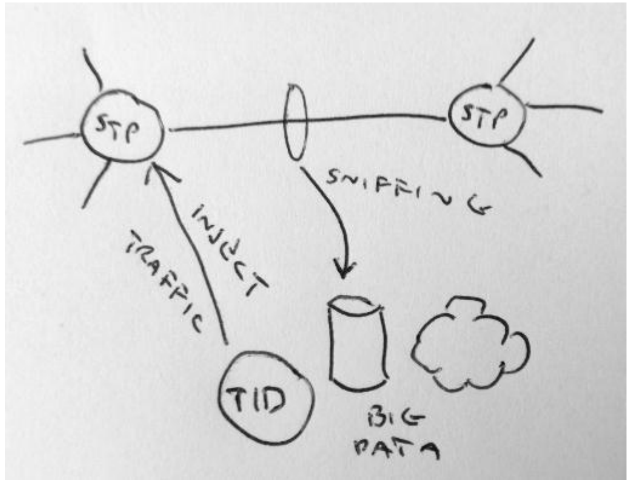

Figure 5.9 - SS7 passive and active attack

5.10 Malformed messages

Description: There is various ways of manipulating and malforming the messages. This could lead into exploitation of the vulnerability in the specific product/version of the network element.

Impact: Could lead to DoS or Exploitation (even DoS of the whole network)

5.11 Advanced Attacks Conclusion

To address the above advanced types of attacks the signalling should be confidentially and integrity protected.

A firewall with only filtering could well protect the home subscribers in HPLMN. But the home subscribers in VPLMN or inbound-roamers in HPLMN could not be easily protected mainly because the SS7, Diameter is vulnerable to spoofing and the Location Update is not authenticated.

The encryption can be done on TCAP layer or Diameter/AVP. (the current work is using proprietary implementation using asymmetric encryption)
Messages can be integrity protected carrying signature. (the current work is using proprietary implementation

**IPSec is not suitable, because the SCCP and IPX network is required to perform routing.*

6 SigFW

Open-source SigFW

- SS7 and Diameter Firewall created under P1 Labs
- Source code is available at https://github.com/P1sec/SigFW

The open-source SigFW should be considered as **reference implementation** and **research project** but **without any warranty** and it is not carrier grade solution.

6.1 Open SS7 Firewall

The SS7 firewall could be considered as roaming and interconnection protection (the reference implementation) for 2G and 3G networks.

6.1.1 Architecture

Frames are forwarded on SCCP layer (using SCCP state-machine).
Filtering is possible up to application layer (in code is currently implemented SCCP, TCAP, MAP).

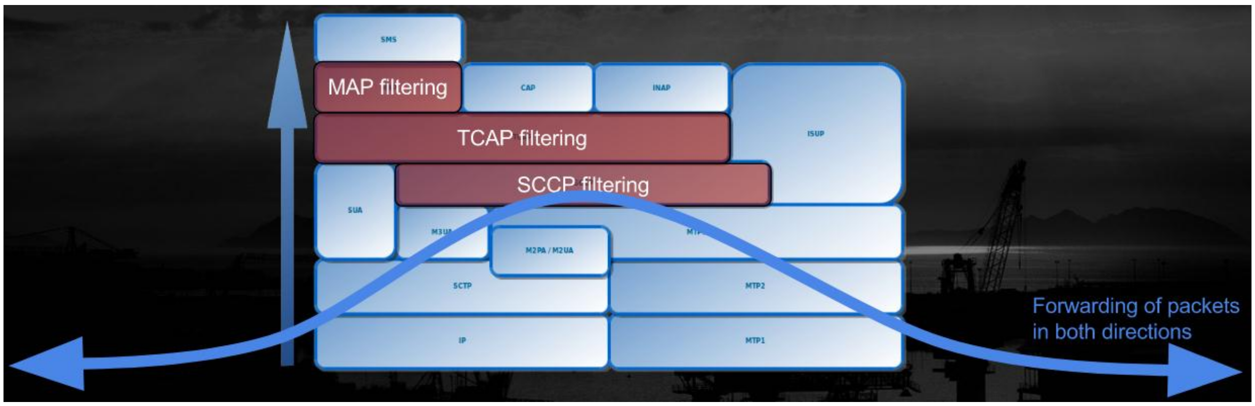

Figure 6.1.1a - SS7 Firewall decoding and filtering

Firewall is acting like M3UA server and M3UA client, without having SCCP GT.
Below is an illustration of the direction of links and associations establishment.

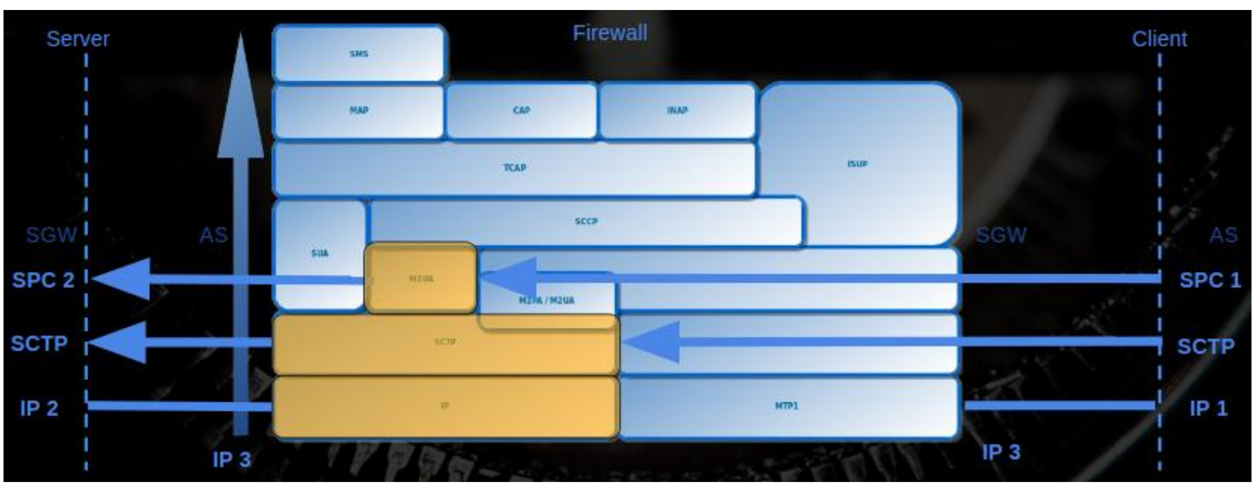

Figure 6.1.1b - SS7 Firewall connections

6.1.2 Deployment

Possible deployment can be loopback on STP towards the FW. Also other deployment scenarios could be FW deployed directly on the link or FW just protecting single network element.

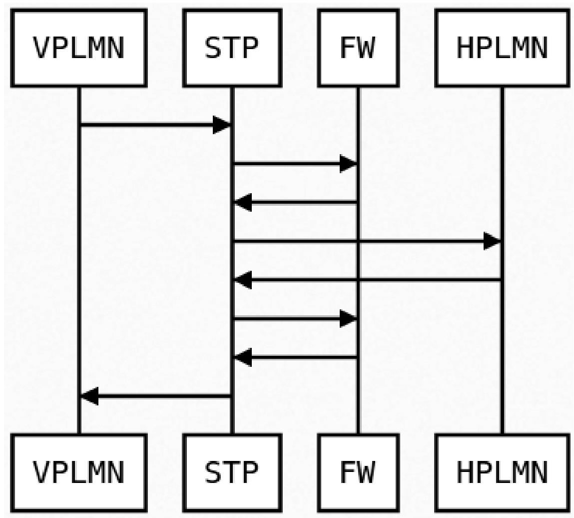

Figure 6.1.2 - SS7 Firewall deployment

6.1.3 APIs

The following REST API are currently implemented on the firewall. The API allows the remote management, provisioning the firewall rules or evaluating the messages or reporting the alerts.

1. Signaling Filter Push API (Manage Firewall Rules)
2. Signaling Message Evaluation API (Message evaluation with external IDS signalling system)
3. mThreat API (to report the detected attacks)

6.1.4 Config

- JSON syntax
- IP, SCTP, M3UA configuration
- Firewall filtering rules
- Encryption and signature keys
- Config is periodically saved to store the changes (changes over API or collected Public Keys if autodiscovery is enabled)

Figure below is the example of the configuration file. For full examples for both SS7 and Diameter see annex.

Figure 6.1.4 - SS7 Firewall config example

6.1.5 Signaling Message Evaluation API

Signalling Message Evaluation API can be used to forward the messages which has not been detected by internal firewall rules to evaluate them in the IDS platform with more advanced detection capabilities.

- FW forwards the SCCP message to Signalling IDS
- Signalling IDS responds back with the result (allow/filter message)
- FW performs the filtering action
- By this integration no need for FW to contain own centralized DB and there could be deployed multiple FW instances
- Signalling IDS can handle more advanced Cat2, Cat3 detection, anomaly detection or threat intelligence decision

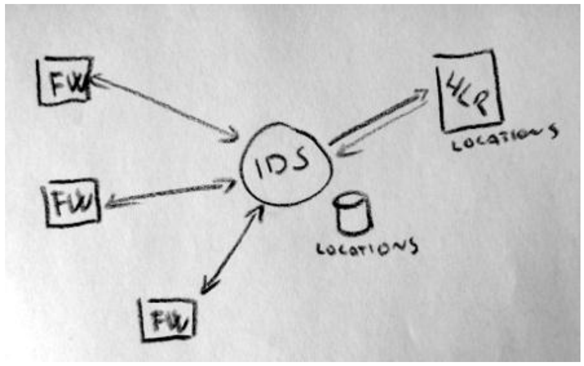

Figure 6.1.5 - SigFW with ISD integration

6.1.6 SS7 Firewall Passive Mode

The firewall can be first tested in passive mode without establishing any active signalling link. The traffic can be mirrored and be send to the FW passive network interface or the pcap/json can be directly replayed. Then the traffic is replayed on the localhost through the local client, firewall and towards the local server.

Passive mode is implemented in VM the following way:

1. tshark live capture to Json EK
2. SS7ClientLiveInput is reading sccp_raw from named pipe and forwarding it to FW
3. SS7FW performs the filtering
4. SS7Server receives the not filtered traffic

Example of replayed traffic on localhost "Passive mode":

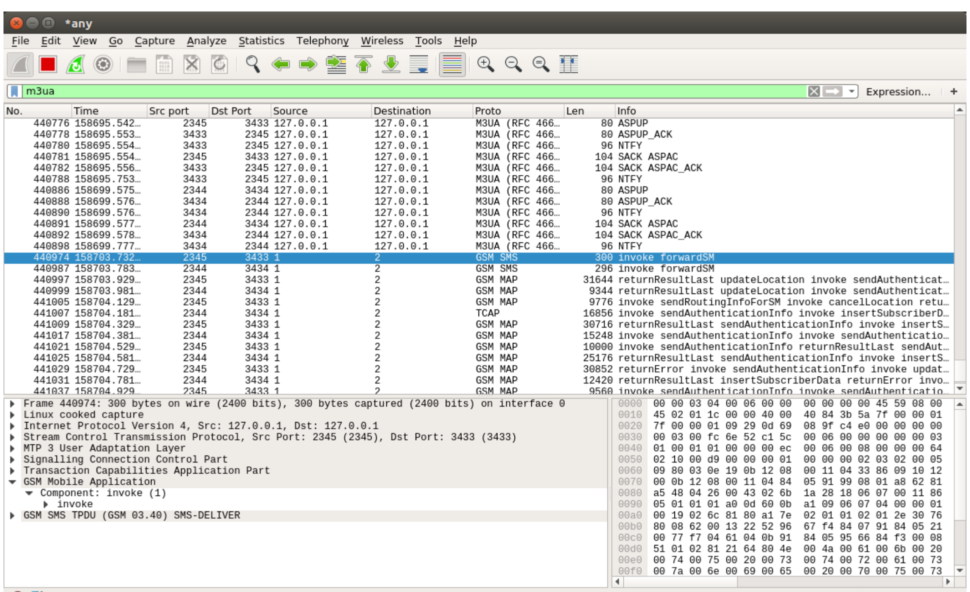

Figure 6.1.6 - SS7 Firewall passive mode

6.1.7 SS7 Encryption

Current version is capable additionally of

- Signing/Verify the SS7 message
- Encrypting/Decrypting SS7 messages

Public/Private keys are used and the security model is similar to email security (signing, encrypting).

Encryption is performed on TCAP level to pass through the STPs.
SCCP layer is not encrypted, but the SCCP addresses are used to calculate signature.

```
"encryption_rules": {
                              "called_gt_encryption": [
                                        {
                                                  "called_gt": "0*",
                                                  "public_key":
"MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCm/PAsXOj7cjirJsQsiIeHauFNLwBIuM1brkUm3aVXeraDIeJ2BWXmWlKMmX/FRZh4Qhe9mUy6YgwTO8PndWdMDRWM
w8vvXJFI7HPJpsNfcBykefSqhr5X4h6HyQr73V8O0U5PtgCBuVoyuOFIj87WFwaLuajHiQgps7NOloeH1wIDAQAB"
                                        }
                              ],
                              "called_gt_decryption": [
                              ],
                    },
                    "signature_rules": {
                              "calling_gt_verify": [
                              ],
                              "calling_gt_signing": [
                              ]
                    }
```

Figure 6.1.7 - SS7 Firewall encryption defined in the config

6.1.8 SS7 Encryption Flow

The below figure illustrates the encryption flow. The FW#1 instance in PLMN#1 encrypt the signaling messages towards the PLMN#2 because the messages matched with the GT prefix of the PLMN#2 network. The FW#2 instance in PLMN#2 network decrypt the traffic and forwards it into PLMN#2 network. The reverse direction is performed in the similar way that the FW#2 instance matches the message called GT with the GT prefix of PLMN#1 network and use the associated public key for message encryption. The messages in current model are encrypted individually without establishing session.

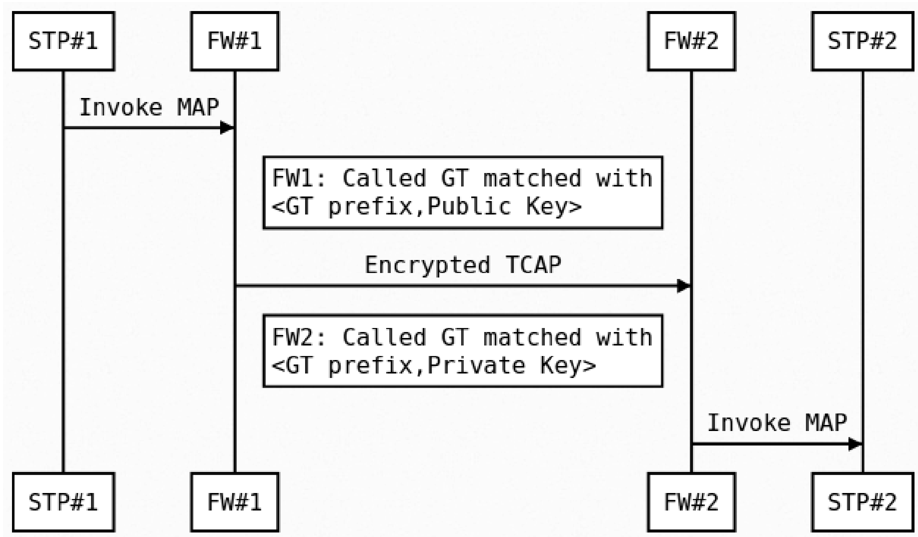

Figure 6.1.8 - SS7 Firewall encryption flow

6.1.9 SS7 Encryption Algorithm

1. Encrypted is the whole TCAP layer
2. Encrypted is the following payload:
    a. version (4 bytes)
    b. encrypted(timestamp (4 bytes) + tcap_layer) // If the key is short the multiple similar blocks are created
3. Encryption algorithm should be mapped with version. Currently in the code only RSA/ECB/PKCS1Padding is used
4. Timestamp is verified after decryption to prevent replay attacks

6.1.10 SS7 Encryption Example

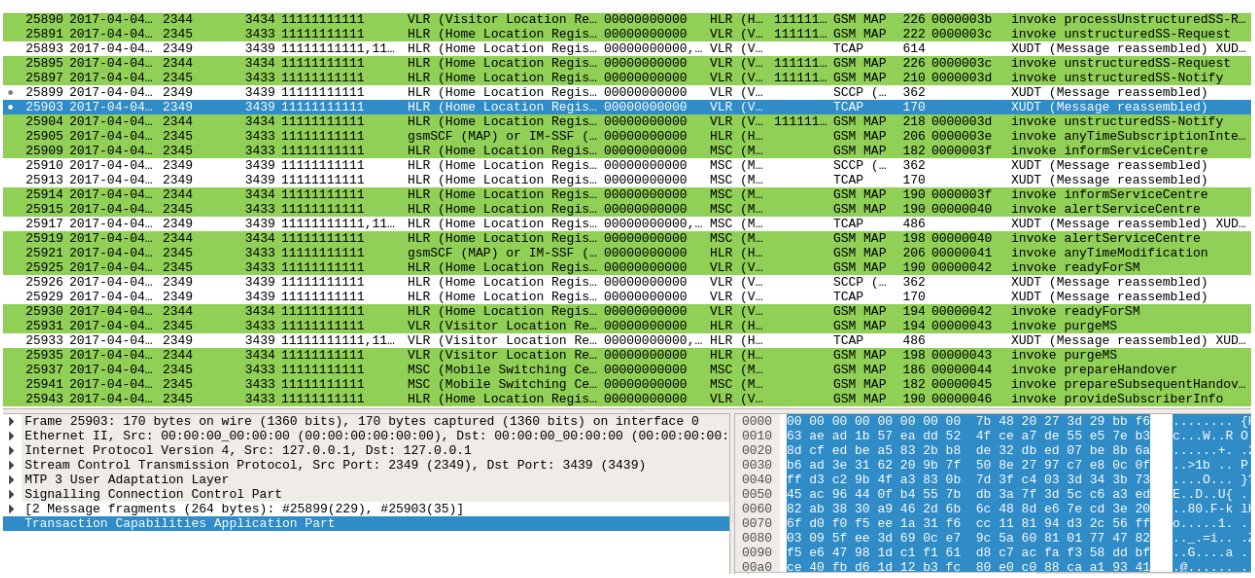

Figure 6.1.10 - SS7 encryption example

6.1.11 SCCP UDT / XUDT

On previous figure has been seen XUDT messages.

The XUDT is used instead of UDT if the payload size has increased and reached the maximum limit of UDT message.
After decryption on the other end the message are again reconstructed into UDT message.

This is the limitation of the current solution, that the SCCP provider have to support and route the XUDT messages.

6.1.12 SS7 Encryption Autodiscovery

Firewall feature to enable encryption autodiscovery. The autodiscovery should enable easier initial key management to receive the public key over the signalling.

1. The FW #1 will send MAP Invoke (New OpCode 99) for destinations with no known Public Key
2. If there is FW #2 in path, it process the Invoke and send Result (including GT prefix and Public Key)
3. FW #1 config is updated with gathered public keys

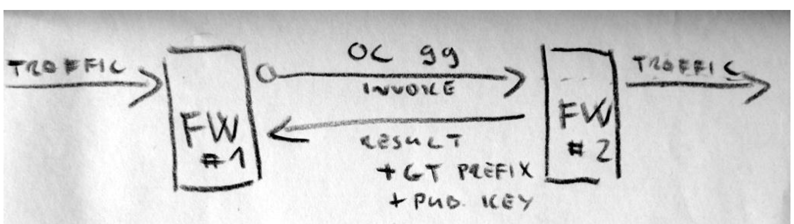

Figure 6.1.12 - SS7 Firewall autodiscovery

Limitation is that during the initial autodiscovery the remote party is not authenticated. If the remote key has expired or has been changed, the public key stored on FW#1 instance can be deleted to re-trigger the autodiscovery again. But during this process the above security aspect should be again considered and manual key management should be understood as more secure.

6.1.13 SS7 Encryption Flow - autodiscovery

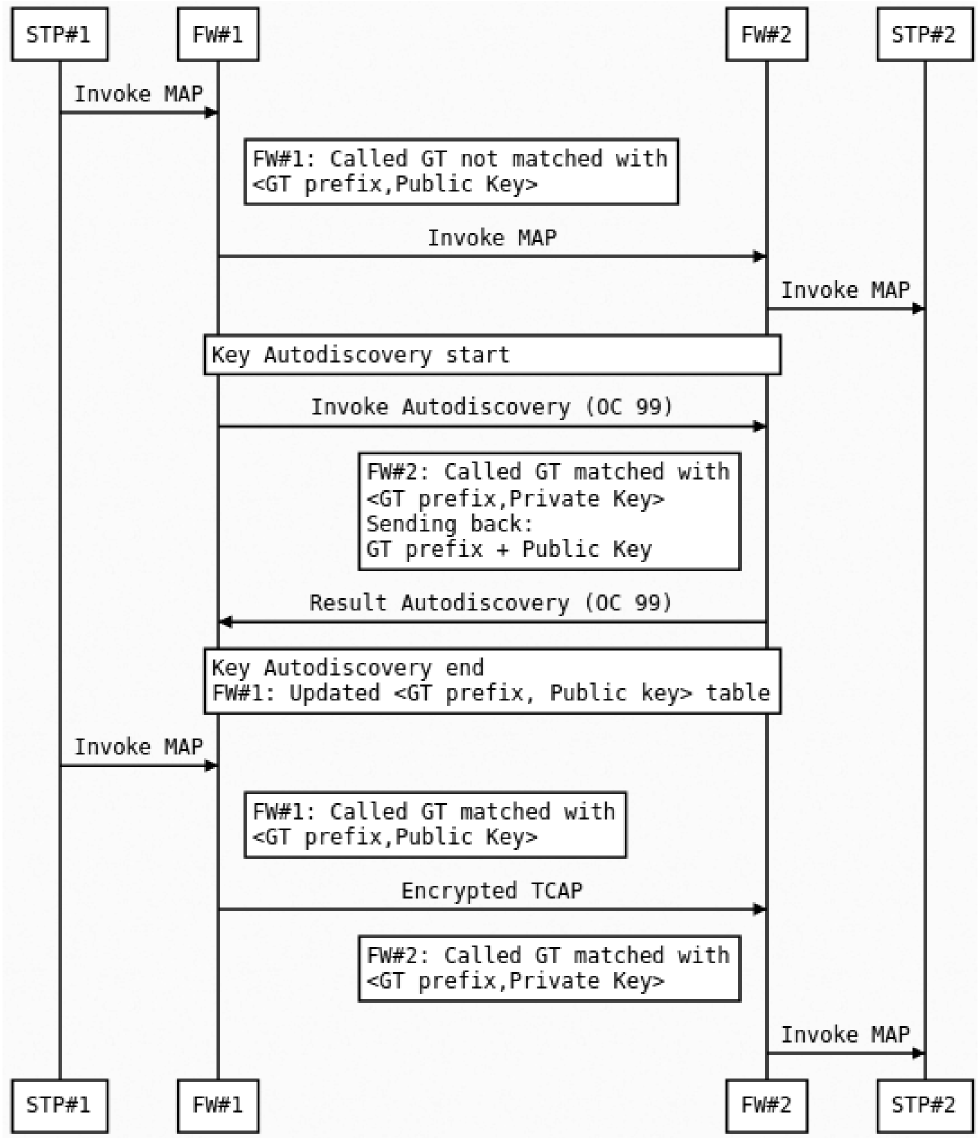

Figure 6.1.13 - SS7 Firewall autodiscovery flow

6.1.14 SS7 Signature

For every TCAP Begin, the second Invoke is added containing the TCAP signature.

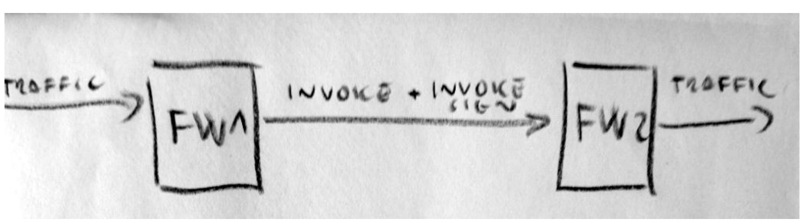

Figure 6.1.14 - SS7 signature

6.1.15 SS7 Signature Algorithm

1. Only TCAP Begins are signed
2. Check if the TCAP already contains some TCAP Invoke signature component. If not, sign it.
3. TCAP signature component will contains:
    a. Version
    b. Timestamp
    c. Signature
4. Signature is calculated:
    a. String dataToSign = calling_gt_digits + called_gt_digits + timestamp + tcap_layer
    b. String tcap_layer = base64(tcap_component_1) + … + base64(tcap_component_N);
    c. String dataToSign is then hashed (currently in code SHA256WithRSA is used)

6.1.16 SS7 Signature Example

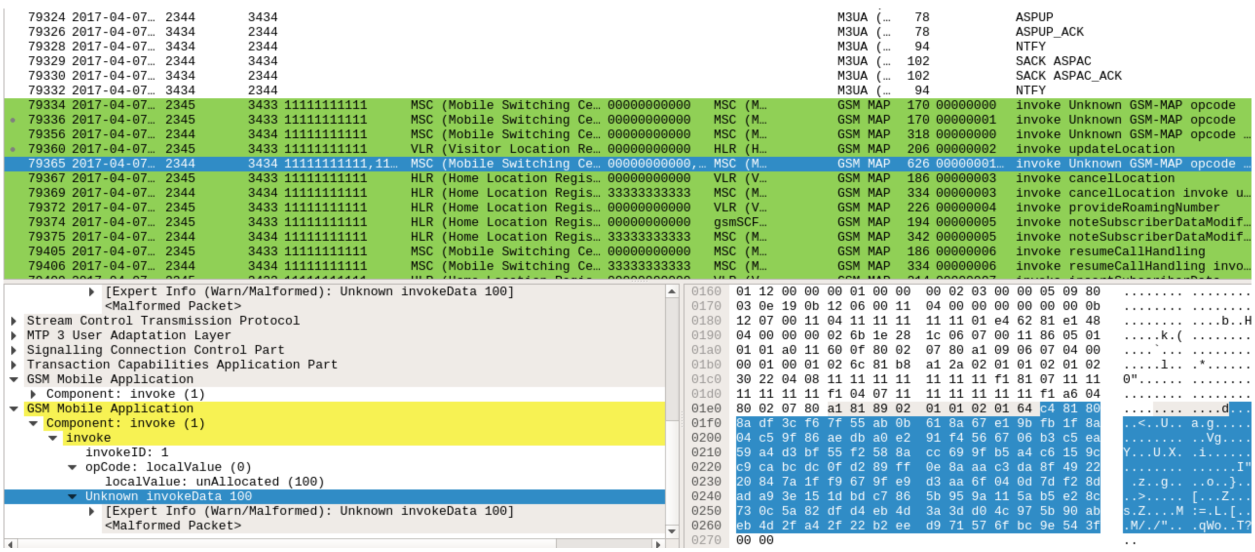

Figure 6.1.16 - SS7 signature example

6.1.17 DNAT to Honeypot

After detecting an attack the FW will perform DNAT for a defined time period for the attacker's GT.

By this approach the signalling honeypot can process the messages and send back the fake results. Additionally most time the attacker performs first the vulnerability probing of the target network and only if the network is vulnerable than conducts the real attack. Honeypot could also enable to capture such latter messages and multistage attacks performed by attacker.

Interesting data collected on the honeypot could be who is the victim of the attack, the attacks parameters (e.g. call forward to number or gsmSCF address) and to collect the whole attack sequence.

From the attacker perspective the interpretation of the results would become more difficult because it could be expected that also fake results could be returned from the networks.

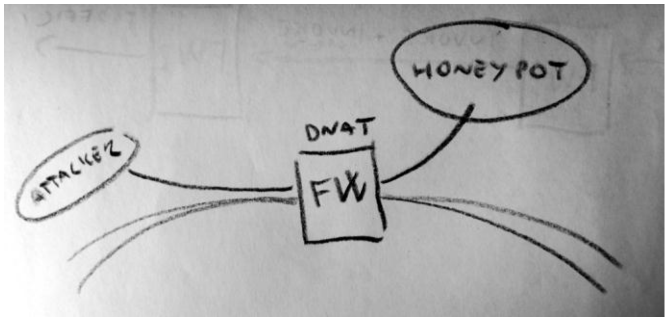

Figure 6.1.17 - DNAT to honeypot

6.1.18 DNAT to Honeypot Example

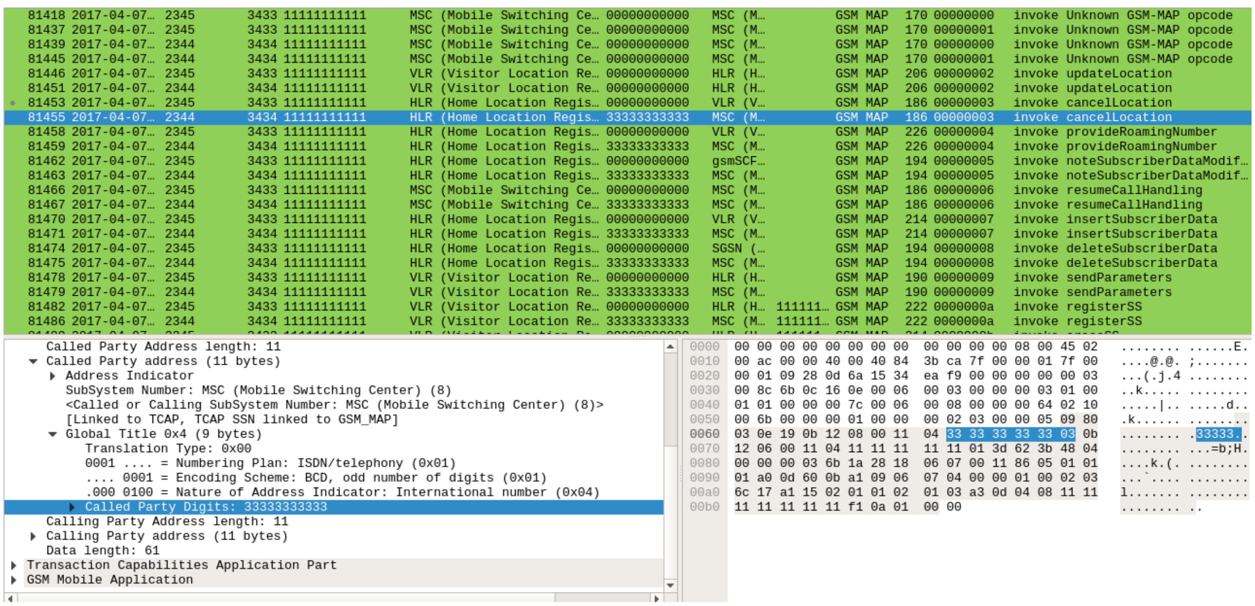

Figure 6.1.18 - DNAT to honeypot example

6.1.19 mThreat

Every firewalled event can be anonymized and send to mThreat. This optional capability and the mThreat URL should be first enabled in the configuration file. Only non sensitive information are sent and the IMSI and MSISDN are anonymized first. The salt used in hash function can be changed in the configuration file.

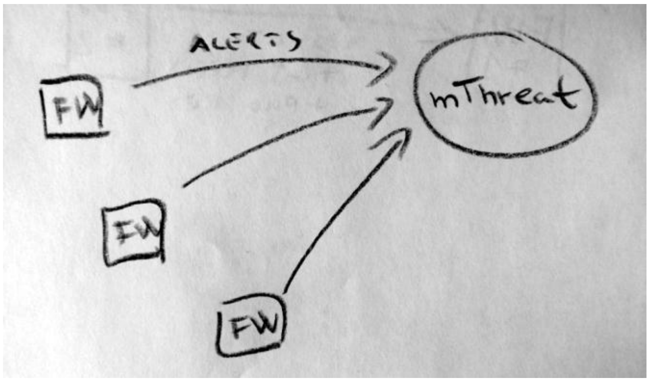

Figure 6.1.19 - SigFW reporting alerts to mThreat

6.1.20 mThreat Example

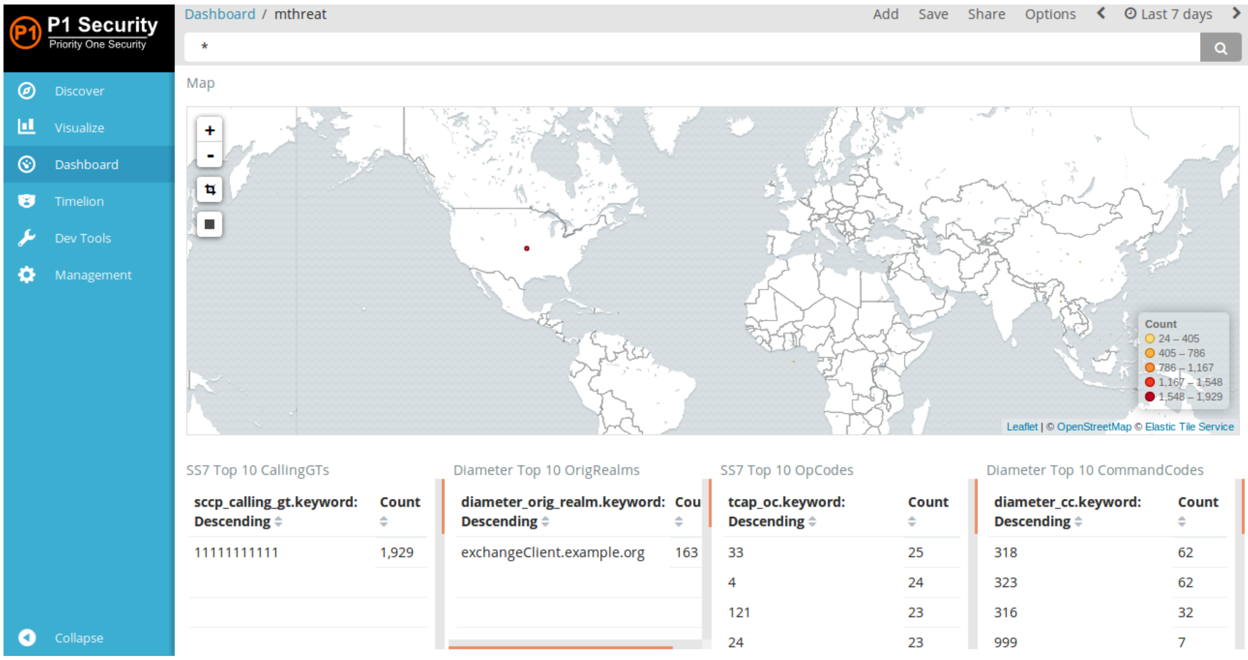

Figure 6.1.20 - mThreat UI using Kibana and Elasticsearch example

6.2 Open Diameter Firewall

The similar functionality has been developed for the Diameter protocol for 4G/LTE networks.
The similar capabilities are included.

6.2.1 Architecture

Frames are forwarded on SCTP layer.
Filtering is possible up to application layer (Diameter layer).

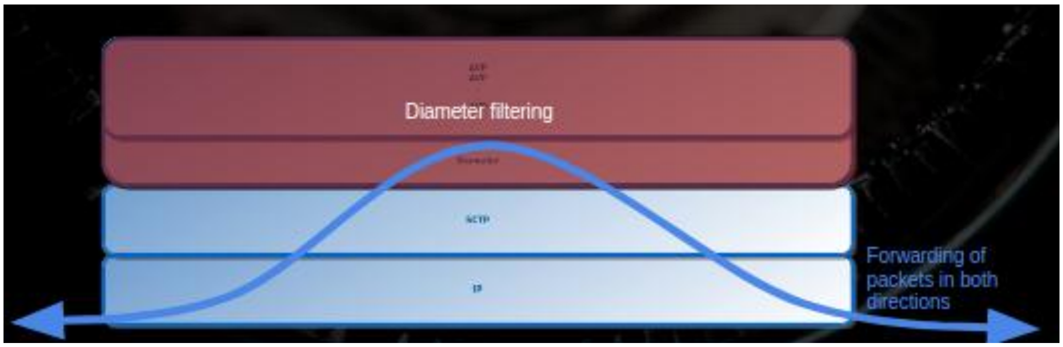

Figure 6.2.1a - Diameter Firewall decoding

Firewall is acting like SCTP server and SCTP client, without having Diameter Address. The Diameter CER, DWR, DPR or forwarded.
Below is illustrated direction of establishing links and associations.

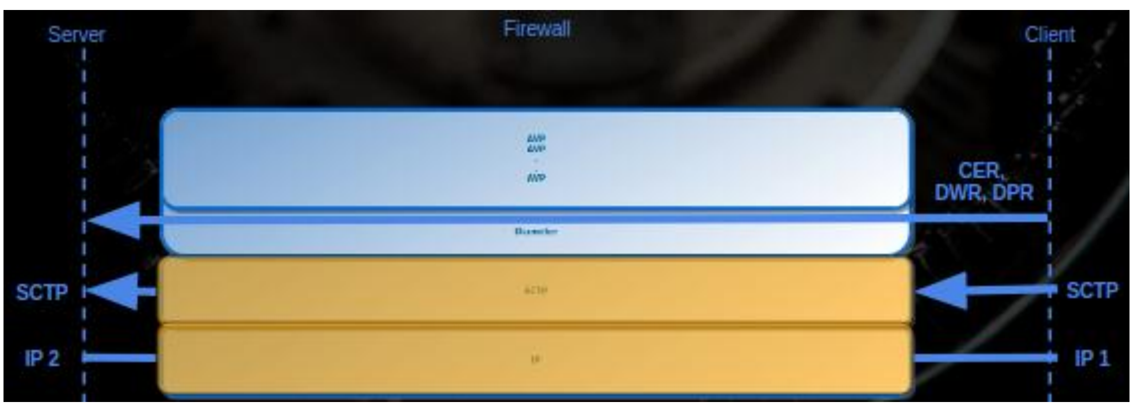

Figure 6.2.1b - Diameter Firewall connections

6.2.2 Deployment

Possible deployment can be loopback on DRA towards the FW. Also other deployment scenarios could be FW deployed directly on the link or FW just protecting single network element.

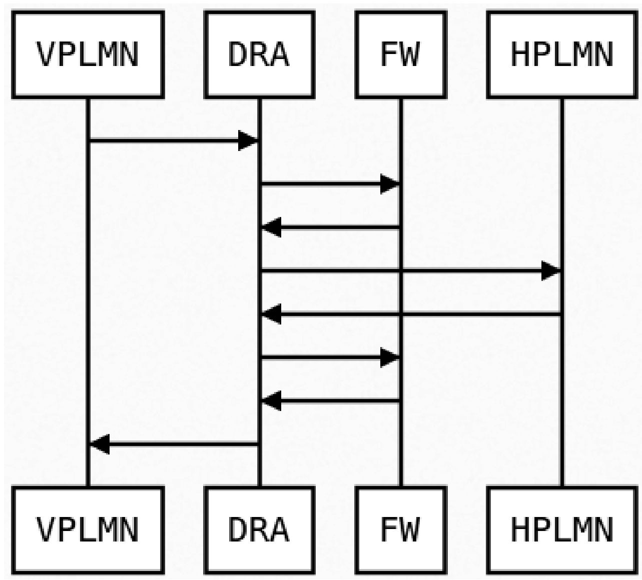

Figure 6.2.2 - Diameter Firewall deployment

6.2.3 Diameter Encryption Flow

The below figure illustrates the encryption flow. The principles are similar to SS7 FW, with the difference that the encryption is on AVP level in Diameter protocol.

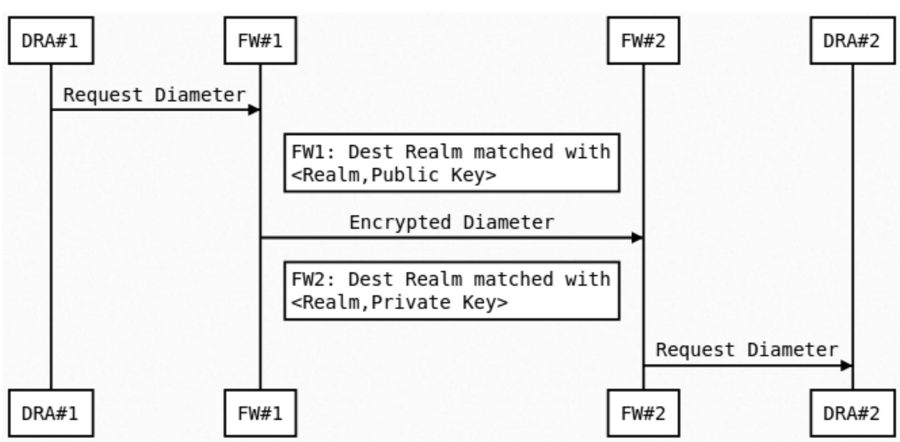

Figure 6.2.3 - Diameter Encryption Flow

6.2.4 Diameter Encryption Algorithm

1. Encrypted is the on the Diameter AVP level
2. Not encrypted AVPs are the AVPs required for IPX carriers (mainly host, realm, route)
3. Encrypted is the following payload for every AVP:
    a. version (4 bytes)
    b. encrypted(timestamp (4 bytes) + avp_bytes)   // If the key is short the multiple similar blocks are created
4. Encryption algorithm should be mapped with version. Currently in the code only RSA/ECB/PKCS1Padding is used
5. Timestamp is verified after decryption to prevent replay attacks

6.2.5 Diameter Encryption Example

```
147 62.930384208  127.0.0.1          127.0.0.1          DIAMET…   462 cmd=3GPP-Notify Request(323) flags=R--- appl=3GPP S6a/S6d(16777251) h2h=4a49277c e2e=6f500011 |
148 62.931295117  127.0.0.1          127.0.0.1          DIAMET…   426 SACK cmd=3GPP-Notify Answer(323) flags=--E- appl=3GPP S6a/S6d(16777251) h2h=4a49277c e2e=6f500011 …
151 62.939193161  127.0.0.1          127.0.0.1          DIAMET…  1334 SACK cmd=3GPP-Notify Answer(323) flags=--E- appl=3GPP S6a/S6d(16777251) h2h=4a49277c e2e=6f500011 …
155 62.957918437  127.0.0.1          127.0.0.1          DIAMET…   426 SACK cmd=3GPP-Notify Answer(323) flags=--E- appl=3GPP S6a/S6d(16777251) h2h=4a49277c e2e=6f500011 …
156 62.957935581  127.0.0.1          127.0.0.1          DIAMET…   410 SACK cmd=3GPP-Update-Location Request(316) flags=R--- appl=3GPP S6a/S6d(16777251) h2h=4a49277d e2e…
159 62.966246812  127.0.0.1          127.0.0.1          DIAMET…  1514 SACK cmd=3GPP-Update-Location Request(316) flags=R--- appl=3GPP S6a/S6d(16777251) h2h=4a49277d e2e…
164 62.985854473  127.0.0.1          127.0.0.1          DIAMET…   410 SACK cmd=3GPP-Update-Location Request(316) flags=R--- appl=3GPP S6a/S6d(16777251) h2h=4a49277d e2e…
165 62.986540937  127.0.0.1          127.0.0.1          DIAMET…   350 SACK cmd=3GPP-Update-Location Answer(316) flags=--E- appl=3GPP S6a/S6d(16777251) h2h=4a49277d e2e=…
168 62.992970061  127.0.0.1          127.0.0.1          DIAMET…  1418 SACK cmd=3GPP-Update-Location Answer(316) flags=--E- appl=3GPP S6a/S6d(16777251) h2h=4a49277d e2e=…
173 63.009762391  127.0.0.1          127.0.0.1          DIAMET…   350 SACK cmd=3GPP-Update-Location Answer(316) flags=--E- appl=3GPP S6a/S6d(16777251) h2h=4a49277d e2e=…
186 92.995232305  127.0.0.1          127.0.0.1          DIAMET…   142 cmd=Device-Watchdog Request(280) flags=R--- appl=Diameter Common Messages(0) h2h=4a492fc2 e2e=70b0…
187 92.996785046  127.0.0.1          127.0.0.1          DIAMET…   142 cmd=Device-Watchdog Request(280) flags=R--- appl=Diameter Common Messages(0) h2h=4a492fc2 e2e=70b0…
188 92.998244255  127.0.0.1          127.0.0.1          DIAMET…   142 cmd=Device-Watchdog Request(280) flags=R--- appl=Diameter Common Messages(0) h2h=4a492fc2 e2e=70b0…
189 92.999627596  127.0.0.1          127.0.0.1          DIAMET…   166 SACK cmd=Device-Watchdog Answer(280) flags=---- appl=Diameter Common Messages(0) h2h=4a492fc2 e2e=…
190 93.000873609  127.0.0.1          127.0.0.1          DIAMET…   166 SACK cmd=Device-Watchdog Answer(280) flags=---- appl=Diameter Common Messages(0) h2h=4a492fc2 e2e=…
191 93.002105486  127.0.0.1          127.0.0.1          DIAMET…   166 SACK cmd=Device-Watchdog Answer(280) flags=---- appl=Diameter Common Messages(0) h2h=4a492fc2 e2e=…

▸ Flags: 0x80, Request
  Command Code: 316 3GPP-Update-Location
  ApplicationId: 3GPP S6a/S6d (16777251)
  Hop-by-Hop Identifier: 0x4a49277d
  End-to-End Identifier: 0x6f500014
  [Answer In: 168]
▸ AVP: Session-Id(263) l=48 f=-M- val=CretedByDiameterLiveClient;1493747508867
▸ AVP: Unknown(1100) l=136 f=--- val=45e945a4a0758023a778a26b851e619db9c671e851e34178...
▸ AVP: Destination-Host(293) l=28 f=-M- val=aaa://127.0.0.1:3868
▸ AVP: Unknown(1100) l=136 f=--- val=3650b1097190e791b8d03f53f9eb1afffbdc34cc5afdb80f...
▸ AVP: Origin-Host(264) l=59 f=-M- val=[redacted]
▸ AVP: Unknown(1100) l=136 f=--- val=79d158db1690d26781fffbcbf040ec7023dabc728bf5556a...
▸ AVP: Unknown(1100) l=136 f=--- val=035f530ec93c18f991225cf91b05cc5dd4e167cb6d4c463b...
▸ AVP: Unknown(1100) l=136 f=--- val=013d585de64a1ef1a068386d375827cb2de27e1c720dbf51...
▸ AVP: Unknown(1100) l=136 f=--- val=5a1f8a2df193160d5fe39c3231630e09c2447ff00b879fd9...
▸ AVP: Unknown(1100) l=136 f=--- val=6510ea0a4bbd1c8ed21c50fee483110cb141c2f58f3d1a98...
▸ AVP: Destination-Realm(283) l=28 f=-M- val=exchange.example.org
▸ AVP: Origin-Realm(296) l=34 f=-M- val=exchangeClient.example.org
▸ AVP: Unknown(1100) l=264 f=--- val=7290d5360b1fcecbf95fe43055cb5f49ef273f3ce8a43b04...
```

Figure 6.2.5 - Diameter Encryption Example

6.2.6 Diameter Encryption Autodiscovery

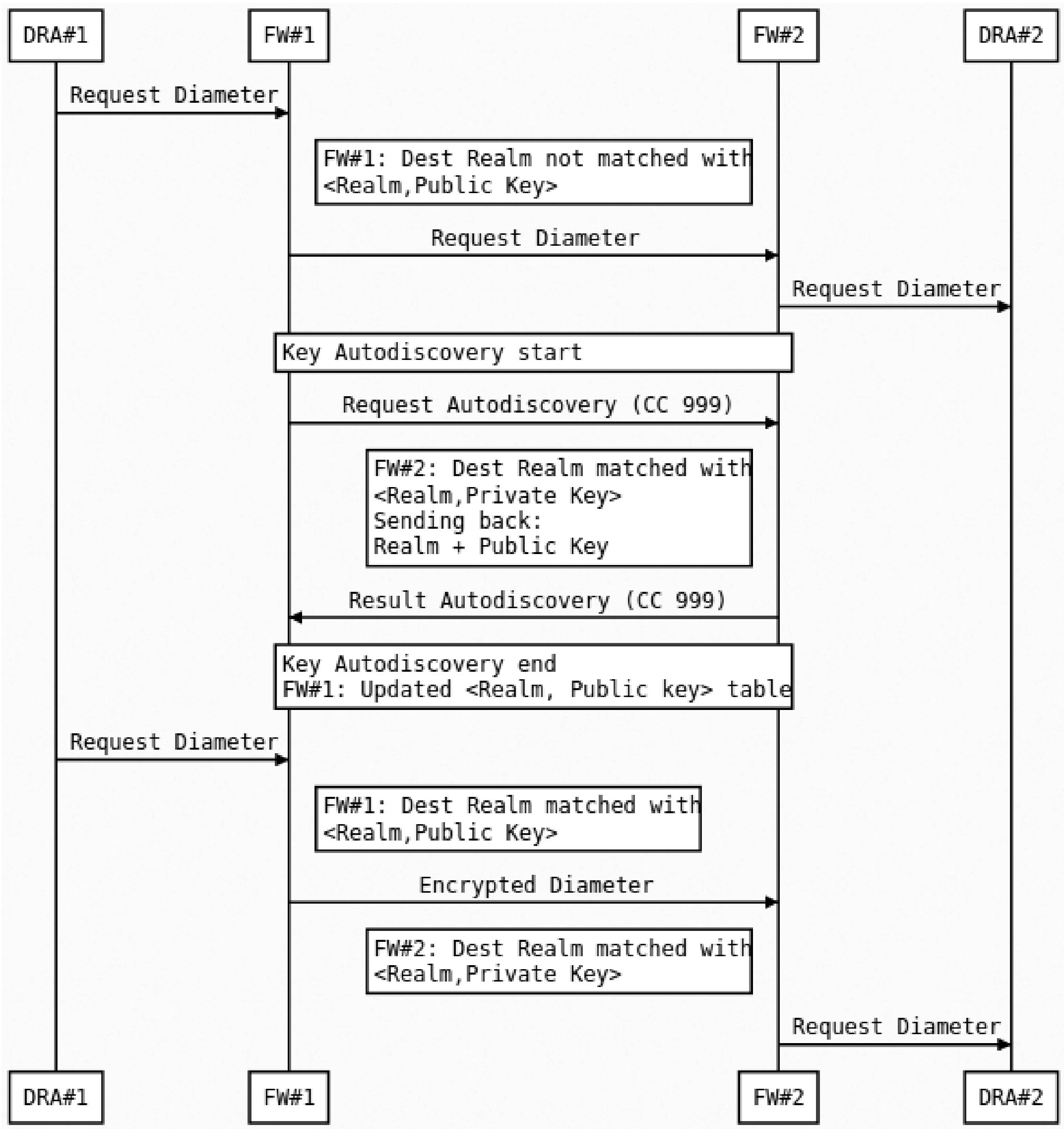

Figure 6.2.6 - Diameter Encryption Flow

6.2.7 Diameter Signature Algorithm

1. Only Diameter Requests are signed
2. Check if the Diameter message already contains some Diameter signature AVP. If not, sign it.
3. Diameter signature is Octet String of the following:
    a. version (4 bytes)
    b. timestamp (4 bytes)

   c. signature
4. Signature is calculated:
   a. String dataToSign = getApplicationId + ":" + CommandCode + ":" + EndToEndIdentifier + ":" + timestamp + diameter_layer;
   b. String diameter_layer = SORT_STRINGS(base64(avp_1) + … + base64(avp_N)); // for AVP != RECORD_ROUTE
   c. String dataToSign is then hashed (currently in code SHA256WithRSA is used)

6.2.8 Diameter Signature

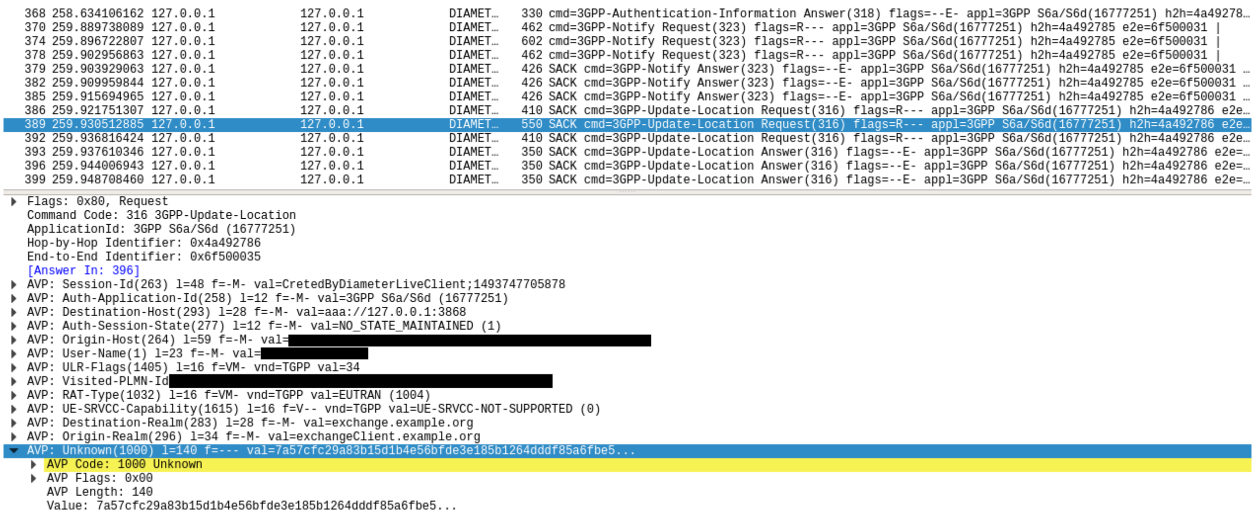

Figure 6.2.8 - Diameter Signature Example

7 Closing remarks

The currently released version of the SigFW should be understood as a research project/reference implementation and not as operational ready solution. The work as well as the filtering capabilities and the confidentiality/integrity protection schemes should be evolved further to find solution which is addressing both operational and security needs.

By this open-source approach we hope we can help to improve the SS7/Diameter security and this project adoption can also help to reveal the source and origin of these SS7/Diameter attacks. The SS7/Diameter security is affecting all mobile users worldwide. We believe that the open source is the right way for the security and should be adopted also in telecom field.

As it is seen, the current work has been created thanks to Telestax open-source signalling stack and Wireshark, Elastic projects.

7.1 VM architecture

VM is available for download at https://github.com/P1sec/SigFW/wiki/VM

Ubuntu Server
    eth0 management
    eth1 signalling (possible to configure the firewall here)
    eth2 passive signalling (used by tshark to feed the VM in passive mode)
Installed ElasticSearch, Kibana
All firewall modules as systemd services
On localhost running SS7ClientLiveInput - > SS7Firewall -> SS7Server
pcap -> tshark -> SS7ClientLiveInput
eth2 -> tshark -> SS7ClientLiveInput
eth2 -> tshark -> curl -> ElasticSearch -> Kibana

7.2 SigFW use cases

The below figures illustrates high-level use cases of the SigFW. The figures outlines the use of SigFW for standard filtering capabilities, the confidentiality and integrity protection of the signalling and also the DNAT towards the honeypot.

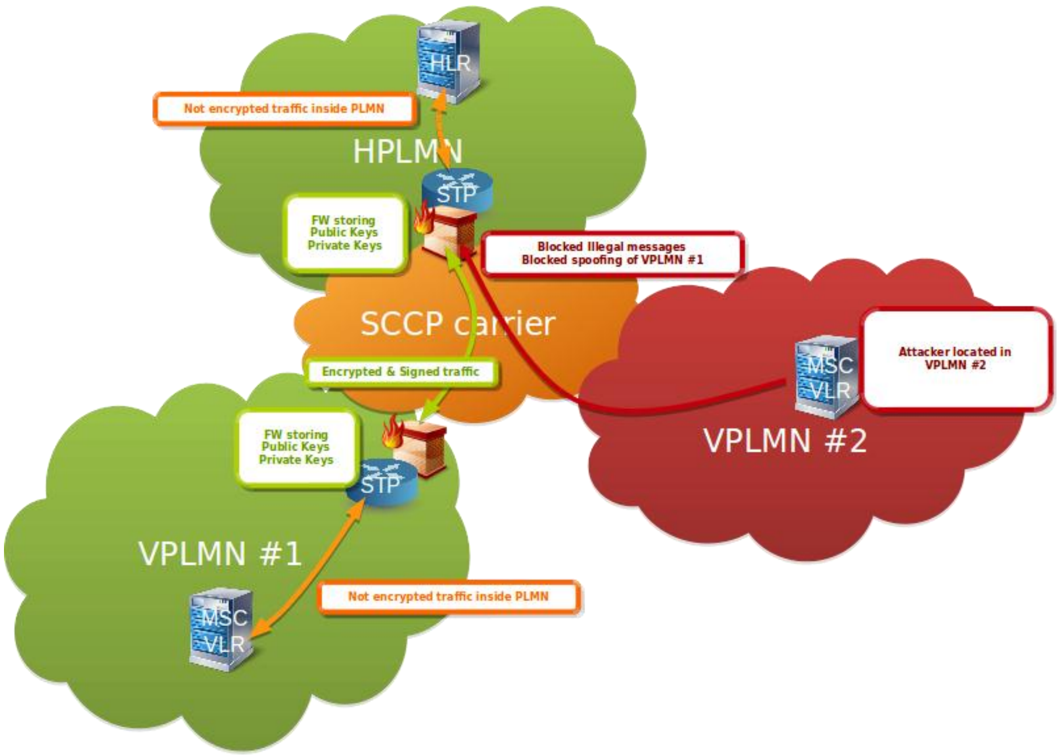

Figure 7.2a - SigFW filtering and confidentiality and integrity protection of signalling

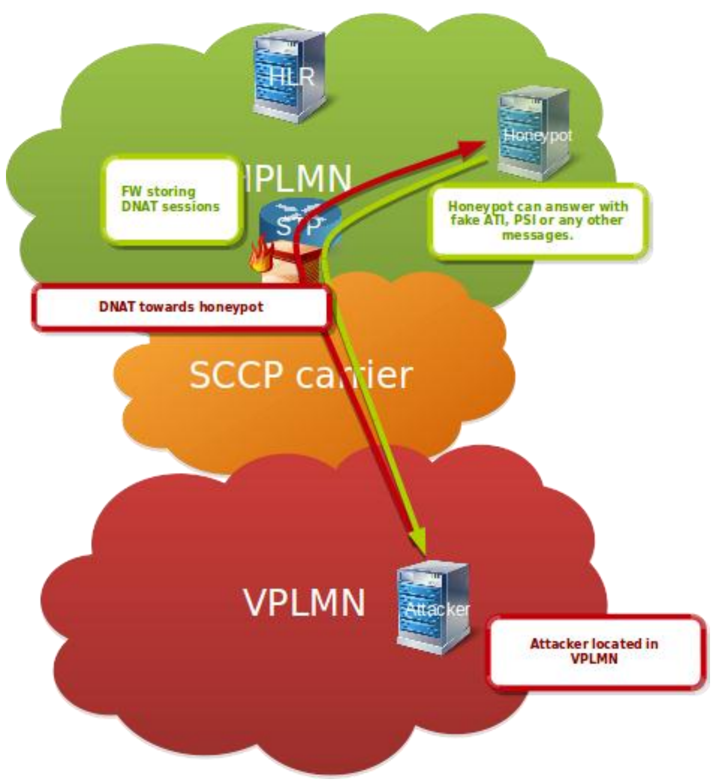

Figure 7.2b - SigFW forwarding the attacker to honeypot

8 Related Open Source Contribution

8.1 Tshark to Elasticsearch export and security monitoring with Kibana

We would like also to highlight the contributed patch to Wireshark project. This features are used in the SigFW VM.

Wireshark is capable to export decoded packets in json format. Additionally the tshark can export json format and also elasticsearch json which can be directly imported into elasticsearch cluster.

This could enable to use tshark as signalling probe and perform signalling monitoring as illustrated on the following figure.

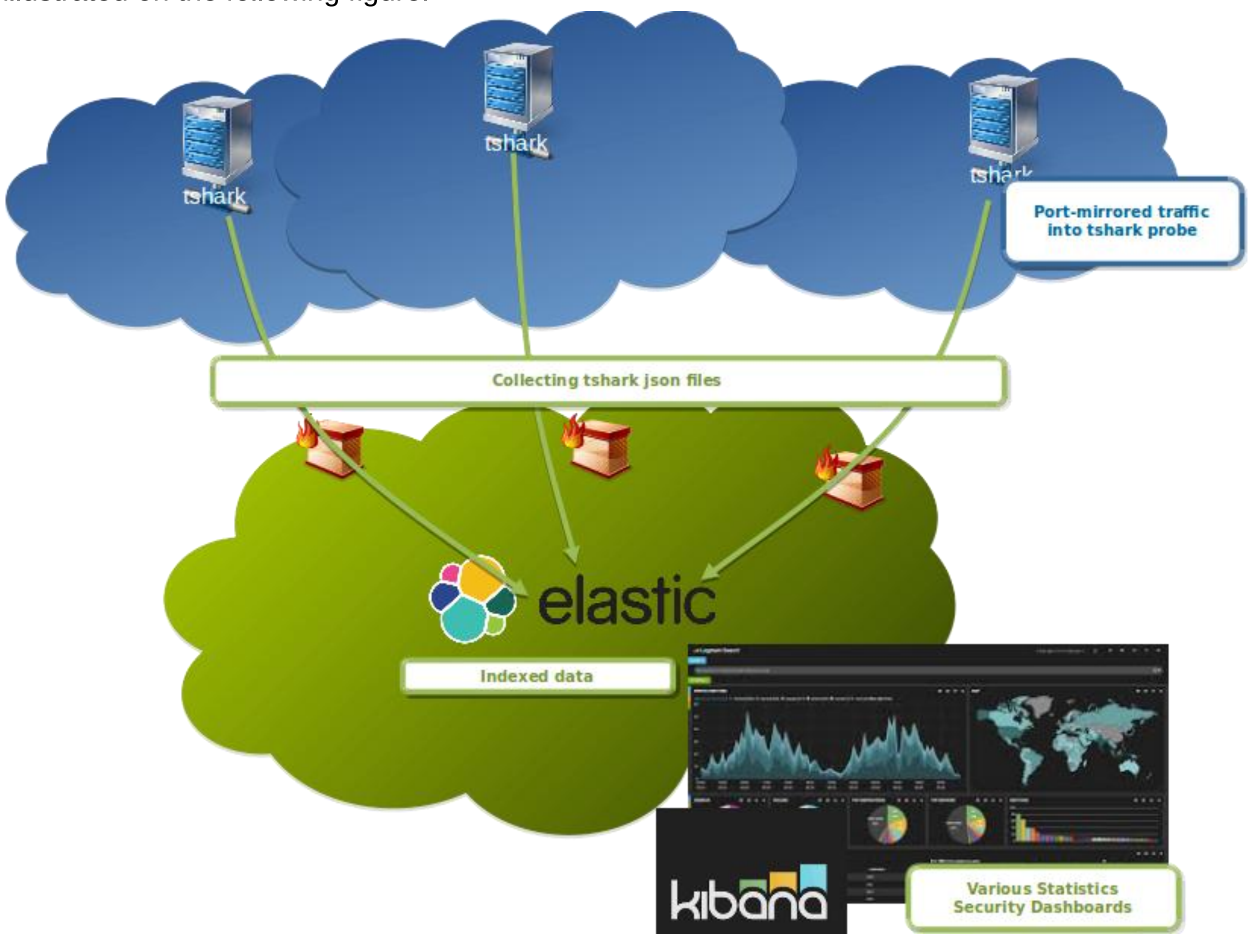

Figure 8.1a - tshark with Elasticsearch

The monitoring could be for network functionality or troubleshooting reasons but also could be used for security monitoring. The light solution could be just using Kibana dashboards for security monitoring.

The following figures illustrates signalling monitoring in Kibana and simple Dashboards.

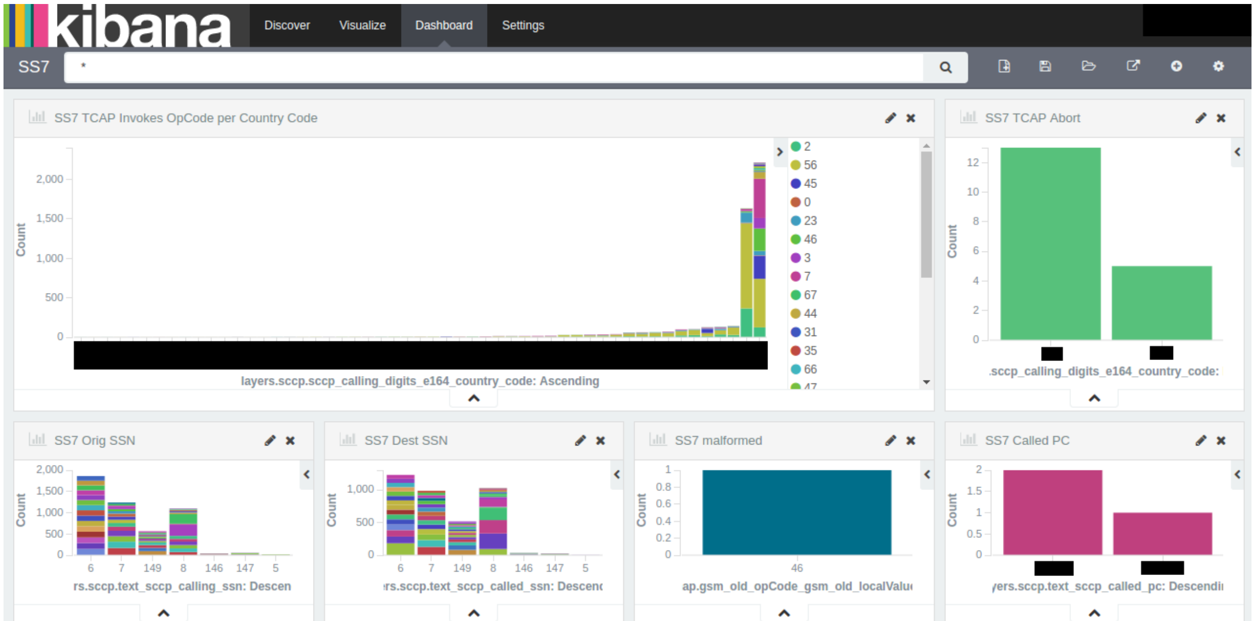

Figure 8.1b - tshark with Kibana example 1

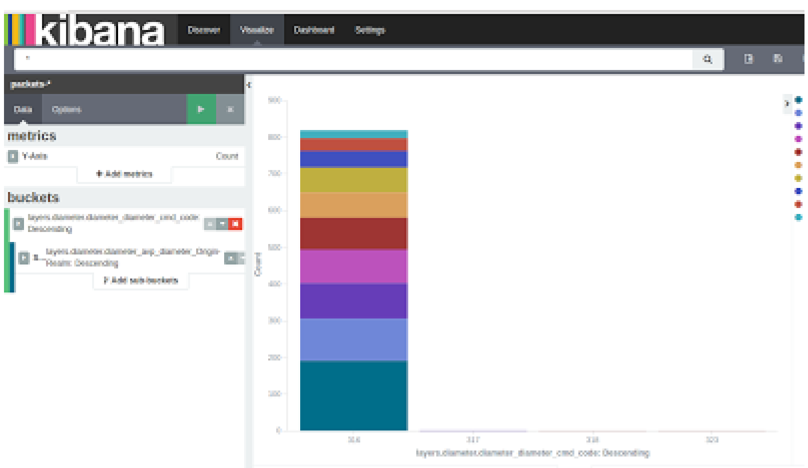

Figure 8.1c - tshark with Kibana example 2

More details are descibed on https://sites.google.com/site/h21lab/tools/tshark_elasticsearch.

9 References and Acknowledgement

[1] GSMA workgroup collaboration (FS.11, FS.19, FS.20 ...)
[2] 3GPP standardization on signaling (TS 29.002, TS 22.078, TS 29.204, TS 33.204, TS 29.272, TS 29.060, TS 29.274, ...)
[3] P1 Security SS7 & Diameter security deployment (http://www.p1sec.com)
[4] P1 Labs SS7map and security research (http://ss7map.p1sec.com/, http://labs.p1sec.com/)
[5] H21 lab blogs, published tools, research (https://sites.google.com/site/h21lab/)

International conferences presentations:
[6] SCTPscan - Finding entry points to SS7 Networks & Telecommunication Backbones, Philippe Langlois, Black Hat 2006
[7] Locating Mobile Phones using SS7, Tobias Engel, CCC 2009
[8] SCCP hacking, attacking the SS7 & SIGTRAN applications one step further and mapping the phone system, Philippe Langlois, CCC 2009
[9] SCCP hacking Attacking the SS7 & SIGTRAN and Mapping the Phone System, Philippe Langlois, 2010
[10] Getting in the SS7 kingdom: hard technology and disturbingly easy hacks to get entry points in the walled garden, Philippe Langlois, Hackito Ergo Sum 2010
[11] Hack In The Box 2012: A 15 Year Perspective on Why Telcos Keep Getting Hacked, Philippe Langlois, Emmanuel Gadaix, Hack In The Box 2012
[12] Worldwide attacks on SS7/SIGTRAN network, Pierre-Olivier Vauboin, Alexandre De Oliveira, P1 Security, Hackito Ergo Sum 2014
[13] Mobile self--defense, Karsten Nohl, SR Labs, CCC 2014
[14] Securing the SS7 Interconnect Tobias Engel, Troopers 2015
[15] SS7: Locate. Track. Manipulate, Tobias Engel, CCC 2015
[16] About SS7 (Signalling System Seven) in 60 Minutes, SR Labs, 2016

10 Annex

10.1 SS7FW VM readme

```
Signalling firewall and monitoring appliance

Interfaces:
   enp0s3 - management (SSH, Web)
   enp0s8 - signalling (SS7FW could be reconfigured here)
   enp0s9 - passive signalling (port-mirrored traffic)

To access Kibana:
   http://<host>:5601/

To access API
   https://<host>:8443/ss7fw_api/1.0/get_status

To check if services are running:
   sudo service tshark_to_ss7fw status
   sudo service tshark_to_ek status
   sudo service ss7fw status
   sudo service ss7server status
   sudo service ss7client status

To replay the pcap on passive interface:
   sudo tcpreplay  --intf1=enp0s9 sigtran.pcap

Description:
   By default the SS7FW is in passive mode.
   Tshark is capturing traffic on enp0s9 and pushing into ElasticSearch.
   Second instance of tshark is pushing capture into named pipe of SS7FW.
   The SS7FW consist of ss7client, ss7firewall, ss7server. ss7client replay
   the captured traffic from enp0s9 towards ss7firewall and ss7server on
   localhost.

   SS7FW is located in /opt/ss7fw/

   Before first run or if the IP has changed, modify /etc/kibana/kibana.yml"


To access logs:
   tail -f /opt/ss7fw/ss7fw/ss7fw.ss7fw-core_jar_1.0.0-SNAPSHOT/ss7fw.log
```

10.2 SS7FW configuration example

```
{
        "operator_configuration": {
                "Home_GT_prefixes_comment": "# Identification of HPLMN network, used to identify incoming and outgoing
traffic of HPLMN",
                "Home_GT_prefixes": [
                        "0"
                ],
                "Home_IMSI_prefixes_comment": "# Identification Home IMSI range for HPLMN network, used to identify home
subscribers",
```



```
                    "Home_IMSI_prefixes": [
                              "111111"
                    ]
          },
          "sigfw_configuration": {
                    "ss7fw_configuration_comment": "# Signalling Firewall configuration. Because of dynamic updates, the
sigfw.json.last is periodically created on filesystem.",

                    "sctp_comment": "# SCTP configuration part of Signalling Firewall",
                    "sctp": {
                              "sctp_management_name": "sctp_mgmt",
                              "sctp_server": [
                                        {
                                                  "server_name": "sctp_server",
                                                  "host_address": "127.0.0.1",
                                                  "port": "3433"
                                        }
                              ],
                              "sctp_server_association": [
                                        {
                                                  "peer_address": "127.0.0.1",
                                                  "peer_port": "2345",
                                                  "server_name": "sctp_server",
                                                  "assoc_name": "sctp_from_client_to_firewall"
                                        }
                              ],
                              "sctp_association": [
                                        {
                                                  "host_address": "127.0.0.1",
                                                  "host_port": "2344",
                                                  "peer_address": "127.0.0.1",
                                                  "peer_port": "3434",
                                                  "assoc_name": "sctp_from_firewall_to_server"
                                        }
                              ]
                    },

                    "m3ua": {
                              "m3ua_comment": "# M3UA configuration part of Signalling Firewall",

                              "m3ua_server": {
                                        "m3ua_management_name": "m3ua_server_mgmt",
                                        "as_name": "RAS1",
                                        "asp_name": "RASP1",
                                        "sctp_assoc_name": "sctp_from_client_to_firewall",
                                        "remote_pc": ["1"]
                              },
                              "m3ua_client": {
                                        "m3ua_management_name": "m3ua_client_mgmt",
                                        "as_name": "AS1",
                                        "asp_name": "ASP1",
                                        "sctp_assoc_name": "sctp_from_firewall_to_server",
                                        "remote_pc": ["2"]
                              }
                    },

                    "firewall_rules": {
                              "firewall_rules_comment": "# Firewall filtering rules configuration",

                              "firewall_policy_comment": "# Allowed value is one from: DROP_SILENTLY, DROP_WITH_SCCP_ERROR,
DNAT_TO_HONEYPOT, ALLOW",
                              "firewall_policy": "DROP_WITH_SCCP_ERROR",

                              "sccp": {
                                        "sccp_comment": "# SCCP firewall rules",
                                        "calling_gt_whitelist": [
                                                  "4*"
                                        ],
                                        "calling_gt_blacklist": [
                                                  "10000000000",
                                                  "222*"
                                        ]
                              },

                              "tcap": {
                                        "tcap_comment": "# TCAP Cat1 firewall rules",
                                        "oc_blacklist": [
                                                  "5",
                                                  "6",
                                                  "9",
                                                  "16",
                                                  "20",
                                                  "21",
```



```
                    "22",
                    "24",
                    "25",
                    "26",
                    "27",
                    "28",
                    "29",
                    "30",
                    "31",
                    "32",
                    "33",
                    "34",
                    "35",
                    "39",
                    "40",
                    "41",
                    "42",
                    "43",
                    "50",
                    "51",
                    "52",
                    "55",
                    "58",
                    "62",
                    "65",
                    "68",
                    "69",
                    "71",
                    "72",
                    "76",
                    "77",
                    "78",
                    "79",
                    "80",
                    "81",
                    "82",
                    "83",
                    "84",
                    "85",
                    "86",
                    "109",
                    "110",
                    "111",
                    "112",
                    "113",
                    "114",
                    "115",
                    "116",
                    "117",
                    "118",
                    "119",
                    "120",
                    "121",
                    "122",
                    "123",
                    "124",
                    "125",
                    "126"
          ]
     },

     "map": {
          "map_comment": "# MAP Cat2 firewall rules",
          "cat2_oc_blacklist": [
                    "3",
                    "4",
                    "7",
                    "8",
                    "70"
          ]
     },

     "lua": {
          "lua_comment": "# LUA Blacklist firewall rules. Currently supported LUA variables are:
sccp_calling_gt, sccp_called_gt, tcap_oc, tcap_ac, tcap_tag, map_imsi, map_msisdn",
          "blacklist_rules": [
                    "sccp_called_gt == '22222222222'",
                    "sccp_calling_gt == '11111111111' and tcap_oc == '59'"
          ]
     },

     "ids": {
```

```
                                        "ids_comment": "# IDS API. After evaluating internal firewall rules, the external IDS
system can be used to check message (e.g. Cat3). If not required remove this ids json block from config.",

                                        "ids_api_type_comment": "# Type of connector. Currently supported only REST",
                                        "ids_api_type": "REST",
                                        "ids_servers": [
                                                  {
                                                            "host":
"https://localhost:8443/ss7fw_api/1.0/eval_sccp_message_in_ids",
                                                            "username": "user",
                                                            "password": "password"
                                                  }
                                        ]
                              },

                              "mthreat": {
                                        "mthreat_comment": "# mThreat API. If the message matches internal firewall or IDS rules,
then the firewall can report the event in anonymized way to mThreat. If not required remove this mthreat json block from
config.",

                                        "mthreat_api_type_comment": "# Type of connector. Currently supported only REST",
                                        "mthreat_api_type": "REST",
                                        "mthreat_salt_comment": "# Change the salt value for unique anonymization",
                                        "mthreat_salt": "XVm4AoKrkicsgEcx",
                                        "mthreat_servers": [
                                                  {
                                                            "host":
"https://51.15.148.211:8444/mthreat_api/1.0/send_ss7_alert_to_mthreat",
                                                            "username": "contact@p1sec.com",
                                                            "password": "contact@p1sec.com"
                                                  }
                                        ]
                              },

                              "honeypot": {
                                        "honeypot_comment": "# Honeypot configuration. Only used if firewall policy is
DNAT_TO_HONEYPOT",

                                        "sccp_gt_comment": "# The firewall after detecting the message will perform DNAT to the
following GT.",
                                        "sccp_gt": "33333333333",

                                        "dnat_session_expiration_timeout_comment": "# After matching the firewall or IDS rules,
the firewall will apply DNAT for calling GT for the defined number of seconds",
                                        "dnat_session_expiration_timeout": "30"
                              }
                    },

                    "encryption_rules": {
                              "encryption_rules_comment": "# TCAP encryption. NTP synchronization of FW instance is required to
work this properly. If autodiscovery is enabled the public keys are added dynamically. Public and private keys are Base64
encoded.",

                              "called_gt_encryption_comment": "# Should include json block with {called_gt, public_key}. For
example of config see sigfw_1.json or sigfw_2.json.",
                              "called_gt_encryption": [
                              ],
                              "called_gt_decryption_comment": "# Should include json block with {called_gt, public_key, private}.
For example of config see sigfw_1.json or sigfw_2.json.",
                              "called_gt_decryption": [
                              ],
                              "autodiscovery_comment": "# When enabled the Firewall will try to retrieve public key for unknown
destinations by sending MAP Invoke with OpCode 99.",
                              "autodiscovery": "true"
                    },

                    "signature_rules": {
                              "signature_rules_comment": "# TCAP signing. NTP synchronization of FW instance is required to work
this properly. Public and private keys are Base64 encoded.",

                              "calling_gt_verify_comment": "# Should include json block with {calling_gt, public_key}. For
example of config see sigfw_1.json or sigfw_2.json.",
                              "calling_gt_verify": [
                              ],
                              "calling_gt_signing_comment": "# Should include json block with {calling_gt, public_key,
private_key}. For example of config see sigfw_1.json or sigfw_2.json.",
                              "calling_gt_signing": [
                              ]
                    }
          }
}
```


10.3 DiameterFW configuration example

```
{
	"operator_configuration": {

		"Home_IMSI_prefixes_comment": "# Identification Home IMSI range for HPLMN network, used to identify home 
subscribers",
		"Home_IMSI_prefixes": [
			"111111"
		],
		"Home_Diameter_Realm_list_comment": "Operator Diameter Internal Realm list, used to identify incoming and 
outgoing traffic of HPLMN",
		"Home_Diameter_Realm_list": [
			"exchange.example.org"
		]
	},
	"sigfw_configuration": {
		"sctp": {
			"sctp_management_name": "sctp_mgmt",
			"sctp_server": [
				{
					"server_name": "sctp_server",
					"host_address": "127.0.0.1",
					"port": "3869"
				}
			],
			"sctp_server_association": [
				{
					"peer_address": "127.0.0.1",
					"peer_port": "13868",
					"server_name": "sctp_server",
					"assoc_name": "sctp_from_client_to_firewall"
				}
			],
			"sctp_association": [
				{
					"host_address": "127.0.0.1",
					"host_port": "13869",
					"peer_address": "127.0.0.1",
					"peer_port": "3868",
					"assoc_name": "sctp_from_firewall_to_server"
				}
			]
		},
		"firewall_rules": {
			"firewall_rules_comment": "# Firewall filtering rules configuration",

			"firewall_policy_comment": "# Allowed value is one from: DROP_SILENTLY, DROP_WITH_DIAMETER_ERROR, 
DNAT_TO_HONEYPOT, ALLOW",
			"firewall_policy": "DNAT_TO_HONEYPOT",

			"diameter": {
				"origin_realm_blacklist": [
					"blacklisted.example.org"
				],
				"application_id_whitelist": [
					"0",
					"16777251"
				],
				"command_code_blacklist": [
					"8388620",
					"8388622"
				],
				"cat2_command_code_blacklist": [
					"317",
					"319",
					"329"
				]
			},

			"lua": {
				"lua_comment": "# LUA Blacklist firewall rules. Currently supported LUA variables are: 
diameter_orig_host, diameter_orig_realm, diameter_dest_host, diameter_dest_realm, diameter_cc, diameter_ai, diameter_imsi, 
diameter_msisdn",
				"blacklist_rules": [
					"diameter_orig_realm == 'exchangeClient.example.org'",
					"diameter_orig_realm == 'exchangeClientB.example.org'"
				]
			},

			"ids": {
```

```
					"ids_comment": "# IDS API. After evaluating internal firewall rules, the external IDS
system can be used to check message (e.g. Cat3). If not required remove this ids json block from config.",

					"ids_api_type_comment": "# Type of connector. Currently supported only REST",
					"ids_api_type": "REST",
					"ids_servers": [
						{
							"host":
"https://localhost:8443/diameterfw_api/1.0/eval_diameter_message_in_ids",
							"username": "user",
							"password": "password"
						}
					]
				},

				"mthreat": {
					"mthreat_comment": "# mThreat API. If the message matches internal firewall or IDS rules,
then the firewall can report the event in anonymized way to mThreat. If not required remove this mthreat json block from
config.",

					"mthreat_api_type_comment": "# Type of connector. Currently supported only REST",
					"mthreat_api_type": "REST",
					"mthreat_salt_comment": "# Change the salt value for unique anonymization",
					"mthreat_salt": "XVm4AoKrkicsgEcx",
					"mthreat_servers": [
						{
							"host":
"https://51.15.148.211:8444/mthreat_api/1.0/send_diameter_alert_to_mthreat",
							"username": "contact@p1sec.com",
							"password": "contact@p1sec.com"
						}
					]
				},

				"honeypot": {
					"honeypot_comment": "# Honeypot configuration. Only used if firewall policy is
DNAT_TO_HONEYPOT",

					"diameter_host_comment": "# The firewall after detecting the message will perform DNAT to
the following Diameter address.",
					"diameter_host": "127.0.0.1",
					"diameter_realm": "honeypot.example.org",

					"dnat_session_expiration_timeout_comment": "# After matching the firewall or IDS rules,
the firewall will apply DNAT for Diameter address for the defined number of seconds",
					"dnat_session_expiration_timeout": "30"
				}
			},
			"encryption_rules": {
				"destination_realm_encryption": [
				],
				"destination_realm_decryption": [
				],
				"autodiscovery": "true"
			},
			"signature_rules": {
				"origin_realm_verify": [
				],
				"origin_realm_signing": [
				]
			}
		}
}
```


10.4 SS7FW API specification

10.4.1 Provisioning FW rules API

Title	sccp_calling_gt_blacklist_add
URL	Configurable
Method	GET
URL Params	**Reguired** @MatrixParam("gt") String gt
Data Params	**gt** String GlobalTitle
Success Response	SS7FW should return String "Successful"
Error Response	SS7FW should return not specified string
Sample Call	`https://XXX.XXX.XXX.XXX:8443/ss7fw_api/1.0/sccp_calling_gt_blacklist_add;gt=11111*`

Notes	`@GET`
	`@Consumes("text/plain")`
	`@Produces("text/plain")`
	`@Path("sccp_calling_gt_blacklist_add")`
	`public String sccp_calling_gt_blacklist_add(@MatrixParam("gt") String gt);`

Title	sccp_calling_gt_blacklist_remove
URL	Configurable
Method	GET
URL Params	**Reguired** @MatrixParam("gt") String gt
Data Params	**gt** String GlobalTitle
Success Response	SS7FW should return String "Successful"
Error Response	SS7FW should return not specified string
Sample Call	`https://XXX.XXX.XXX.XXX:8443/ss7fw_api/1.0/sccp_calling_gt_blacklist_remove;gt=11111*`

Notes	`@GET`
	`@Consumes("text/plain")`
	`@Produces("text/plain")`

	`@Path("sccp_calling_gt_blacklist_remove")` `public String sccp_calling_gt_blacklist_remove(@MatrixParam("gt") String gt);`

Title	sccp_calling_gt_blacklist_list
URL	Configurable
Method	GET
URL Params	
Data Params	

Success Response	SS7FW should return String containing the SCCP GTs
	
	Example:
	`10000000000`
	`222*`

Error Response	SS7FW should return empty string
Sample Call	`https://XXX.XXX.XXX.XXX:8443/ss7fw_api/1.0/sccp_calling_gt_blacklist_list`

Notes	`@GET`
	`@Consumes("text/plain")`
	`@Produces("text/plain")`
	`@Path("sccp_calling_gt_blacklist_list")`
	`public String sccp_calling_gt_blacklist_list();`

Title	tcap_oc_blacklist_add
URL	Configurable
Method	GET
URL Params	**Reguired** @MatrixParam("oc") int oc
Data Params	**oc**

	String OpCode
Success Response	SS7FW should return String "Successful"
Error Response	SS7FW should return not specified string
Sample Call	`https://XXX.XXX.XXX.XXX:8443/ss7fw_api/1.0/tcap_oc_blacklist_add;oc=71`

Notes	`@GET`
	`@Consumes("text/plain")`
	`@Produces("text/plain")`
	`@Path("tcap_oc_blacklist_add")`
	`public String tcap_oc_blacklist_add(@MatrixParam("oc") int oc);`

Title	tcap_oc_blacklist_remove
URL	Configurable
Method	GET
URL Params	**Reguired** @MatrixParam("oc") int oc
Data Params	**oc** String OpCode
Success Response	SS7FW should return String "Successful"
Error Response	SS7FW should return not specified string
Sample Call	`https://XXX.XXX.XXX.XXX:8443/ss7fw_api/1.0/tcap_oc_blacklist_remove;oc=71`

Notes	`@GET`
	`@Consumes("text/plain")`
	`@Produces("text/plain")`
	`@Path("tcap_oc_blacklist_remove")`
	`public String tcap_oc_blacklist_remove(@MatrixParam("oc") int oc);`

Title	tcap_oc_blacklist_list
URL	Configurable
Method	GET

URL Params	
Data Params	

Success Response	SS7FW should return String containing the OCs
	
	Example:
	`109`
	`110`
	`111`
	`112`

Error Response	SS7FW should return empty string
Sample Call	`https://XXX.XXX.XXX.XXX:8443/ss7fw_api/1.0/tcap_oc_blacklist_list`

Notes	`@GET`
	`@Consumes("text/plain")`
	`@Produces("text/plain")`
	`@Path("tcap_oc_blacklist_list")`
	`public String tcap_oc_blacklist_list();`

Title	map_cat2_oc_blacklist_add
URL	Configurable
Method	GET
URL Params	**Reguired** @MatrixParam("oc") int oc
Data Params	**oc** String OpCode
Success Response	SS7FW should return String "Successful"
Error Response	SS7FW should return not specified string
Sample Call	`https://XXX.XXX.XXX.XXX:8443/ss7fw_api/1.0/map_cat2_oc_blacklist_ad` `d;oc=3`

Notes	`@GET`
	`@Consumes("text/plain")`
	`@Produces("text/plain")`
	`@Path("map_cat2_oc_blacklist_add")`
	`public String map_cat2_oc_blacklist_add(@MatrixParam("oc") int oc);`

Title	map_cat2_oc_blacklist_remove
URL	Configurable
Method	GET
URL Params	**Reguired** @MatrixParam("oc") int oc
Data Params	**oc** String OpCode
Success Response	SS7FW should return String "Successful"
Error Response	SS7FW should return not specified string
Sample Call	`https://XXX.XXX.XXX.XXX:8443/ss7fw_api/1.0/map_cat2_oc_blacklist_remove;oc=3`

Notes	`@GET`
	`@Consumes("text/plain")`
	`@Produces("text/plain")`
	`@Path("map_cat2_oc_blacklist_remove")`
	`public String map_cat2_oc_blacklist_remove(@MatrixParam("oc") int oc);`

Title	map_cat2_oc_blacklist_list
URL	Configurable
Method	GET
URL Params	
Data Params	
Success Response	SS7FW should return String containing the OCs Example: 3 4 7
Error Response	SS7FW should return empty string
Sample Call	`https://XXX.XXX.XXX.XXX:8443/ss7fw_api/1.0/map_cat2_oc_blacklist_list`

Notes	`@GET`
	`@Consumes("text/plain")`
	`@Produces("text/plain")`
	`@Path("map_cat2_oc_blacklist_list")`
	`public String map_cat2_oc_blacklist_list();`

Title	map_cat2_oc_blacklist_list
URL	Configurable
Method	GET
URL Params	
Data Params	

Success Response	SS7FW should return String containing the SS7FW status
	
	Example:
	`Jetty Server Status = STARTED`
	`Jetty Date = Date: Fri, 21 Apr 2017 07:41:24 GMT`
	`Jetty URI = https://127.0.1.1:8443/`
	
	`SCTP Associations`
	`Name = sctp_from_client_to_firewall`
	`Details = Association [name=sctp_from_client_to_firewall, associationType=SERVER, ipChannelType=SCTP, hostAddress=null, hostPort=0, peerAddress=127.0.0.1, peerPort=2345, serverName=sctp_server, extraHostAddress=[]]`
	`isStarted = true`
	`isConnected = true`
	`Name = sctp_from_firewall_to_server`
	`Details = Association [name=sctp_from_firewall_to_server, associationType=CLIENT, ipChannelType=SCTP, hostAddress=127.0.0.1, hostPort=2344, peerAddress=127.0.0.1, peerPort=3434, serverName=null, extraHostAddress=[]]`
	`isStarted = true`
	`isConnected = true`
	
	`SCTP Servers = {Server [name=sctp_server, started=true, hostAddress=127.0.0.1, hostPort=3433, ipChannelType=SCTP, acceptAnonymousConnections=false, maxConcurrentConnectionsCount=0, associations(anonymous does not included)=[sctp_from_client_to_firewall, ], extraHostAddress=[]]}`
	
	`M3UA Server`
	`Name = RAS1`
	`isConnected = true`
	`isUp = true`
	`M3UA Server Route = {1:-1:-1=org.mobicents.protocols.ss7.m3ua.impl.RouteAsImpl@36f8d985}`
	
	`M3UA Client`
	`Name = AS1`
	`isConnected = true`
	`isUp = true`
	`M3UA Client Route = {2:-1:-1=org.mobicents.protocols.ss7.m3ua.impl.RouteAsImpl@48e54e27}`
	
	`SCCP M3UA User Parts = {0=rsp=1 rsp-flag=0 mask=0 rsp-prohibited=false rsccp-prohibited=false, 1=rsp=2 rsp-flag=0 mask=0 rsp-prohibited=false rsccp-prohibited=false}`
	
	`OS statistics`
	`Available processors (cores): 8`
	`Free memory (bytes): 133897184`
	`Maximum memory (bytes): 3711959040`
	`Total memory available to JVM (bytes): 470286336`
	`File system root: /`
	`Total space (bytes): 53424500736`
	`Free space (bytes): 38042247168`

	`Usable space (bytes): 35304783872`
	
	`Network interfaces`
	`Display name: vboxnet0`
	`Name: vboxnet0`
	`InetAddress: /fe80:0:0:0:800:27ff:fe00:0%vboxnet0`
	`InetAddress: /192.168.56.1`
	`Display name: wlp5s0`
	`Name: wlp5s0`
	`InetAddress: /fe80:0:0:0:5c77:8239:7d6b:5b16%wlp5s0`
	`InetAddress: /192.168.1.62`
	`Display name: lo`
	`Name: lo`
	`InetAddress: /0:0:0:0:0:0:0:1%lo`
	`InetAddress: /127.0.0.1`

Error Response	SS7FW should return empty string or String with error message
Sample Call	`https://XXX.XXX.XXX.XXX:8443/ss7fw_api/1.0/get_status`

Notes	`@GET`
	`@Consumes("text/plain")`
	`@Produces("text/plain")`
	`@Path("get_status")`
	`public String get_status();`

10.4.2 Evaluation API

Title	eval_sccp_message_in_ids
URL	Configurable
Method	GET
URL Params	**Reguired** @MatrixParam("sccp_raw")
Data Params	**sccp_raw** The sccp_raw parameter is hex string of SCCP layer. (e.g. in tshark sccp_raw in json output)
Success Response	IDS should return String “1” on alert detection
Error Response	IDS should return String “0” if the message is not alert.
Sample Call	`https://XXX.XXX.XXX.XXX:8443/ss7fw_api/1.0/eval_sccp_message_in_ids` `;sccp_raw=aabbccddeeff`

Notes	`@GET`
	`@Consumes("text/plain")`
	`@Produces("text/plain")`

	`@Path("eval_sccp_message_in_ids")` `public String eval_sccp_message_in_ids(@MatrixParam("sccp_raw") String sccp_raw);`

10.4.3 mThreat API

Title	send_ss7_alert_to_mthreat
URL	Configurable
Method	POST
URL Params	**Reguired** String alert

Data Params	**String alert**
	The alert is JSON containing currently in API v1.0 the following variables. But in future the mThreat will support also more values and also Diameter should be supported.
	`String sccp_calling_gt = "";`
	`String sccp_called_gt = "";`
	`String tcap_oc = "";`
	`String tcap_ac = "";`
	`String map_imsi = "";`
	`String map_msisdn = "";`
	
	Example of JSON
	`{"map_msisdn":"0016883DA7B9FAFD9BD9BE3E7FD4171A5058E4E3","tcap_tag":"2","tcap_oc":"82","map_imsi":"0016883DA7B9FAFD9BD9BE3E7FD4171A5058E4E3","sccp_calling_gt":"11111111111","tcap_ac":"[0, 4, 0, 0, 1, 0, 1, 2]","sccp_called_gt":"00000000000"}`

Success Response	mThreat should return String "1"
Error Response	mThreat could return String "0" in case of some failure or other not specified string.
Sample Call	`https://XXX.XXX.XXX.XXX:8444/mthreat_api/1.0/send_ss7_alert_to_mthreat`

Notes	`@POST`
	`@Consumes("text/plain")`
	`@Produces("text/plain")`
	`@Path("send_ss7_alert_to_mthreat")`
	`public Response send_ss7_alert_to_mthreat(String alert)`