

## **Refusal, Remediation and Renewability in Marlin**

THE MARLIN DEVELOPER COMMUNITY, INCLUDING INTERTRUST, PANASONIC, PHILIPS, SAMSUNG AND SONY MAKE NO REPRESENTATION OR WARRANTY, EXPRESS OR IMPLIED, CONCERNING THE COMPLETENESS, ACCURACY, OR APPLICABILITY OF ANY INFORMATION CONTAINED IN THIS DOCUMENT. THE MARLIN DEVELOPER COMMUNITY, INCLUDING, INTERTRUST, PANASONIC, PHILIPS, SAMSUNG AND SONY DISCLAIM ALL LIABILITY OF ANY KIND WHATSOEVER, EXPRESS OR IMPLIED, ARISING OR RESULTING FROM THE RELIANCE OR USE BY ANY PARTY OF THIS DOCUMENT. THIS DOCUMENT IS PROVIDED TO YOU "AS IS". THE MARLIN DEVELOPER COMMUNITY, INCLUDING, INTERTRUST, PANASONIC, PHILIPS, SAMSUNG AND SONY MAKE NO REPRESENTATIONS CONCERNING THE APPLICABILITY OF ANY PATENT, COPYRIGHT OR OTHER PROPRIETARY RIGHT OF A THIRD PARTY TO THE INFORMATION CONTAINED IN THIS DOCUMENT OR ITS USE. THE RECEIPT OR ANY USE OF THIS DOCUMENT OR ITS CONTENTS DOES NOT IN ANY WAY CREATE BY IMPLICATION, ESTOPPEL OR OTHERWISE, ANY LICENSE OR RIGHT TO ANY PATENT, COPYRIGHT, TRADEMARK OR TRADE SECRET OF THE MARLIN DEVELOPER COMMUNITY, INCLUDING, INTERTRUST, PANASONIC, PHILIPS, SAMSUNG AND SONY, WHICH ARE OR MAY BE ASSOCIATED WITH THE IDEAS, TECHNIQUES, CONCEPTS OR EXPRESSIONS CONTAINED HEREIN.

# Refusal, Remediation and Renewability in Marlin

Marlin is a secure, trusted system for governed access to media and media-related network services. Marlin provides a rich set of security mechanisms to manage risks and mitigate threats that arise from attacks and compromises of devices, applications and services. Marlin approaches security holistically, recognizing that risk management is a continuous process which must support a dynamic lifecycle. Marlin systemically integrates renewal and relies on renewal mechanisms to respond to security events, remediate a breach, and restore failed components to a trustworthy state. It suffices to say a renewable system survives changing requirements and market demands better than a fixed solution. By designing for renewability, Marlin is poised to adapt to new and evolving requirements that a compliance regime may stipulate.

## 1 Introduction

This paper describes the various technical mechanisms that can be used to mitigate threats. Additionally, it describes circumstances in which it is appropriate to apply a given countermeasure and the complementary renewal mechanisms for informing systems of the failed entity and remedy the ostracized entity. Remediation in this context relies upon renewability to bolster the defenses of the system. In brief, the renewal mechanisms of Marlin serve two distinct purposes. First, they can be used to convey up-to-date information to trusted system entities that allow them to refuse access or service to distrusted system entities. Second, the renewal mechanisms enable a distrusted entity to regain trusted status by updating the compromised component. The refusal countermeasures can be further characterized as exhibiting one or more of the following behaviors:

- Revocation, or annulling a credential (typically by blacklisting some entity)
- Exclusion, or denying access by applying cryptographic or policy enforcement mechanisms
- Shunning, or denying access or service based on an identity or some other attribute bound to a credential
- Expiration, or annulling a credential or privilege based on a temporal event.

Marlin prescribes refusal mechanisms to counter threats such as device cloning, impersonation attack, protocol failures, policy enforcement failures, application security failures, and stale or suspicious information.

Table 1 below describes the possible threats, risks they pose, and mechanisms to remedy the threat and renew system security.

**Table 1: Threats and Countermeasures**

<b>Threat</b>	<b>Risks</b>	<b>Remediation Mechanism</b>	<b>Renewal Mechanism</b>
Cloned Device	Free-access devices.	Broadcast Encryption	BKB Update.
Compromised Certified Key	Unauthorized licenses, links, device state, identities, service access.	Certificate Revocation	CRL Distribution. Key renewal.
Implementation Failure	Recipes for device hacking.	Specification Version Assertion	Software upgrade
Protocol Failure	Compromised keys. Ungoverned access to licensed content.	Security Metadata Assertion	Software upgrade
Stale Security Metadata	Bogus service interaction. Clock rollback, reliance on compromised information.	Security Metadata Assertion	Security Metadata update service. Software upgrade.

## 2 Revocation

Revocation is a remediation mechanism that relies on blacklisting an entity. Typically, what is revoked is a credential such as a public-key certificate. Upon revoking a credential, the blacklist must be updated and a renewal mechanism must be used to convey the update so that a relying party may benefit from the remedy.

In Marlin, the credentials belonging to devices and services that pose a threat to the system can be revoked. In order to get content or access a service, Marlin devices can be required to present identity certificates, other credentials, and a variety of security data before they are given the information necessary to consume content. Similarly, in order for a Marlin client to trust a service (and so use it), the service may need to provide its credentials to the client. Marlin provides the means for entities to inspect one another to ensure that the ones participating in a service interaction are playing their proper roles with the proper credentials and have trustworthy software and security data. Marlin service providers can check that this information is available and is valid before providing service.

There are several methods that Marlin entities can use to effectively invalidate information necessary for service access, such as:

1. Use of Certificate Revocation Lists (CRLs)
2. Use of credential and data validity services, such as an Online Certificate Status Protocol (OCSP) responder
3. Response to commands for self-destruction of credentials and data

### 2.1 Certificate Revocation Lists (CRLs)

Marlin uses a services framework that prescribes a number of effective means for distributing lists, including CRLs. The Marlin trust infrastructure employs an indirect CRL, so there is a single CRL governing the entire ecosystem. In addition, Marlin entities can advertise (or publish) the CRL and entities can subscribe to an update service. The CRL can be distributed peer-to-peer in a viral fashion and/or a portable device can receive a published CRL when it is tethered. This is a highly effective way of distributing CRLs. Although such P2P mechanisms are not currently required, over time Marlin will provide a number of alternative mechanisms for efficiently distributing CRLs and other lists that include both requirements and normative methods for the P2P propagation of CRLs, security data, and system security advisories. NEMO has methods that can be used for these tasks.

Within Marlin, different role players can use revocation lists simply and effectively to revoke identity certificates, licenses, Octopus links<sup>1</sup>, and other security assertions. This mechanism is most effective to remedy the situation which results from a service being compromised.

## **2.2 Validity services**

Validity services can be used to provide up-to-date information on the status of credentials and other security related data. Validity services can perform either active validation operations on behalf of a relying party or they can be used to manage security information on behalf of relying parties. An example of an active validity service is one that can check the validity of a credential or attribute. Examples of validity services that manage security information are those which disseminate CRL or security policy updates, or provide a secure time service.

Marlin requires the use of various validity services to ensure that relying parties have current data to inform governance decisions.<sup>2</sup> Marlin's services framework is flexible enough to allow for future expansion of the services it exposes. Thus, new types of validity services can be defined and deployed to accommodate changing security requirements. Although the initial implementation of a validity service is in the Broadband context, other delivery system implementations (e.g., Broadcast or Mobile Gateway) may make use of this capability in the future.

Not all system entities will need up-to-the-minute information on the validity of credentials and security data. For example, not all consumer devices will use an Online Certificate Status Protocol (OCSP) service to validate a license server's certificate chain each time a license is used or a new license is obtained. However, a license server may use an OCSP service with some frequency to check the validity of subscriber credentials. Policy (which can be easily updated) can determine when and what services must be used. By providing an opportunity to dynamically update policy, license servers can adapt to operational changes. Thus, Marlin allows security policy to evolve based on experience, technological progress and market factors. Consequently, mechanisms that have a higher probability of being effective will be deployed.

## **2.3 Directed self-destruction of security objects**

Self-destruction of credentials and data by an entity is appropriate when the integrity of the entity's security processing is not suspect. When this option is available, it is often the most straightforward, expeditious, and efficient method of revocation. It should be used whenever there is no suspicion of breach of integrity, and bi-directional communication supports a protocol allowing specific directions for destruction along with verification that destruction has been completed.

In Marlin, there are a number of security objects that ought to be destroyed or disabled. For example, when a device leaves a domain, or a content license times out, the associated objects that contain keys and can be used to access content should be destroyed. If the exiting device enforces policy, then those objects will be destroyed. There should be no necessity to immediately update shared key objects whenever the domain configuration changes. However, domain keys may need to be refreshed as part of a key renewal process to ensure relative freshness of key material. This serves a different purpose and a key update protocol that is attuned to the dynamics of domain state can be used.

The Octopus Agent Technology is well suited to the implementation of self-destruction mechanisms. Octopus Agents can be crafted to destroy state in secure storage (i.e., the SeaShell database) to affect changes in domain membership or to remove keys that are no longer usable (e.g., due to changes in

---

<sup>1</sup> An Octopus link is an object which is used to enable the expression of governance rules.

<sup>2</sup> This is discussed in the 'Marlin Core System Specification©' in the section on *Data Certification Trust Hierarchy*.

membership or policy.)

### 3 Exclusion

Exclusion is a remediation mechanism which bars a bad actor (or group of bad actors) from participating in future consumption of goods and services. Due to the severe consequences exclusion imposes, it should only be used when circumstances warrant. Exclusion relies on a mechanism that effectively blacklists the bad actors thereby prohibiting them from consuming media and media-related services. Dissemination of the blacklist relies upon a renewal mechanism to enable this remediation. However, exclusion does not necessarily provide a renewal mechanism to restore a bad actor to a trusted status.

#### 3.1 Key Exclusion

Key exclusion is a key management mechanism that is used to broadcast key information to a set of receivers (Marlin devices) in such a way that at any given time a decision can be made to logically exclude some subset of receivers from the ability to decrypt future content. This is activated by using efficient techniques to construct a Broadcast Key Block (BKB) that includes information necessary for each member of a large group of receivers to decrypt content.<sup>3</sup> The BKB is structured in such a way that it can be easily updated, excluding one or more members of the group from the ability to decrypt the content. In other words, the design of the BKB allows for an authority to update the system with a new BKB, so that a content provider can specifically exclude a target set of devices from making use of the BKB, even though s/he may have access to it.

This mechanism is particularly effective against a cloning attack, where a pirate reverse engineers a legitimate device, extracts its keys, and then deploys copies of those keys to clone devices. The clones externally act like the original, except that these clones will not necessarily adhere to the governance model. Once the compromise is discovered, an updated BKB can be deployed that excludes the compromised device and all of its clones<sup>4</sup>. However, Key exclusion incurs some storage, transport, and computation overhead that in some situations make it less efficient than other methods. This is especially true when the content is not broadcast or when there is a back channel.

Since Marlin core devices can receive content from broadcast receivers, and since BKBs are necessary to decrypt broadcast content, all Marlin Core devices must be able to receive BKBs. However, not all Marlin core devices necessarily need to be excluded using a BKB if there are other mechanisms for disabling them. For example, revocation or shunning can effectively eliminate a device that typically receives its content from an online service.

### 4 Shunning

Shunning is a remediation mechanism very similar in behavior to exclusion but with less severe repercussions. Essentially, it is a means for refusing service because of a runtime policy decision. An earlier section of this paper on CRLs alludes to shunning. Instead of more heavy-handed approaches to disable a device's capability through directed self-destruction or access denial via key exclusion, shunning offers a simple approach to disabling a device by having service providers refuse to supply it with services. With the current trend towards extending the value of devices by using externally provided services, shunning becomes a more effective security mechanism.

Device shunning is driven by policy and can be used to discriminate against entities (clients, servers, specific role players) that do not produce all of the appropriate credentials that policy requires. Policy could, for example, require that an entity demonstrate it has administered the latest security update. Therefore shunning can be either a consequence of revocation or the failure to take some specific

---

<sup>3</sup> BKB info is a necessary but insufficient condition for access to content.

<sup>4</sup> The broadcast encryption mechanism used in Marlin is defined in the Starfish specification.

action. Shunning can be facilitated in a peer-to-peer fashion using the inspection services defined by Marlin. Also, the Data Certification Service (an instance of a validity service) can perform shunning at policy enforcement time. After a system entity has been shunned, it can be informed of the specific credential or object that is failing to comply with the policy of the service. This can trigger the shunned entity to renew the object through a Marlin Security Data Provider service or Data Update Service.

## 5 Expiration

Expiration is a remediation mechanism that relies upon some temporal event to invalidate a credential or object. Expiration is effective in enabling temporary access to media or media services; once these have expired, the governance model ensures that access is no longer permitted. Effective use of expiration may require renewal mechanisms whereby the credential or object can be refreshed to enable continued access to media or media services.

Presently, Marlin uses expiration to invalidate credentials and Octopus links.

### 5.1 Expiration of credentials

In Marlin, certified keys have various expiry attributes assigned to protect relying parties. Expiration of credentials is used to ensure that entities whose certificates have expired are refused service and used in conjunction with key rollover and key renewal procedures.

When entities are expected to be frequently connected to a wide area network, best-practice dictates renewing credentials and other security data regularly. Another best-practice is to keep the validity period of these objects as short as reasonable. Various techniques such as overlapping validity periods and grace periods in validity checking policies can be used to ensure smooth operation during transitions. Short validity periods also help to reduce the size of CRLs.

### 5.2 Expiration of Octopus Links

Octopus links may be assigned validity periods. Upon expiration, a link is deemed invalid and therefore the governance engine will not consider it in the construction of its graph.<sup>5</sup> This mechanism can be used very effectively to enable temporary access to goods and services. Octopus links can be renewed so that continued access to media may be granted so long as it is permitted by policy. Because Octopus links are lightweight self-protected objects they can be easily distributed over peer-to-peer protocols. In Marlin, Octopus links (new or renewed) can be delivered by a service implementing the Provider DRM Object protocol<sup>6</sup>.

## 6 Renewability Mechanisms

Renewability mechanisms are pervasive in Marlin. As outlined in the previous sections, refusal and remediation mechanisms are complemented by a supporting renewal method.

The remainder of this section describes best practices and methods to renew security.

### 6.1 Application and Policy renewability

Marlin requires efficient application renewability mechanisms. Primarily, this is done to allow the rapid deployment of remedies to protocol failures, which are the dominant security problems seen in security applications (including in DRM systems). Software updates are then used to update the business logic and security protocols. When applications are designed to separate security policy and trust policy from application logic, a separate mechanism can be used to update policy; this is a less risky approach. In fact, peer-to-peer publishing mechanisms can be used to rapidly update policy. Otherwise, the application deployer's software update methods can be used to update security and trust policy. Marlin services can be used as notification for the need to update. Marlin provides a rich set of inspection and notification capabilities; how they are used is considered an implementation choice.

---

<sup>5</sup> The mechanics of Octopus governance are out of scope for this document.

<sup>6</sup> This is discussed in the 'Marlin Core System Specification©' in the section on *Service-Specific Protocols*.



## 7 Using the right tool for the right job

Lightweight tools should be used whenever they can get the job done. Using credentials with limited validity periods and policies that check validity dates can help keep the overall population of entities to a manageable size and eliminate the need for growing CRLs too rapidly. Shunning an entity rather than excluding it from access to keys can extend the lifetime of BKBs; moreover, it has the advantage of enabling fine-grained policies that can be temporary and change with circumstances. Different CRLs that track specific types of credentials of interest to different role players can be used instead of BKBs which can be deployed where they are most effective (such as dealing with cloned receivers). Policies should direct the use of online validity services when those services can be expected to provide a reasonable return on investment of time and effort, where fresh credentials are very important, and where slower revocation mechanisms are inadequate. When a Marlin node is likely to have integrity and can be expected to do the right thing, however a license or security object (such as an Octopus link for a subscription or a domain link) needs to be revoked, then the more reasonable approach is to tell the node to destroy the object. In such a situation, there is no need to tell the world that the license is invalid and there is no need to deploy a BKB or re-key a domain. Self-destruction driven by local policy or by an authoritative command is one of the more efficient methods for revocation.

Different situations call for different tools. It is necessary to deploy means that allow using the most efficient and effective tool for the job. It is also necessary to build new tools whenever possible using the existing toolset.

## 8 Conclusion

As Marlin moves into a deployment phase, the focus on operational compliance becomes paramount. A fundamental requirement of deploying a DRM system is to have the necessary hooks to renew security as needed. By design, Marlin takes this consideration into account and defines refusal, remediation and renewal into the core of the architecture.

With experience Marlin will evolve to address operational compliance requirements that include:

- Specifying more detailed operational requirements for system renewability, including scheduled updates that can allow us to introduce more effective security methods over time
- Creating an inventory of security objects used in Marlin, and
  - Determining reasonable validity periods for such objects
  - Determining the simplest, most effective and least disruptive methods possible for revocation and renewal
  - Comparing the various approaches and determine which approach is best given the user context and environment
  - Designing policies and methods that are driven by threat analysis
- Designing interoperable protocols for secure P2P propagation of security object updates
- Reviewing all requirements and implementation specifications to ensure that every opportunity is taken for self-destruction of security objects that are logically revoked.

Marlin currently has a number of effective methods for maintaining the integrity of a content distribution system. These are expected to evolve and adapt according to the needs of the market, not only for robustness and renewability, but also for utility and efficiency.