Challenges of Standardizing Renewable Broadcast Security

Hagai Bar-El
hagai.bar-el@discretix.com

Abstract

An important component of a secure broadcast scheme is a renewability mechanism. This mechanism enables the system to cope with successful attacks that are widely perceived to be inevitable. When standardizing a broadcast security scheme one needs to address not only the technical issues, such as secure delivery and update, but also the conceptual difficulty of overcoming the inherent unsuitability of standardization processes for providing timely response to attacks. This paper discusses some of the challenges of both types and suggests ways to overcome them.

1 Introduction

Renewability is an important factor of content protection mechanisms. It is acknowledged that no scheme can be written which will resist all future attacks. This especially holds true for schemes that are used to protect multimedia content in consumer devices. The high incentive of hackers for breaking these devices, along with the nature of such devices that are often operated within a hostile environment, leads to the understanding that any scheme that is ever deployed in them must assume that it will eventually be broken, and allow for its recovery from such compromise.

Renewable schemes are widely available, especially in pay-TV and satellite broadcast products. Many proprietary set top boxes and broadcast schemes employ renewability mechanisms to assure that a hack of a component does not necessarily lead to the complete collapse of the system. It is desired to have attacks handled much like software bugs - as occasional errors that need to be fixed and that can be fixed. Standardizing a scheme that allows for renewability is a challenge mainly due to the complexity of setting a standard, which is permanent in nature, for achieving renewability, which is, by-definition, constantly evolving.

This document will address some of the difficulties that are encountered when standardizing renewability and suggest possible solutions. The challenges will be divided into two: The first part of this document (chapter 2) will address the conceptual challenges as the one mentioned above, which derive from the general concept of standardized renewability. The second part (chapter 3) will address some of the technical concerns that any renewability mechanism must address.

2 Conceptual Challenges

This chapter discusses challenges that are conceptually bound to the fact that the renewability mechanism is to be standardized.

2.1 Renewable While Standardized?

As written above, the concept of renewability assumes the ability to replace a vulnerable component with a new, perceivably more robust, one. The renewal process assumes that the modification of the system is done sometime throughout its lifecycle of the renewed device and involves information which was not known when the device was designed. A good renewability mechanism assumes quick responsiveness of the system designers to flaws, while assuring the suitability of the device for accepting these fixes.

A renewability mechanism assures that devices are technically capable of incorporating the fix as soon as it is delivered, but assume the availability of the fix. As opposed to the design of proprietary solutions, which typically involves a single company's engineers making up a fix for their own product, the design of a standard fix for a standard scheme involves a longer and more complex engineering and political effort, which does not suit the "quick-response" model that renewability relies on. A fix to a vulnerability that takes several months to design and agree on is not much better than no renewal at all because the incentive for hackers to break the system is not eliminated, as it is with prompt renewal. Also, the time during which exploits may be available (and lead to continuous leakage of prime content or service) may be long enough to

hamper the business case of the conditional access system.

The author perceives this to be the biggest challenge when designing a standard renewable scheme. Following are a few of the approaches that can be taken to overcome this limitation.

2.1.1 Re-Keying-Only Renewability

A somewhat trivial solution could be to limit the renewability mechanism to one that only allows keys to be refreshed, rather than enable the introduction of more complex scheme changes, upon attack. It is assumed that attacks will usually result in the compromise of one or more keys and thus replacing the compromised keys will be all that is necessary to prevent the attack from returning its cost.

This is a risky assumption because the attack is often such that can either be scaled or re-run continuously to compromise new keys after their introduction into the device. As long as this is the case, renewability that offers re-keying only has no way to ever halt this loop. Given that it takes longer to renew a compromised system than it takes to run the hack, re-compromise the system, and leak content or service through it (especially if discovered pirated content or service is the trigger for the renewal process), persistent damage to the content owner is almost guaranteed to be caused in this scenario.

Re-keying must be supported in every renewability scheme because a compromise of a device results in revealed keys that lead to content or service leakage to some scope, depending on the key that was disclosed. However, as demonstrated above, if the compromise is such that can be automated, as the case is for most software hacks, a renewed key is not enough.

Nevertheless, key renewal may be effective for itself as long as the attack is of a discrete nature, e.g. an attack that is too expensive to repeat on the same or on other devices. This condition typically applies to hardware based attacks on specific devices or models. Another assumption that must hold true when relying on re-keying-only renewability is that the attacks will always target the key stores rather than the scheme, its protocols, etc. An attack against the scheme, protocol, etc. is, by its nature, cheap to repeat.

2.1.2 Generalizing the Scheme

The generalization approach aims at renewing parts of the scheme as well, not just the keys, when the need arises. When using the generalization approach we assume that there will not be enough time to fix the standard when an attack occurs and therefore it should be made sure that the standard contains enough leeway for fixes to be introduced within its scope, that is, without violating compliance on one hand and without harming interoperability on the other hand. This can be achieved by drawing a generalized standard and allowing the actual scheme being used at any given moment to be a narrow profile of this standard. This approach to the standard can bear the benefit of renewability as long as the particular profile that is used at any given time is well defined at all points, and thus no ambiguity is caused, and as long as the profiling method is versatile enough to allow the required fixes to be introduced by merely changing profiles.

When adopting such a model for renewability the scheme should be divided into two distinct components:

- A *permanent component*, which is the fixed, non-renewable, part of the scheme. This component consists of modules that provide elementary functionality such as encryption and decryption. There is no particular function of the scheme that this component must provide, but at a minimum this component must provide the renewability functionality. Moreover, introduction of updates shall not be possible unless done by a module of the this component.
- A *modifiable component* that consists of modules that can be replaced if found broken, and that may rely for its execution on functions provided by the permanent component.

The permanent component is the focus of the standard. The scope of the standard can be the functionality specification of the permanent component and the way it interprets and interacts with the modifiable component. With regards to the modifiable component, the standard addresses only its form and interface with the permanent component.

A somewhat similar concept is presented in an MPEG-4 IPMP Extension[1] that defines the reliance on “tools”, which are not part of the device definition, for content protection.

[1] *MPEG-4 IPMP Extension*, Ming Ji, SM Shen, Wenjun Zeng, Taka Senoh, Takafumi Ueno, http://www.cs.missouri.edu/~zeng/MPEG-4˙IPMP final˙manuscript.pdf

The first task that needs to be done when attempting to standardize a generalized scheme is the definition of what functionality belongs to the permanent component and what functionality belongs to the modifiable component. This is discussed in the next sub-chapter.

2.1.2.1 Drawing the Line

An important part of the standardization work is to define which components of the scheme belong to the permanent component and which belong to the modifiable component. When making this judgment the following restrictions should be considered:

- The module that is responsible for the renewability mechanism, along with all data that it requires for its secure execution (algorithm implementation as well as permanent pre-loaded keys) must be part of the permanent component. The update process must not rely on or make use of functions or data that are not provided by this component.
- Modules that are likely to be implemented by hardware (such as for bulk decryption) must be part of the permanent component.
- Modules that are perceived as more likely to being compromised, e.g. modules that are more complex, or that are otherwise hard to assure, should be part of the modifiable component.
- The permanent component must consist of modules that provide all functionality that is necessary for the device to execute any scheme that will ever be required of it, given the appropriate modifiable component. The permanent component should therefore support enough functionality to allow the scheme to be renewed without requiring additional modules to be added to the permanent component.

Other than following those strict limitations, dividing the functionality between the two components is a matter of balance according to these basic guidelines:

- The more functionality carried out by the modifiable component - the wider the scope of possible renewability is. A flaw can be fixed using the renewability mechanism (hence, without changing the standard) only if it can be fixed by introducing modification to a module of the modifiable component. Therefore, the more functionality put in the modifiable component, which is the non-standardized part - the more effective the renewability mechanism is.
- The more functionality carried out by the modifiable component - the more difficult it would be to design and carry-out the standard handling of this component so to retain interoperability. Sub-chapter 2.1.2.3 discusses the issue of making sure the modifiable component is interoperable.

Once it is decided what part of the functionality belongs to the permanent core and what part is modifiable, the main challenge is encountered. The challenge is the need to define the scheme and the method of renewability activation.

2.1.2.2 Definition and Provision of the Modifiable Component

Fortunately, the definition of the broadcast security scheme is beyond the scope of this discussion. As we discuss the introduction of a renewability mechanism we assume that a starting point, which is a base scheme, is already defined and implemented using the two components mentioned in subchapter 2.1.2.1. The permanent component of the scheme is fully standardized whereas the modifiable component is only standardized in terms of its structure and its interface with the permanent component. The part of the scheme that is implemented by modules of the modifiable component can be defined outside the standard and still fit the broadcast model as long as they follow the standardized structure and relation to the permanent component, and as long as the content provider supports them.

There still is a need to specify the method by which updates to the modifiable component are introduced in a way that does not require an additional time-consuming standardization effort, and while retaining interoperability.

At this point it is granted that all devices are capable of running any of the new (or fixed) schemes, following a renewal process. The only prerequisites that need to be fulfilled for executing the scheme is the correct operation of the permanent component, which is standard and is available on all compliant devices, and the interoperability of the modifiable component, which is addressed in the next sub-chapter. It needs to be assured, however, that all devices get and use the appropriate modifiable component. The following paragraph presents one possible way to get this done effectively.

Piggybacking on Content Stream. One way to get the new, corrected, modifiable component defined uniformly in all relevant devices is by shipping it with the service content stream. Not all content streams have to carry the same scheme definition (modifiable component) but as long as the device gets the right modifiable component that it shall use to process that content stream - the system works. A significant advantage of this method is that it allows the content owner to determine the type of scheme that is to be used to protect its content and deploy it without relying on other parties. Such an approach is part of the *Self Protecting Digital Content* concept[2].

For summary, the problem of having to have the new corrected scheme defined on time is solved by taking the modifiable component of the scheme out of the standardization scope. The new (or corrected) scheme can be defined by the content owner or by anyone else for that matter. As long as the content owner agrees to use it, and as long as it is delivered to all relevant devices, no standard agreement is required to retain interoperability, as all devices can process all modules of valid modifiable components given the common permanent component.

2.1.2.3 Assuring Interoperability of the Modifiable Component

Having the modifiable component with internals specified by the industry rather than by the standard assures that quick changes are possible, but at the same time allows for interoperability issues to arise. To prevent the modifiable component from introducing interoperability problems, the modifiable component must follow the openness requirements that apply for the other components of the system, so any device can use any modifiable component, regardless of its source. Two approaches may be considered regarding the form of the modifiable component, to assure that the modifiable component does not end up breaking interoperability:

Using an Interpreted Language. By this approach, the modifiable component is entirely written in an interpreted language (as opposed to being written in native code). The permanent component is responsible not only for the introduction of updates to code of this component, but also for its interpretation in runtime. The use of interpreted code that is executed by a standard interpreter assures that no matter who authors the modifiable component, it will be executable out of the box on all compliant devices.

Locally-Compiled Code. It is often desired to avoid interpreted code altogether, as it introduces cost and complexity to the permanent component as well as possibly vast standardization efforts. If this is the case, the modifiable component can consist of native code (compiled code), but only code which utilizes modules on the permanent component using a standard API and that is compiled from a standard language by the entity that ships it to the device, rather than by its author. This allows for the definition of the modifiable component to be done by the industry (see 2.2) in an open manner and in a way that assures compliance with all standard-compliant devices. It is assumed that the entity that delivers the modifiable component to the device (the last entity on the delivery chain) is aware of the device characteristics and can thus be responsible for checking and compiling the modifiable component code properly.

2.1.3 Pre-Defined Scheme Sets

The standardization body may not define a single scheme but a set (cluster) of possibly related schemes at once. When the need arises, a change of scheme will be decided upon, typically by the content provider, possibly following a recommendation of the standardization body or of any appropriate body. A target scheme needs also to be determined as it is possible that a single vulnerability that is found has an impact on more than one scheme of the pre-defined set.

2.1.3.1 Schemes Definition

There are numerous ways by which sets, or clusters, of schemes can be defined. The most straightforward way is by viewing individual schemes as discrete designs and inventing a set of such completely independent designs. Each scheme may be assigned a unique identifier that will be used to identify the scheme to the device when that scheme is to be used. The disadvantage of this method is that many schemes need to be defined to counter class vulnerabilities (vulnerabilities that affect more than one scheme) and each scheme may take significant efforts to define using this method.

An alternative method is by defining functionality clusters and defining the schemes as collections of elements from these clusters. This allows for the bulk generation of schemes, as

[2] *Self Protecting Digital Content*, Paul Kocher, Joshua Jaffe, Benjamin Jun, Carter Laren, Nate Lawson, http://www.cryptography.com/resources/whitepapers/SelfProtectingContent.pdf

variations of a few fundamental schemes, at lower a cost. When taking this approach to scheme generation the identification of a scheme may be based on a vector. This vector contains identifiers of elements which together form a combination that represents the scheme, e.g. a vector consisting of an ID of an encryption algorithm, an ID of a keying mechanism, an ID of a mechanism for service key derivation, etc. When a flaw is discovered, it will result in the ban of affected elements of various clusters, depending on the nature of the flaw, and in communicating (either through the content itself or off-band) a new vector, representing a new scheme, to the compliant devices.

2.1.3.2 Installation of Schemes on Devices

The device must have the set of schemes installed before switching to using one of these schemes. The set of schemes may take the form of a modifiable component (see 2.1.2) that can be delivered to the device at any time, either as part of content streams or by other means. Such a modifiable component may also be shipped with the device such as in a device that supports several schemes out of the box. Alternatively, the set of schemes may be provided as a permanent software and hardware component in the device, such as a part of the broadcast reception application.

It is advantageous to not ship the device with all schemes at once. Doing so might reveal information on possible future implementation flaws (flaws in schemes that are not used yet) to attackers. Obviously, the opponent knows the design details of all schemes before their showtimes, because these schemes are part of an open standard. Nonetheless, there is an added value to security that is gained by hiding the particular implementation code from the opponent for as long as possible, as this implementation code may contain coding-level implementation-specific flaws that might not have been discovered and fixed during the review process (see 3.1.3).

2.1.3.3 Pros and Cons

Using pre-defined scheme sets will solve the problem of having to wait for a new standard, once an attack has been discovered, but not entirely. At some point, it is possible that the pool of schemes runs out. When this occurs, a new set of schemes has to be defined and standardized. This approach does not reduce the standardization work that will inevitably be required. Moreover, some of the schemes may never be used. Due to wider-scope vulnerabilities, along with the possible relation between schemes, it is likely that some of the effort of designing standard schemes will go down the drain. However, this approach is significantly better than no built-in renewability in that it shifts the standardization effort and time to a period at which more time is conveniently available and at which every day does not result in leaked content or service. This assures that scheme standardization is done at leisure, each scheme is designed properly (and thus more securely), and, most importantly, that no content loss occurs during the time it takes to define the scheme, as would happen was no renewability implemented.

2.2 Ownership of the Process

It is not enough to have a renewability mechanism that allows devices to be updated. It is also necessary to have the process in place to activate this mechanism when necessary, as well as to make the necessary decisions according to the renewability mechanism that is used. For example, when the scheme is broken, and if using the generalization method presented in 2.1.2, someone needs to design the new scheme (program the new modifiable component). The standard's suitability for renewability assures that any scheme that is defined by the standard's rules is compliant and runs properly on all compliant devices, but this scheme needs to be defined. Similarly, when using the method presented in 2.1.3, someone needs to assess which ones of the pre-defined schemes should be considered as unusable (broken) and which scheme should be used from that point onwards.

Generally, the process of activating renewability consists of the following steps:

- Detecting that an attack has occurred and assessing its scope (see 3.1)
- Determining the implication of the attack on the existing scheme and on alternative schemes
- Deciding on the target scheme, or scheme variation, to be used from that point onwards
- Distributing the relevant information to devices

When renewability is part of a proprietary product, the vendor company is responsible for taking care of this entire chain. When the renewability is incorporated into a standard, expectations as to the ownership of the process

have to be well clarified before the system is usable.

One approach puts the burden of following this procedure on content owners. Content owners are the main stakeholders in the broadcast security scheme so they often seem to be the ones who are entitled and most motivated to make the decisions that have such a direct impact on the robustness of the service and content protection. Moreover, since the most direct interface to the device can utilize the content (or service) data, content generators are naturally put at a position at which they can program the device by using this interface without relying on other parties.

On the other hand, assigning the task to content owners alone puts a significant burden on content owners. The analysis of exploits, as well as the engineering of corrective actions, requires significant manpower, skills, as well as other costs that content owners may or may not wish to bear.

An alternative way is to expect the product provider (e.g. the device manufacturer) to have the facilities in place to support the renewability mechanism. These providers may have to be involved in the process of content packaging, to some extent, and thus provide both sides of the solution. This approach will allow providers to differentiate themselves also by the quality of service they provide which, in this case, translates also to security.

It is believed by the author that a standard specifying a renewable broadcast security mechanism should not necessarily cover in its scope the definition of the players responsible for utilizing the mechanism. The standard should define how renewability is to be supported in the application, in a way that allows elements of various vendors (devices, client-side software, server-side software, etc.) to interoperate, while leaving the process and ownership considerations to be determined by market conditions.

3 Technical Issues

This chapter lists technical issues that need to be addressed by the design of a broadcast security scheme, for incorporating renewability. These issues are not specific to the case of a standardized renewability scheme. Of course, the list of issues in this chapter is far from being all-inclusive but rather is considered by the author to cover the main security related issues that need to be addressed, along with possible solutions.

3.1 Triggering Renewability

An important component of a renewability method is its method of triggering. When using the broadcast security system, it needs to be determined *when* something was broken and *what* it is that was broken, so proper renewal can be carried out using the renewability mechanism. The *when* needs to be known so the renewal process is triggered, replacing the faulty part, and the *what* needs to be known so the update process takes the right form, e.g. the faulty module is positively identified and changed, the correct key is replaced, or a new scheme is chosen, that is perceived to not suffer from the same flaw that triggered the renewal process.

It is believed that a standard for broadcast security renewability should not delve into the detailed specification of the process by which the renewal mechanism is put to use. Nonetheless, this process must be considered when defining the standard so the scheme is designed in a way that allows the introduction of fixes that match elements that may be spotted as compromised.

Following are the methods by which attacks can be discovered and analyzed.

3.1.1 Existence of Pirated Content

A typical trigger indicating a flaw is the existence of pirated content in a darknet or otherwise illegally available. Unfortunately, this may provide the indication too late as apparently the specific content that was discovered has already leaked to outside the protected domain. Such discovery of pirated content, can, however, trigger a renewal act that will prevent additional content from leaking out. Another limitation of this approach is that content is often delivered using more than one delivery mechanism and in this case it is difficult to determine whether the fault is of the standardized protection scheme or of another delivery mechanism. Watermarking can often help in determining the source of the leak. Another problem is that the mere existence of pirated content tells the designers nothing about *what* component of the scheme is flawed and what the flaw is.

3.1.2 Monitoring Hackers Activity

Another way of knowing about new vulnerabilities that are introduced into the scheme is by research into the activities of the black-hat

communities. These communities consist of individuals and groups of people that discover flaws and distribute exploits that take advantage of these flaws, for the benefit of their customers.

As opposed to individuals who break broadcast systems for their own benefit of free-content or service, and who do not present a significant threat to the content industry, the higher-scale pirates break systems so they can obtain one or both of these benefits:

- *Community recognition*: The recognition and praise of the hacker communities to which they belong.
- *Monetary gain*: Revenue generated by selling products, content or services, made available to the attacker by exploiting the vulnerabilities that he discovered.

Fortunately, both incentives to break the broadcast systems lead to the ability to determine when and what is wrong with the scheme, and with modest effort.

Receiving the recognition of the community requires publicity of the attacks to great detail. While black-hat hackers do not publish their findings in the New-York Times, hacker forums and other forms of underground publications are often monitored by companies, so to be used as a source of information about exploits that are discovered. An advantage of getting the attack indication through this channel is that it is most informative. Other than knowing that the scheme was broken, one can often tell exactly what the design flaw that enabled the compromise was.

Attackers who discover flaws to be exploited for revenue are often less willing to share their exact findings. However, financial gain cannot be obtained without adequate marketing efforts, which typically include advertising the availability of cloned Conditional-Access cards, or of some service that provides premium content. At the time of writing, spam mail is one of the most commonly used bearers for such advertisements; the author alone gets dozens each week. Unfortunately, the way to knowing the nature of the flaw that enabled the attack is longer and typically requires one to buy a cloned device and reverse-engineer it to determine the flaw that it exploits. Forms of watermarking may also assist in determining the source of pirated content.

3.1.3 Utilizing Peer-Review

Another way to know about vulnerabilities is by legal reviews of the scheme. From time to time, vulnerabilities are discovered in fielded systems by white-hat hackers as well as by information security analysts. They do not use this information to impress the underground community or to sell illegal products or services, but rather to inform the stakeholders who get the chance to fix the flaw before it gets maliciously exploited. Independent peer-review may be the most valuable source of detailed information about flaws, hopefully before, but also after, those flaws are being exploited.

A standard broadcast security and renewability scheme has a very strong advantage over proprietary schemes in this respect. Generally speaking, proprietary schemes are not exposed to the cryptographic and security communities at large and this has a negative impact on the amount of peer-review they are ever subject to. A broadcast security scheme that is based on an open standard has an advantage of utilizing the aggregated skill of the academic, and non-academic, communities at large, for spotting vulnerabilities on time. Proprietary schemes may enjoy a head start resulting from their concealment, but their assurance in the long term is typically reliant on the skill of a small number of security reviewers.

Consequently, whereas peer-review is not the most notable source for information on flaws of broadcast systems today, the author feels that there is a strong basis for the belief that it can become a highly significant source of such information in a scheme that is based on an open standard. The cryptographic and security communities already share a common notion that good cryptography has to be publicly known and analyzed, and that applications are more likely to be secure if their source code is open for professional public scrutiny. The author's belief is based on this notion.

3.2 Communicating the Modification

For broadcast security renewability to work, it must be possible to communicate the modification to the device, when necessary. If renewability is based on permanent and modifiable components, then a corrected modifiable component needs to be delivered. If the scheme is based on a set of standardized methods, then it must be possible to notify the device of the method that is to be used from that point onwards.

The delivery problem can be split into two sub-problems: assuring the update is at all delivered and deployed, and assuring that it is delivered securely. These two sub-problems are ad-

dressed, in reversed order, in the following subchapters.

3.2.1 Assuring Secure Delivery

Delivery of the modification data (renewed modifiable component, scheme identifier, keys, etc.) can be done by several means, either by using the broadcast bearer or by using any other connectivity.

Security of the change delivery is of paramount importance and thus has to be implemented by a mechanism that is resistant to more powerful attacks than the broadcast mechanism itself. It is feasible to design a delivery mechanism for renewability that is stronger than the broadcast scheme it renews because the delivery security scheme is significantly simpler than the broadcast security scheme, having to fulfill a much narrower set of functional requirements.

To understand the need for a significantly more robust update (renewal) mechanism, let us briefly examine the impacts of a compromised update mechanism. As a first, immediate, consequence, if the update mechanism is broken, then the device cannot be renewed. Given there is a method for assuring update prior to trusting a device with new content (see 3.2.2), this will not imply that the compromised device can leak content or service indefinitely. However, if the attacker can remotely damage the renewability mechanism of a large number of devices, he may prevent flaw-fixing from being possible, possibly leading to an extended lifetime of the flawed system. The above mentioned requirement for handling renewability by the permanent component alone is aimed at mitigating this scenario.

A more serious impact of a compromised update mechanism is that it might lead to the attacker's ability to introduce his own data (and code) into the device. If the attacker can get this accomplished, two risks occur:

- The attacker can use the update mechanism to introduce new, intentional, vulnerabilities into the system even if the system was secure before. For example, the attacker can change the scheme implementation on his own device to one that surrenders the device keys, or even just program keys, which can in turn allow other devices to receive service they are not entitled for.
- At the worse case, which is less likely, if the above mentioned requirement, of having the update mechanism implemented by the permanent component alone, is not thoroughly met, the attacker may be able to introduce flaws to the device that will survive future updates. This may result in non-fixable service or content leakage.

In addition, secure delivery must be assured to protect keys in transit. Scheme updates are likely to include key updates so to be effective (see 3.2.2). An unprotected key delivery mechanism may result in an attacker being able to gain possession of valid device keys that can later be used for device or card cloning, as well as for unauthorized access to services.

Additionally, lack of strong cryptographic binding between the update data and associated keying data may allow an attacker to enjoy continued service while avoiding the update, thus being able to leak service and content indefinitely (see 3.2.2).

The update delivery mechanism should thus be more robust than the broadcast scheme itself. The following guidelines may be followed to obtain this goal:

- The update mechanism should be as simple as possible, so to make it easier to assure. E.g. it should not consist of overly-complex data objects or of complex authorization levels. The security protocol involved should be the most fundamental one that can carry out the task properly.
- The update mechanism should be implemented in a tamper-proof way. It shall be part of the permanent component, so not to enable its indefinite substitution, and it must rely on hardware based security, so to prevent compromise using software tools, such as downloadable exploits and debuggers.
- The update mechanism must use common cryptographic means to assure the authenticity, integrity, and confidentiality (when needed), of update data as it arrives, before using it.
- The keys that are used to assure authenticity and integrity of the update data must be kept in tamper resistant storage that is protected against key compromise or replacement.
- Keys used for the above mentioned validation must not be shared with other components on the device, must not be used for any other purpose, and must not be exportable from the update mechanism in

plaintext form. They must also not be provisioned using less-secure device provisioning mechanisms.

- Global secrets, e.g. global keys, in devices must be avoided at any cost.
- It should be impossible to carry out any update, by whatever mechanism, other than by running and following the process enforced by the update mechanism. E.g. it must not be possible to make changes to the broadcast security scheme at large (including the modifiable component) using general-purpose download and update facilities that may be available on the device.

3.2.2 Guaranteeing Update

It is important to assure that a device has gone through all the expected renewability procedures before trusting it with new content or with further service. Failure to meet this requirement practically voids the renewability mechanism altogether. If an opponent can prevent a device (typically his own) from being updated, while having access to new content, or having further access to a service, then content could be leaked from that device indefinitely. Alternatively, following a compromise the attacker will be able to design an exploit or clone, which will allow free access to protected services indefinitely, regardless of a possible issuance of a fix.

Fortunately, it is easy to assure that devices are updated prior to providing them with content, by binding versions of the scheme to updated, relevant, key material, depending on the flaw that lead to the update. By re-keying devices following an update, and using the new keys, or their derivatives, to deliver further service, it can be assured that a device cannot obtain access to new content unless it was properly updated. The fact that an update is carried out only by a permanent component means that this component can enforce binding of the scheme update to a matching key update, e.g. so not to allow an update session at which re-keying occurs without being accompanied by a matching change to the scheme. Given the assumed robustness of the update handling module (see requirements in 3.2.1), fake updates will not be able to re-key the device properly. As this key is required to obtain new content, no new content will be obtainable by the device until the correct update is introduced to it.

There are additional ways by which the update module can enforce using the correct version of the scheme before allowing access to new content or to further service. These additional ways should be subject to further study.

4 Summary

We discussed two types of issues that need to be addresses when standardizing a broadcast security scheme that supports renewability. The discussion was limited to the issues that need to be resolved to enable security renewability. The first two issues were ones that emerge from the fact that the solution is to be defined and maintained by a standardization committee, rather than by a single provider. First, the scheme has to support amendments without requiring re-standardization. This could be solved by limiting renewability to re-keying, by splitting the scheme into permanent and modifiable components, or by defining a set of possibly-related schemes. Second, the industry needs to define an owner for the renewal process, which can naturally be done outside the committee without violating the purpose of the standard.

The second part of the document discussed a few of the technical issues to be considered when defining renewability mechanisms, not necessarily as part of a standard. This chapter presented the problem of knowing when attacks occur and their nature, which can be achieved using either information from the underground (as also done today), as well as by using the facility of the professional communities' peer-review, which is a typical bonus for designing security in open standards. Another issue was getting the update to be delivered and deployed in a secure manner on the device. This issue can be resolved using well-established and fielded technologies of digital signatures, encryption, and binding of content to keys.