**3GPP TSG-SA3 Meeting #104-e *S3-212523***

**16 – 27 August 2021, Online**

**Source: CableLabs, Deutsche Telekom AG, InterDigital, Johns Hopkins University APL, US National Security Agency**

**Title: Short-lived public key-based solution for KI#2**

**Document for: Approval**

**Agenda Item: 5.1**

# 1 Decision/action requested

***It is requested to approve the pCR.***

# 2 References

|  |  |
| --- | --- |
| [1] | 3GPP TS 23.041, “Technical realization of Cell Broadcast Service (CBS)”.  |
| [2] | 3GPP TR 33.809, “Study on 5G Security Enhancement against False Base Stations.” |
| [3]  | 3GPP TR 33.969, "Study on security aspects of Public Warning System (PWS)". |
| [4] | IETF RFC 3820. Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile |
| [5] | IETF RFC 4082. “Timed Efficient Stream Loss-Tolerant Authentication (TESLA): Multicast Source Authentication Transform Introduction". |
| [6] | IETF Internet-Draft. Delegated Credentials for TLS. https://datatracker.ietf.org/doc/draft-ietf-tls-subcerts/ |
| [7] | Dan Boneh, Glenn Durfee, and Matt Franklin. "Lower bounds for multicast message authentication.", International Conference on the Theory and Applications of Cryptographic Techniques. Springer, Berlin, Heidelberg, 2001. |
| [8] | Syed Rafiul Hussain, Mitziu Echeverria, Ankush Singla, Omar Chowdhury, and Elisa Bertino. “Insecure Connection Bootstrapping in Cellular Networks: The Root of All Evil”, The 12th ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec), 2019.  |
| [9] | Zhenhua Li, et al. “FBS-Radar: Uncovering Fake Base Stations at Scale in the Wild”, NDSS 2017. |
| [10] | Qualcomm, “New WID on NR Repeaters”, RP-202927. December 2020.  |

# 3 Rationale

Several asymmetric key based solutions have been proposed to address key issue #2 by providing authenticity of the system information. However, there are still a few outstanding questions about those solutions that need further investigation, including:

* How are asymmetric keys and certificates managed, such as certificate revocation?
* How can replay attacks be mitigated?
* How shall UE behave in various scenarios when protected system information fails verification?

This pCR provides a comprehensive solution to address those outstanding issues.

# 4 Detailed proposal

## Solution 6.X - Short-lived asymmetric key-based solution for protecting system information

### 6.X.1 Introduction

#### 6.X.1.1 General

For key issue #2 (security protection of system information) in TR 33.809 [2], there are multiple proposals including solutions 7, 9, 11, 12, 14, 19, and 20. These solutions can be classified into three categories (see Table 1 for a summary),

* public key based digital signature (solutions 7, 11, 12, 20),
* symmetric key based message authentication codes (MAC) (solution 9), and
* message hash consistency check without digital signature or MAC (solutions 14, 19).



Table 6.X.1.1-1: Summary of the proposed solutions for KI#2

We next briefly discuss each category of the proposed solutions for KI#2.

#### 6.X.1.2 Hash-based consistency checks

Hash based consistency checks leverage existing NAS security context to detect spoofed or falsified broadcasting messages when UEs are in CONNECTED state. It provides certain security benefits without requiring any additional overhead in key management. However, it does not protect UEs in all states.

#### 6.X.1.3 Symmetric key based MAC

It is a common understanding that traditional symmetric key based MAC works well for two-party communication, or n-party communication when n is small, but not well for n-party communication when n is large and dynamic. This is because a symmetric key must be pre-shared among all group members [7], requiring a complex key distribution mechanism when the group is large, especially when the group membership is dynamic.

Mobile networks have a large number of base stations and UEs, and the relationship between UEs and base stations is dynamic, thus it is an n-party communication with n being large and dynamic. Therefore, traditional symmetric key based MAC does not appear suitable for authenticating broadcast messages in mobile networks.

Further, any member within a shared key group may be able to use the shared key to forge or tamper with messages unless the key usage is strictly limited to MAC verification only. Although USIM can be leveraged to protect the shared key so that it cannot be read out, it is not clear if it is possible for an attacker to use a USIM to generate a MAC using the shared key.

A non-traditional symmetric key based MAC, namely, TESLA [5], does not require a symmetric key to be pre-shared among group members. Rather, it discloses a key to all members. The key is not disclosed in the same message where its MAC is included. Rather, it is disclosed in a future message with a delay in time. Although TESLA nicely solves the problem of key distribution in a large and dynamic group, it requires other mechanisms such as digital signature to protect anchor keys that are used to authenticate regular keys used to generate MACs. Therefore, TESLA alone cannot solve the problem.

Editor’s Note: the use of TESLA to reduce computational overhead in digital signing and its potential quantum-resistance features is FFS.

#### 6.X.1.4 Asymmetric key based digital signatures

Asymmetric key or public key based digital signatures are naturally suited for authenticating broadcasting messages because they allow messages to be broadcast from one party and verified by many parties without the need to share the signing key. This is demonstrated by the proposed solutions (e.g., #7, #11, and #20) based on digital signature, which can offer protection to UEs in all states.

However, there are still some gaps in the existing solutions that require further improvement. Particularly, outstanding issues in key management, replay mitigation, and cell selection need to be resolved for an asymmetric key based solution to be both secure and practical.

This solution fills these gaps and can satisfy the security requirements of KI#2 while also addressing various challenges in the solution deployment.

We next perform comprehensive analysis of various design options and outline the rationale of our design choices in three main areas including authenticity of system information, replay mitigation, and UE cell selection strategy.

### 6.X.2 Solution details

#### 6.X.2.1 Authenticity of system information

##### 6.X.2.1.1 Signing entities

In principle, a message should be signed by an entity that originates the message. Since system information blocks (e.g., MIB and SIB1-5) are originated by gNBs, gNBs should be able to sign those messages thus need to store the private keys used to sign the messages.

To minimize security risk, some deployments may want to reduce the exposure of signing keys and keep them in secure locations (e.g., at core networks to reduce the number of entities that can access the signing keys). This requirement could arise when the security protection of an entity (e.g., gNB) is not considered sufficient, e.g., due to the lack of physical security or the use of a shared environment, or outsourced management of the environment, or other security reasons.

Note that the significance of a signing key is different from other symmetric keys (e.g., for AS security) stored in gNB, the latter is of local impact while the former is of global impact. More specifically, a compromise of a particular gNB and the stored symmetric keys only allows an attacker to access traffic for the users served by that gNB. However, a compromise of a signing key may allow an attacker to sign arbitrary system information, resulting in attacks on many more potential users. Thus, a signing key needs to be protected with more caution.

In the undesirable case when a signing key is stored in a gNB location, a central signing entity, e.g., the Digital Signing Network Function (DSnF) in solution 20, at the core networks can be used to sign messages for a gNB so that it does not need to store the signing keys.

**(D1): Both gNB and a core network function can perform digital signing of system information based on operator’s deployment and security requirements.**

##### 6.X.2.1.2 Raw public keys or certificates

Digital signatures are generated by a private key and verified by the corresponding public key. Public keys and trust anchors need to be distributed to UEs to enable them to verify digitally signed system information. A public key can be distributed in the form of a raw key or a public key certificate. A raw key consists of a public key, and maybe some other data such as a key identifier. A public key certificate such as X.509v3 has a well-defined data structure supporting key life cycle management, key hierarchy, and key usage, among other functionalities.

A raw public key is simple and of smaller size (comparing to a public key certificate), and is usually flat (i.e., it does not belong to a key hierarchy). This creates a challenge in trust anchor management. If each gNB uses a unique public and private key pair, each gNB’s public key would need to be configured as a trust anchor in UEs, resulting in many trust anchors and the difficulty to provision them out of band. If multiple gNBs share a public and private key pair, the number of trust anchors can be reduced but such practice will violate the basic security principle of key separation.

A public key certificate has rich semantics and supports key hierarchy, allowing multiple public key certificates to be verified using one trust anchor. But a public key certificate (e.g., X.509v3) is usually larger in size and may not fit into a single SIB due to the size limitation of SIB in the physical layer. Further, a public key certificate use leads to further management complexities and needs to be verified for its revocation status, e.g., using Certificate Revocation List (CRL) or Online Certificate Status Protocol (OCSP). Both CRL and OCSP require network connectivity, which had not yet been established at the stage of SIB acquisition.

One way to address the certificate revocation issue is to use short-lived public key certificates [8]. Since a public key certificate has a short expiration time (e.g., a few hours), a revocation check can be skipped assuming a compromised key will expire quickly. One challenge with short-lived certificates is that new certificates need to be issued frequently, thus requiring the issuing CA to be online. There are security risks in operating a CA online, which should be avoided if possible.

To this end, we propose a hybrid approach that combines the use of public key certificates and short-lived public keys. More specifically, a public key certificate is issued to an entity at the core network. This core network entity (e.g., DSnF in Solution 20) uses its private key to sign a gNB public key and other minimal data such as expiration time, key usage, etc. This signed object is valid for a short period of time (e.g., a few hours) and is referred to as short-lived public key (similar to the idea of delegated credentials for TLS [6]).

The advantages of using short-lived raw public keys include a) short-lived raw public keys belong to a key hierarchy, allowing them to be verified using one trust anchor; 2) certificate revocation check is not needed; b) there is no need to operate a CA online, since the short-lived raw public keys can be issued by an end entity (e.g., DSnF); and d) a short-lived raw public key is about 150 bytes (using 256-bit ECDSA) and can fit into a single SIB.

**(D2): A public key certificate is issued to an entity in the core network, which then issues short-lived public keys to gNBs.**

##### 6.X.2.1.3 Trust anchors

Based on D2, a certificate chain consists of a root CA certificate, zero or more intermediate CA certificates, an end entity certificate, and a short-lived public key (see Figure 1). In theory, any certificate (from the root CA certificate, any intermediate CA certificate, or an entity certificate) on this certificate chain could be provisioned as a trust anchor. However, if a CA certificate is configured as a trust anchor, all subsequent certificates under the trust anchor including the short-lived public key must be sent to UEs for them to verify signed system information, resulting in higher transmission overhead.



Figure 6.X.2.1.3-1: Certificate chain and trust anchor

Based on Table 6.20.2.5.1-1 in solution 20, an X.509v3 certificate for ECDSA-256 with SHA-256 is of about 545 bytes. However, a SIB has a size limit of 2976 bits (372 bytes). It would take two system information blocks to carry one X.509v3 certificate, which does not appear desirable. Thus, we want to try to avoid transmitting public key certificates to UEs to minimize message overhead. In this solution, the end entity certificate issued to the core network is configured as a trust anchor. In this way, only a short-lived public key needs to be transmitted to UEs and it can fit into a single SIB.

**D3: The trust anchor on the UE is an end entity certificate issued to a core network entity that issues short-lived public keys to gNBs.**

##### 6.X.2.1.4 PKI trust models

A PKI is often used to manage the lifecycle of public key certificates. Various PKI trust models are discussed in solution 20. While a fully centralized PKI (e.g., GSMA managed PKI for all mobile operators) significantly reduces the number of trust anchors required, it also creates deployment dependence and other issues (e.g., operational costs).

Since an end entity certificate, not a CA certificate, is configured in UEs as a trust anchor, there is no obvious benefit in adopting a centralized PKI. Therefore, decentralized PKIs, each of which can be managed by an individual operator, are supported to remove deployment dependence on other parties.

**D4: A decentralized PKI trust model is supported, i.e., each operator issues its own certificates and manages its own trust anchor.**



Figure 6.X.2.1.4-1: PKI trust models - centralized vs decentralized

Note that a trust anchor (an end entity certificate) can be issued by an existing Certification Authority (CA) that an operator already has, or by a new PKI that is created (e.g., using tools such as OpenSSL) for the sole purpose of digitally signing system information.

The challenge in trust anchor provisioning in roaming scenarios (e.g., each roaming partner has its own trust anchor) can be managed with in-band provisioning (see 6.X.2.1.5).

##### 6.X.2.1.5 Trust anchor provisioning

Trust anchors can be provisioned into UEs (e.g., in USIM) using either out-of-band or in-band mechanisms. Out-of-band provisioning of trust anchors leaves no risk window since it allows a UE to verify signed SIs starting from the first registration with the network, thus it offers the best possible security. However, it may create dependence on other parties, particularly in roaming scenarios. In-band provisioning of trust anchors allows an operator to provision its own trust anchor independently while offering deployment flexibility. However, in-band provisioning may have a risk window during which a digitally signed message cannot be verified if the trust anchor has not been provisioned.

To avoid risk window, trust anchors should be provisioned out-of-band if possible, e.g., by a home network (the procedure for provisioning the public key for SUPI encryption can be reused), or by a Standalone Non-Public Networks (SNPN). In-band provisioning (e.g., based on the procedure proposed in solution 7) should also be supported to update existing trust anchors (e.g., due to certificate expiration).

**D5: Trust anchors are provisioned into UEs (e.g., stored in USIM) out-of-band, e.g., by a home PLMN or an SNPN but can be updated in-band when necessary.**

In roaming scenarios, the trust anchor of the roaming partners should be provisioned by a user prior to leaving the home network. If the trust anchor of a roaming partner is not provisioned, the UE is still able to select the partner’s network by following the cell selection strategy outlined in 6.X.2.3, provided that UE security policy allows for such selection.

Note that not all trust anchors of all roaming partners need to be preconfigured into the UE at the same time, reducing storage requirement of the trust anchors. For example, if a UE only visits a few roaming partners, the UE will only be provisioned with a few trust anchors. In a possible but unlikely event that a UE visits all the roaming partners and there is not enough storage space for all trust anchors, a stored trust anchor can be overwritten by a new one, e.g., based on the last time the trust anchor was accessed.

Editor’s note: Process and requirements for trust anchor/signature validation upon SIB reception are ffs.

##### 6.X.2.1.6 Delivering signatures and short-lived public keys

With an entity certificate configured as the trust anchor, the digital signature (along with some other data, e.g., a timestamp) of System Information (SI) and a short-lived public key needed to be delivered to UEs for them to verify the SI. We consider three options for delivering a digital signature and a short-lived raw public key.

First, both the digital signature (e.g., of 64 bytes) and the short-lived raw public key (e.g., about 150 bytes) are included in a SIB. Although the combined size is about 214 bytes, which can fit into SIB (up to 372 bytes), it would reduce its capacity by more than half (e.g., to about 160 bytes). This does not appear ideal. Further, while a signature is based the content of a particular SIB, the short-lived raw public key cam be valid for many instances of different SIBs. Thus, neither is it necessary nor desirable to transmit the short-lived raw public key within each SIB.

Second, a new SIB (namely SIB\_x) is defined to carry both the digital signature and the short-lived raw public key. This option has the advantage of leaving other SIBs untouched. However, it may create a new problem in the scheduling of the new SIB\_x. Since the digital signature is computed based on the content of a particular SIB, the new SIB\_x carrying the digital signature may need to be scheduled precisely according to the schedule of existing SIBs to ensure that the SIBs and the digital signature acquired in two separate messages match each other. This is of particular challenge for SIB1, since it is repeated several times within a periodicity of 160ms. Although the repeated SIB1 may not change in content, timing related attribute associated with a repeated SIB1 may change. Thus, separating the digital signature from SIB1 would require precise scheduling of the new SIB\_x, which does not appear desirable.

Third, the short-lived raw public key is carried in a new SIB, but the digital signature of SIB is carried within the SIB. This option adds reasonable amount of message overheads to a SIB, but avoids the complexity of scheduling the new SIB\_x. Thus, we select this option.

**D6: the digital signature protecting a SIB is carried within the SIB, and the short-lived public key used to verify the SIB signature is carried by a new SIB to be defined.**

##### 6.X.2.1.7 System information to be protected

System information (SI) includes MIB and 9 SIBs (SIB1-9). We next discuss how each SI should be digitally signed.

Since MIB does not have enough space to carry a digital signature, one option is to digitally sign MIB and SIB1 together. Note that MIB and SIB1 are broadcasted at subframes 0 and 5 respectively at their scheduled frames. One challenge is that MIB and SIB1 have different periodicities of 80ms and 160ms respectively, and MIB and SIB1 are also repeated (e.g., 8 times) within their respective periodicity. Luckily, the SFN field within the MIB contains only the 6 most significant bits of the 10-bit SFN, thus it changes every 160ms. In other words, although MIB has a smaller periodicity than SIB1, it remains unchanged within the periodicity of SIB1, i.e., every 160ms. Therefore, MIB and SIB1 can be digitally signed together. For example, in Figure 3, there are 9 MIBs shown, each of which is broadcasted at SFN=0 (0x0000000000) to 8 (0x0000001000) respectively. While the 10-bit SFN changes in every MIB repetition, the 6 most significant bits (MSBs) of the SFN contained in the MIB remain 0x000000----) and do not change within 160ms.



Figure 6.X.2.1.7-1: Examples of MIB and SIB1 scheduling

**D7: MIB (including the SFN field) and SIB1 are digitally signed together with one digital signature.**

Since SIB2-5 may be broadcasted at very different periodicity from SIB1 or may be acquired on-demand, it does not appear practical to sign some of them together or with SIB1. Thus, each of SIB2-5 can be signed individually. Assume that SIB1 and the new SIB\_x will be acquired prior to SIB2-5, only the digital signature of SIB2-5 needs to be delivered, which can be carried within each SIB.

SIB6-8 are originated from an external entity to the core network (CBCF and AMF) and then to gNBs. We currently do not consider the digital signing of SIB6-8 in this solution.

#### 6.X.2.2 Replay mitigation

Digitally signed messages cannot be tampered with but can be replayed. In order to mitigate replay attacks, message timeliness needs to be provided. This is often accomplished by including a time variant parameter along with the message when computing its digital signature. In addition, some properties of a message originator, if they cannot be easily spoofed, can also be included in the computation of the digital signature to further enhance mitigation against replay attacks.

##### 6.X.2.2.1 Message timeliness

Three types of time variant parameters are often used to provide message timeliness or uniqueness, including *random numbers*, *sequence numbers*, and *timestamps*.

A random number is often used in real-time interactive communication, such as a challenge-response authentication protocol, to ensure a response is uniquely computed based on a specific challenge. Since system information is not acquired by UE interactively with gNB, a random number is not suitable for providing message timeliness in system information.

A sequence number, either a serial number or incremental counter value, can uniquely identify a message. To use a sequence number to detect replay attack, a recipient needs to receive each and every message from an originator (e.g., gNB) and also maintains a state (e.g., the latest sequence number) for the originator. These requirements do not hold in system information broadcasting, since a UE does not acquire every system information block from every gNB. Thus, sequence number is not suitable for detecting replayed system information.

Timestamp can be included with a message to provide message timeliness to mitigate replay attacks. It does not require any state to be stored by a recipient but requires the local clocks of a message originator and a recipient to loosely synchronize. Given that a UE synchronizes its clock with a gNB prior to acquiring system information, we suggest that it is reasonable to assume the clocks between UE and gNB can be loosely synchronized. Therefore, a timestamp is recommended for mitigating replay attacks.

**D8: A timestamp is included with each SIB, and is digitally signed along with the SIB to mitigate replay attacks.**

A UE can check a timestamp in a SIB against its local time to see if the time difference is within an allowed window (t\_w). The smaller the t\_w is, the more likely a replay can be detected.

In a typical scenario where a message is sent out after a local time is obtained and the digital signature is then computed, the total network delay for the message to reach a receiver includes:

1. the computational delay - the time taken to compute the digital signature (in the order of us, based on Figure 4);



Figure 6.X.2.2.1-1 - OpenSSL speed test for ECDSA on a VM

Editor’s note: The timing performance is based on a different software and hardware platform than the one typical for UEs. The timing performance on UE is FFS.

1. transmission delay at the sender – the time taken to transmit the message out from the sender’s memory to the network (in the order of us);
2. the propagation delay – the time taken to transmit the message from the sender to the receiver over the network (e.g., over the air) (in the order of us), and
3. transmission delay at the receiver – the time taken to transmit the message from the network to the receiver’s memory (in the order of us).

In this scenario, the total network delay (t\_d) is likely less than 1ms, but let’s say t\_d=1ms . Thus, the allowed delay window (t\_w) can be set to a value slightly higher than the network delay (t\_d+τ) to accommodate time deviation between the sender and the receiver. Let’s say τ=2ms, which would be the required accuracy of time synchronization. Then t\_w= t\_d+τ = 1ms+2ms = 3ms. Note these numbers are used here just as examples.

In reality, MIB and SIBs are broadcasted according to precise scheduling. More specifically, a MIB is broadcasted every 80ms and repeated every 10ms (at the starting of a frame), and SIB1 is broadcasted at a periodicity of 160ms and also repeated within a periodicity (e.g., every 20ms at the starting of a half frame). There are several options in assigning timestamp (thus a new digital signature) to SIB1.

First, each SIB1 repetition is assigned a new timestamp (see Figure 5). Thus, a new digital signature is also computed for each SIB1 repetition in every 20ms. In this case, there is no scheduled delay to be accommodated. Thus, t\_w= t\_d+τ = 1ms+2ms = 3ms.



Figure 6.X.2.2.1-2: Each SIB1 repetition has its own timestamp and a digital signature

Second, all SIB1 repetitions within a periodicity share a common timestamp, as well as a common digital signature (see figure below). Thus, a new timestamp is obtained, and a new digital signature is computed at the beginning of each SIB1 periodicity of 160ms. In this case, the allowed delay must include the maximum scheduled delay (t\_s), i.e., the delay of the last SIB1 repetition within the periodicity, which is 140ms. Therefore, t\_w=t\_s+t\_d+τ=140ms+1ms+1ms=143ms.



Figure 6.X.2.2.1-3: All SIB1 repetitions within a periodicity carry the same timestamp and digital signature

Third, it is also possible to have multiple SIB1 repetitions within a periodicity or across multiple periodicities that share a common timestamp. To generalize, let’s assume a timestamp is shared by N>=1 SIB1 repetitions. Then the maximum scheduled delay t\_s= (N-1)\*20ms. Thus, t\_w= t\_s+ t\_d+τ =(N-1)\*20ms +3ms. The value of N has two aspects.

On one hand, the smaller the N is, the more effective it is against replay attacks. For example, if N=1 and t\_w is set to 3ms, it would make it very difficult, if not impossible, for a false base station to successfully replay a SIB1. Since a replayed SIB1 has to be processed by the UE portion of the false base station and then broadcasted based on scheduling, it could add more than 3ms of delay, thus can be detected. However, if N is larger than 1, e.g.,N=8 and t\_w=143ms, it may leave enough time for an attacker to replay a SIB1 without being detected.

NOTE: Reusing a same signed timestamp in all SIB1 within a given time-period reduces the CPU overhead; however, it also gives more chances to an attacker to perform a replay attack if the time check is performed as above. For instance, a time window of 143 ms is a long period of time when 5G achieves ~ 1millisecond latency. A way of addressing this issue is to modify the time validation as follows:

* First, it is required to sign the time and SFN when the signed SIB timestamp appears first. These two values are denoted as SIB\_time\_ref and SIB\_SFN\_ ref.
* Second, when a UE receives a signed SIB1, the UE checks the SFN in which the SIB1 has been received, and obtains the sending time of this received SIB as: SIB\_time = 10\*(SFN – SIB\_SFN\_ref) + SIB\_time\_ ref.
* Third, the UE checks the validity of the SIB as follows:
	+ 1) The UE compares its current time UE\_time with SIB\_time and accepts the received SIB if the difference between SIB\_time and UE\_time is smaller than a given threshold. This check acts as a first filter and does not create any additional latency in the acceptance/validation of the received SIB.
	+ 2) the UE computes when the next SIB including a “new” SIB\_SFN \_ref and SIB\_time\_ref is to be received. The UE can compute it given the SFN value of the received SIB. The UE waits till that SIB with the new SIB\_SFN \_ref and SIB\_time\_ref is to be received and checks that this SIB including the new timestamp is received at the expected time. This second check is required since an attacker (MitM) might be delaying the communication in purpose, e.g., to perform a replay/MitM attack. This second check introduces a delay in the acceptance/validation of the received SIB proportional to the timestamp reuse factor.

This alternative approach achieves an accuracy t\_w as small as in the case in which each SIB1 includes a new fresh signed timestamp at the cost of increasing the delay in the acceptance/validation of the received SIB proportional to the timestamp reuse factor.

On the other hand, the smaller the N is, the higher the computational cost it incurs. For example, if N=1, a digital signature needs to be computed every 20ms. Based on the OpenSSL speed test of ECDSA-256 on my VM, it takes about 24us to compute a digital signature. Although the computational overhead appears small, it is not negligible, particularly when one gNB-CU needs to compute digital signatures for multiple gNB-DUs in a distributed gNB architecture. Therefore, the value of N needs to be balanced between security and performance. To this end, we recommend leaving N configurable.

**D9: The number of SIB1 repetitions that share a timestamp and the digital signature should be configurable and can be adjusted dynamically by network operators.**

##### 6.X.2.2.2 gNB unique properties

Since timestamp alone may not fully mitigate replay attacks if the allowed delay window is large, additional parameters may be used to augment timestamp to improve replay mitigation. For example, if a sender has some properties that cannot be easily forged by an attacker or can be detected if forged, such properties can be included along with a timestamp to improve mitigation against replay attacks.

To this end, PCI and downlink frequency, as proposed in Solution 7, can be included in the computation of digital signature of system information. To replay a digitally signed MIB/SIB1, an attacker would have to use the same PCI and downlink frequency as a legitimate gNB. Although both PCI and downlink frequency can be forged, a forged PCI may allow a UE to detect the presence of an attacker when two cells use the same PCI. When an attacker is forced to use the same downlink frequency as a legitimate cell, it may introduce additional signal interference, also an indication of the potential presence of an attacker. Thus, both PCI and downlink frequency help improve mitigation against replay attacks, albeit not preventing them.

**D10: PCI and downlink frequency can be digitally signed along with timestamp and MIB/SIB1 to further mitigate replay attacks.**

Editor’s note: It is ffs the need and discussion of (dis)advanges of (not) signing additional fields in the PBCH such as the beam index.

##### 6.X.2.2.3 Time synchronization issues

By using a timestamp to mitigate replay attacks, we introduce an assumption that UE’s time and gNB’s time are loosely synchronized, e.g., to an accuracy of 2ms. This assumption is realistic since UE and gNB need to synchronize their clocks in the physical layer to the accuracy of 1ms or less and IEEE 1588 provides time synchronization accuracy in the order of 10-100ns.

However, we need to consider scenarios where UE’s time may not be synchronized with gNB’s time. Examples of such scenarios include, but are not limited to, a) The first time a UE is powered on and has never synchronized with a network before; b) UE has been powered off for too long so that its time is out of sync with the network; and c) UE’s time may be manipulated (e.g., by an attacker using other methods).

In those scenarios, the UE may reject the timestamp from a legitimate gNB, resulting in a denial of service of UE. This can be mitigated by checking the consistency of timestamps from multiple cells (assuming the gNB’s clocks are synchronized), as proposed in solution 20.

For example, let’s assume that a UE is located within the coverage area of two cells, each of which assigns a new timestamp for each SIB1 repetition. Let’s assume that the allowed delay window (t\_w) is set to 3ms, but the UE’s time deviates from a gNB’s time by one hour (3,600,000ms).

When the UE acquires a SIB1 repetition from the first cell, the SIB1 timestamp is t1 and the UE’s time is u1. If we ignore the network delay, the difference between u1 and t1 is about 3,600,000ms, let’s say |u1-t1|=3,600,000ms. Since this is significantly larger than t\_w=3ms, the UE will not accept the SIB1.

When the UE acquires another SIB1 repetition from the second cell, the SIB1 timestamp is t2 and the UE’s time is u2. With the network delay ignored, |u2-t2|=3,600,000ms, which is also significantly larger than t\_w=3ms. Thus, the UE will not accept the SIB1 from the second cell either.

This time deviation from the network could result in denial of service to the UE. However, by checking the consistency of |u1-t1| and |u2-t2|, the UE can detect that its time has gone out of sync with the network and can proceed to select one of the cells since their times are consistent. Note that if the two cells assign a new timestamp to SIB1 repetitions at different rates, the scheduling delay needs to be considered, e.g., by leveraging the redundancy version in each SIB repetition.

In the presence of FBSs, it can be considered as a byzantine fault problem and the consistency check of times continues to work as long as 2/3 of the received SIB1 by a UE belong to real base stations.

Editor’s note: the protocol for consistency checking and its underlying assumptions needs to be clarified and detailed.

Editor’s note: potential limitations of consistency checking (e.g.., when a single gNB in reach) and discussion on complementary methods (if required) is ffs.

**D11: Consistency check of gNB times can be performed when UE’s time goes out of sync with the network to avoid failure in timestamp verification.**

##### 6.X.2.2.4 Limitations

Digital signatures provide authenticity and message integrity of system information, preventing an attacker from originating any MIB/SIBs or tampering with an existing MIB/SIB. However, a timestamp may not prevent a replay attack if the allowed delay window is set to a value that is long enough for a replay to succeed.

Further, PCI may not always allow a UE to detect a replay if the UE is out of the range of a legitimate gNB whose system information is being replayed. This can be illustrated in Figure 6.X.2.2.4-1.

Editor’s note: Figure 6.X.2.2.4-1 assumes an omnidirectional antenna. Attacking area when a gNB uses beamforming is ffs.

Assume R1 be the broadcasting area of a real base station (RBS1), and F be the broadcasting area of the false base station (FBS). If the FBS replays the system information from RBS1 and uses the same PCI as the RBS1, A UE located within the intersection of R1 and F, denoted by F∩R1, will be able to detect the duplicated PCI from two base stations. However, a UE located within the area of F but outside the area of R1, i.e., F-F∩R1, will not be able to detect the duplicated PCI. Although the area where UEs may be attacked by an FBS is reduced by digitally signing PCI, an FBS may choose to replay the system information from another RBS to expand its attacking area. For example, the FBS can replay the system information from RBS2 to attack UEs located within the area of F∩R1.

|  |  |
| --- | --- |
|  |  |

Figure 6.X.2.2.4-1: Attacking areas of an FBS by relaying system information from RBSs

Although it is possible to replay system information in theory, there are practical challenges an attacker needs to overcome in order to succeed.

First, to replay system information in real-time (or relay system information), an FBS needs to include a UE and a gNB. By using the same PCI and downlink frequency, an attacker must find a strategy to prevent FBS-UE from connecting to FBS-gNB.

Editor’s note: FBS-UE and FBS-gNB are under the control of the attacker. It is ffs what prevents an attacker from modifying the software of the FBS-UE to not connect to the FBS-gNB.

Second, even if an FBS manages to successfully relay system information from a legitimate gNB, it is not clear if it can also successfully lure UEs away from their camped cells. According to [9], an FBS needs to use a tracking area code (TAC) that is different from a UE’s current TAC to trick the UE into believing that it has entered a new tracking area to reselect the FBS. Since TAC in SIB1 is integrity protected, the FBS is unable to modify the TAC. Thus, it is not clear how much impact replayed system information could have on UEs currently camped on the nearby cells.

Third, replayed system information can be detected by the UEs which can receive both the original system information and the replayed one. In other words, an FBS cannot go undetected. It is not clear what strategy an FBS can adopt to hide.

Fourth, when a replayed system information is detected based on duplicated PCI, a UE could report to the network. If necessary, the network can adjust its configuration on the SIB1 timestamp to reduce the allowed delay window so that other UEs can detect replayed SIB1 based on timestamp.

To summarize, while it is theoretically possible to relay integrity protected system information, the security impact from the relay attack remains to be seen. We suggest this could be an active research area in the near future. Additional mitigation can be further introduced if needed, e.g., based on future research results.

It is also worth noting there are legitimate devices that need to relay signals from legitimate cells, e.g., to expand network coverage. For example, RAN recently approved a new WID [10] on NR repeaters. Thus, it is also necessary to allow system information to be relayed.

#### 6.X.2.3 Cell selection and reselection

With MIB/SIB1 digitally signed, cell selection and reselection need to take into consideration of not only signal strength, but also the authenticity and freshness of MIB/SIB1. Section 6.20.2.5 in solution 20 provides an example of how cell selection and reselection can be performed based on digitally signed MIB/SIB1.

Table 2 lists 10 possible scenarios based on the combinations of the states of digital signature, trust anchor, timestamp, and PCI uniqueness. The first case (Case #A) is the normal case, in which the digital signature of MIB/SIB1 is valid, the public key used to verify the digital signature is trusted (i.e., signed by a provisioned trusted anchor), the timestamp is fresh, and there is no PCI conflict. In this case, the cell can be immediately selected if the signal strength meets the criteria. In all other cases, there is some issue with at least one of the parameters. Therefore, in any of those cases, the cell cannot be immediately selected, nor fully rejected. If UE immediately selects a problematic cell, security could be compromised. If the UE fully rejects such cell, it may result in UE service degradation or outage.

We suggest that UE should prioritize cells for selection and reselection based on the results of MIB/SIB1 verification. Table 2 suggests a priority for each case. The first case (#A) is given the priority of 1.1, the highest priority. Case #B is of priority 1.2, the second highest, since it is likely that the UE clock is out of sync with the network, requiring consistency check with additional cells. Cases #C and #D likely indicate that the UE leaves its home network and enters a roaming partner’s network. The rest of cases are likely the results caused by an attacker. Note that the last case (#J) is likely from a legitimate base station that does not implement digital signing. However, we may not want to give it a higher priority. Otherwise, a MITM attacker could strip digital signature from SIB1 (and other related fields) to pretend that a cell has not implemented the feature.

If all available cells have a priority of 3.x, the UE could decide how to proceed based on local policy. The strictest security policy may force the UE to go out of service temporarily since an invalid digital signature or the absence of a digital signature does not provide guarantee that the cell is authentic. A moderate security policy may allow the UE to randomly select one of the cells to continue the service but may log and report the event. Although there is a security risk here, it forces an attacker to interfere and jam all other cells in order to force this scenario, significantly raising the bar for the attack.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Case # | Priority in cell selection | Is digital signature valid?  | Is digital signature trusted | Is timestamp fresh?  | Is PCI unique?  | Notes |
| A | 1.1 | Yes | Yes | Yes | Yes | Normal case |
| B | 1.2 | Yes | Yes | No | Yes | UE clock is out of sync with the network, or replayed MIB/SIB1 |
| C | 2.1 | Yes | No | Yes | Yes | Trust anchor has not been provisioned (e.g., first time in a roaming network) |
| D | 2.2 | Yes | No | No | Yes | Trust anchor has not been provisioned, and UE clock is out of sync with the network or replayed MIB/SIB1. |
| E | 3.1 | Yes | Yes | Yes | No | Replayed MIB/SIB1 |
| F | 3.1 | Yes | Yes | No | No | Replayed MIB/SIB1 |
| G | 3.1 | Yes | No | Yes | No | Replayed MIB/SIB1, no trust anchor |
| H | 3.1 | Yes | No | No | No | Replayed MIB/SIB1, no trust anchor |
| I | 3.2 | No | Not relevant |  Not relevant | Not relevant | MIB/SIB1 has been tampered with, e.g., using bit-flipping attack. Other parameters become irrelevant since they can be forged.  |
| J | 3.2 | Without signature | Not relevant | Not relevant | Not relevant | Signature may have been stripped, or digital signing has not been implemented (e.g., eNodeB) |

Table 6.X.2.3-1 - Priority list for cell selection and reselection

**D12: UE should support prioritize cell selection and reselection based on the results of MIB/SIB1 verification and UE local security policy.**

### 6.X.3 Assessment using clause A.3

#### 6.X.3.1 UE aspects

UE needs to be provisioned of a list of trust anchors (see clause 6.X.2.1.3).

UE needs to support secure storage of trusted anchors.

UE needs to take into consideration results from signature and freshness verification of SI for cell selection and reselection (see clause 6.X.2.3).

UE needs to be configured with a local policy to prioritize security and availability for cell selection and reselection.

#### 6.X.3.2 UE actions upon detection of invalid signature

Upon detection of invalid signature, UE should perform cell selection and reselection as described in clause 6.X.2.3.

#### 6.X.3.3 Threats that are mitigated by signed SI messages

Man-on-the-side attacks (e.g., SigOver), man-in-the-middle attacks (e.g., replay and relay), and some denial of services (e.g., from tampering with SI) are mitigated.

#### 6.X.3.4 Threats that are not mitigated by signed SI messages

Some denial of services (such as from bit-flipping or radio jamming of all available cells) cannot be mitigated if all available cells are under the attack simultaneously.

#### 6.X.3.5 Provisioning of keys

Trusted anchors can be provisioned into UE during manufacture, onboarding, or after registration.

#### 6.X.3.6 RAN aspects

gNB needs to obtain short-lived keys from the core network.

gNB needs to perform digital signature computation of SIs.

gNB needs to broadcast digital signatures along with SIs, as well as the short-lived public key.

#### 6.X.3.7 VPLMN aspects

If the trust anchor of VPLMN is provisioned into a UE, the UE is protected when accessing the VPLMN.

#### 6.X.3.8 HPLMN aspects

If the trust anchor of HPLMN is provisioned into a UE, the UE is protected when accessing the HPLMN.

#### 6.X.3.9 Network sharing aspects

When a gNB is shared by multiple PLMNs, the operator owning the gNB can issue short-lived public key to the gNB, as long as the trust anchor of the gNB operator is provisioned into a UE.

#### 6.X.3.10 Roaming aspects

See clause 6.X.3.6 VPLMN aspects.

#### 6.X.3.11 Regulatory aspects

Regulatory requirements, if there are any, can be supported.

#### 6.X.3.12 Signature schemes

Potential signature schemes include:

- **ECDSA (recommended with named curves)**

Editor's Note: the ECDSA profiles for SUCI can be reused.

- **RSA**

- others

#### 6.X.3.13 Signature length

RSA: 256 bytes

ECDSA: 64 bytes

#### 6.20.3.14 Resistance against Quantum Computing

TBD.

### 6.X.4 Evaluation

This solution addresses KI#2 and fulfils its potential security requirement by providing message authenticity to prevent unauthorized modification and mitigate replay of system information independently of UE state.

This solution requires UEs to be provisioned with a trust anchor (e.g., the public key certificate of a network) to verify the authenticity of messages signed by a particular network. An operator deploying this solution can prevent its own subscribers from accepting unauthorized system information within its own network. If a roaming partner also deploys this solution and the public key certificate of the roaming partner network is provisioned into a UE, the UE is also prevented from accepting unauthorized system information over the roaming partner's network.

This solution requires SIB1 to be extended to carry a digital signature.

This solution requires new a system information block to carry a short-lived public key used to verify the digital signature.

This solution requires gNBs to communicate with the core network to obtain short-lived public keys.

This solution supports incremental deployment with incremental security benefit (i.e., protecting UE against unauthorized system information) in that it can be deployed over time by an operator independently.

Editor's Note: Further evaluation is FFS.