EDPACS
Publication details, including instructions for authors and subscription information:
http://www.informaworld.com/smpp/title~content=t768221793

Key Management: The Key to Encryption

Benjamin L. Tomhave

Online Publication Date: 01 October 2008

To cite this Article Tomhave, Benjamin L.(2008)'Key Management: The Key to Encryption',EDPACS,38:4,12 — 19
To link to this Article: DOI: 10.1080/07366980802265914
URL: http://dx.doi.org/10.1080/07366980802265914

KEY MANAGEMENT: THE KEY TO ENCRYPTION

BENJAMIN L. TOMHAVE

INTRODUCTION

If you have been involved with compliance efforts for the Payment Card Industry Data Security Standard (PCI DSS), then you are probably aware of Requirement 3 and its provisions for protecting the Primary Account Number (PAN). You may even have implemented a cryptographic system to protect the PAN (good job!). Everything seems grand until that fateful day when your Qualified Security Assessor (QSA) comes on-site and starts asking hard questions. Questions like "How often do you rotate the key?" and "What's your encryption policy?" and "Where do you store your backup key?" That sinking feeling, for many people, comes at this moment, when they realize that their homegrown or turnkey solution only addressed the technology need, and not the technology management requirements.

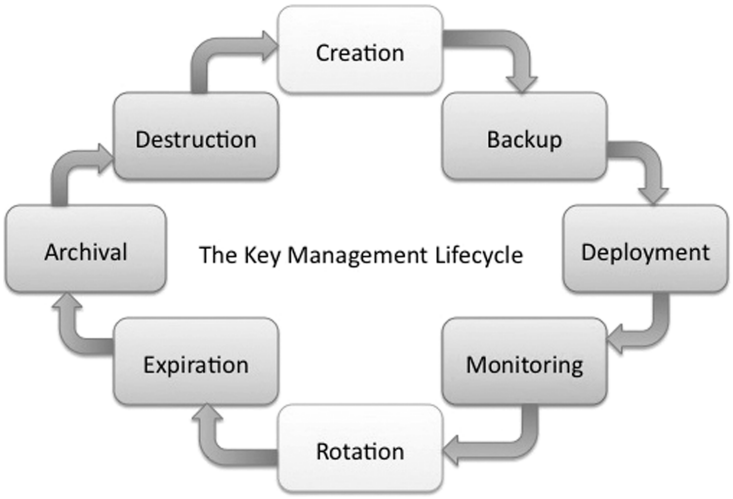
The goal of this article is to provide an introduction to the primary aspects of key management, as well as to introduce a few additional considerations. For the most part, the cryptographic systems for which this guidance is applicable will be servers and databases, rather than workstation solutions like whole-disk encryption. That being said, some of these concepts (e.g., key escrow) can be applied to endpoint solutions and may, as such, warrant additional investigation.

It is worth pointing out the work published by the U.S. National Institute of Standards and Technology (NIST) under Special Publication 800-57 (SP 800-57) as it is very complete, especially when compared with this article. Herein, an eight-stage process is advocated, which uses a high-level approach to key management. The NIST guidance advocates a four-stage process, each detailing numerous steps, along with supporting documents that provide further detailed guidance on many technical aspects of key management. It is an excellent reference, and one that should be considered when implementing technical key management procedures. The guidance that follows is intended to provide an introductory survey of concepts and concerns relative to key management, not to be a complete or technical procedural guide.

THE KEY MANAGEMENT LIFECYCLE

In a general sense, the key management lifecycle is compromised of eight major stages, as represented in Figure 1. Each stage represents a major process group that must be addressed both in documentation and in practice.

Figure 1 ***The eight stages of the key management lifecycle.***



Creation

The first step in the key management lifecycle entails generating the key. Key creation must be conducted in a secure environment (hardened system), and may include requirements for separation of duties. In most cases, the key in question is a symmetric key (a.k.a., "shared key"), which lends itself better to performance requirements. The key should be of an adequate size and strength, and is reliant in part on the underlying ability to generate a random number. Key creation should be performed using known good libraries that have been properly reviewed by knowledgeable people. Use of proprietary algorithms that have not withstood scrutiny should be avoided.

Once your key is generated, it may then be desirable to protect it by encrypting it with the public half of an asymmetric key pair (a.k.a., public key cryptography). In situations where the encryption key must be distributed to other systems, particularly via a network file transfer, it is often good to follow this practice.

From the standpoint of separation of duties, an organization can have one team own and manage the key generation capability, but bar them access to the encryption system itself, requiring them to instead encrypt the symmetric key with the public key before providing it to the deployment team.

Backup

Before a new key is rolled into a production environment, it is of the utmost importance that a backup of the key be made. The backup could be as simple as writing the key to external media (e.g., CD, DVD, USB drive) and storing it in a physical vault. Or, it may be

desired to back it up using existing traditional backup solutions (local or networked).

In either case, but especially in the case of traditional backups, it is highly recommended that a second asymmetric key pair be used to protect the symmetric key. Think of this second public key pair as your "escrow" key. As with the deployment key pair, it is recommended that the public key be used to encrypt the shared key, and then it becomes imperative that the private half of this key pair be protected and available for recovery operations.

One other consideration in performing backups is to apply the same disaster recovery plans that you would in any other business continuity planning process. The key should be stored in and retrievable from a location that can be accessed within the time requirements specified by the appropriate business owners.

Deployment

As already mentioned in the discussion of key creation, it is highly recommended that the symmetric key—used for the vital encryption activities—be itself encrypted by the public half of an asymmetric key pair prior to being delivered for deployment. This simple step provides a method for separation of duties in an environment where key management activities are not strictly contained within a single system (e.g., encryption appliances).

In all cases, from a separation of duty perspective, workflow practices should be documented and enforced such that one person is not able to perform or responsible for creation, backup, and deployment of the symmetric encryption key.

The purpose of this deployment phase is to introduce the new key into the cryptographic system, but this phase does not include removing the old key from that system. Specifically, it is advisable that the new key be deployed and tested for a pre-defined period of time to ensure that key operations are successful before risking a data outage.

With modern appliances, these concerns have decreased, but, from a lifecycle perspective, they are still worth bearing in mind. When working with cryptographic systems, one must tend toward caution, lest a key be lost, effectively removing access to important data. That is to say, errors with cryptographic systems can be quite costly.

Monitoring

The monitoring phase of the lifecycle has been jammed here into the middle, but it could just as easily be an area of responsibility that is parallel to the entire key management lifecycle. There are several materials aspects to monitoring that should be considered.

First, it is important to monitor for unauthorized administrative access to cryptographic systems to ensure that unapproved keys and key management operations are not performed. Any sort of unauthorized operation could have serious consequences for your system, and for your data.

Second, monitoring performance on your cryptographic systems is important. The performance of cryptographic operations tends to be processor-intensive, which means that your systems may be under significant load. Events such as flash popularity can result in

a denial of service on its own, but when combined with an overloaded encryption system, the results could be far more serious, including data corruption or unavailability.

Finally, as mentioned in the deployment phase, monitoring of the key in production is also important to ensure that the key has been created and deployed properly. If a corrupt key is deployed too quickly and without proper vetting, the results could be catastrophic. Similarly, if a fault in the cryptographic system occurs, the results could also interrupt service, with a negative impact to the business.

Rotation

The concept of key rotation has ties to key deployment, but they are not generally synonymous. In rotating keys, the goal is to not only bring a new encryption key into lead use by the cryptographic system, but to also convert all your stored, encrypted data to the new key. This process can be extremely time-consuming and processor-intensive. However, assuming that all previous steps of the lifecycle have been followed, then this phase can be discretely focused on the conversion activity, and less on the activating of the new key for new encryption requests.

The lifecycle of the first key (K_1) overlaps with the lifecycle of the newer key (K_2) at the Rotation (K_2) and Expiration (K_1) stages. The key in understanding these overlapping lifecycles is in realizing that key rotation is as much about the new key as it is about the old key. Rotation represents the turnover of primary cryptographic activities to the new key, usually in conjunction with the Expiration of the old key.

Note that there may be reasons not to perform a batch rollover of stored data from the old key to the new key. For example, if the data is highly transient (that is, accessed, written, or rewritten at a very high frequency), then it may be adequate to instead flag a condition where data will be automatically converted to the new key as part of the read/write activity. Using such a capability can reduce some of the load associated with key rotation.

A word of caution: Do not remove an old key from a production system until it can be proven that no data in production is still encrypted with the old key. Failing to perform due diligence, such as doing a manual query for the old key ID, may result in data loss or a service outage.

Expiration

The chosen strength of an encryption key will primarily take into consideration the length of time for which the data may be valid. The goal is to choose a key that is large enough that a brute force attack cannot theoretically succeed while the data is still valid or valuable. For example, if a credit card is valid for

four years, then you want to choose a key size such that it will take longer than four years to guess the correct key and thus retrieve the data.

In addition to key strength, it has also become best practice (and dictated by regulations like PCI DSS) to require that the key be expired and replaced on a timeframe shorter than the calculated lifespan of the key. The minimum time span that is advocated these days is one year for each key, and preferably less often for keys protecting data of the highest sensitivity. Note that this timeframe is much shorter than the presumed valid lifespan of the key.

The Expiration phase of the key management lifecycle represents the beginning of the deprecation period of life for the key. Key rotation for a new key should be completed prior to expiration of the old key, with all data encrypted with that old key converted to the new key. The objective is to have the key replaced within the production system (but not removed) before it expires. In a sense, expiration represents a gating factor to plan around as much as a discrete phase in the lifecycle.

Archival

The absolute last thing that you want to do when managing cryptographic systems is to destroy a key that still has data associated with it. As such, this second-to-last phase in the lifecycle is included, with a potentially open-ended mandate to not proceed to the final phase.

Archival of expired, decommissioned keys should be based on a determination of whether or not data still exists somewhere in the data ecosystem that may be encrypted with the archived key. The data ecosystem extends beyond "live" data in production to backups that may exist in disaster recovery sites, as well as to all offline backups. If there is a requirement for data to be recoverable, then it is imperative that the keys be archived in parallel to that data.

There are a couple tips to keep in mind when archiving a key. First, make sure to document and index the key and the data in such a way that should you need to recover data in the future with an archived key, you can do so in as effective and efficient a manner as possible. Second, ensure that the archived copy of the key has itself been secured. As was recommended in the Creation and Deployment phases, it may be useful to encrypt the symmetric key with the public half of an asymmetric key pair for safe storage. Finally, make sure to include a timestamp with the key, as well as an "effective" time range that reflects when the key was used for production purposes.

It is worth noting that some cryptographic appliances automatically archive expired keys in a secure fashion and may not ever move to the final phase, Destruction. Overall, this is a business risk decision that should be considered on a case-by-case basis. It may be rightfully concluded that encryptions keys will never advance from Archival to Destruction because the risk of such a change, and the associated permanence of the data loss, may outweigh the risk of exposing the archived key.

Destruction

The life of a key will truly end when it is destroyed. Key destruction should follow secure deletion procedures so as to ensure that it is properly obliterated. Be forewarned: key destruction should not be taken lightly, and should only occur after an adequately long Archival phase, and after at least two reviews have been completed to ensure that loss of the key will not correspond to loss of data.

OTHER CONSIDERATIONS

In addition to the basic key management lifecycle, there are other considerations that organizations may need to factor into policies, standards, and practices. For example, key loss can represent a catastrophic failure in a cryptographic system. To supplement backups, it may be deemed appropriate to maintain an escrow key that can be used to recover data in an emergency situation.

Exercising Key Management Processes

Having key management processes documented is a good first step, but it is equally important to test and exercise those processes on a regular basis. An adept emergency response team will be able to execute the first steps of an incident response intuitively and reflexively. So should key management be intuitive in the event that a key change needs to be made in a rapid manner (e.g., under a suspected key compromise scenario). Setting up parallel cryptographic systems in development or test environments may provide an easy means for regular testing of key management processes and procedures.

Separation of Duties

Alluded to earlier, there may business or regulatory requirements for separation of duties. The basic idea is that you do not want one person or team to have full end-to-end access to cryptographic keys, the cryptographic system, and the data. For this reason, it is often best to implement key creation as described earlier, where one team generates the key, then hands it off securely to another team for deployment. With respect to the data itself, the developers or operations personnel should not be able to decrypt protected data without proper authorization and monitoring.

Key Escrow

The concept of key escrow has been around for many years (if not a few decades). Much alarm was raised around the U.S. government's idea to hold the recovery key on behalf of users or businesses by way of the Clipper Chip system in the 1990s. Today, key escrow has grown to be viewed in less alarming and more useful terms.

The most common case for using key escrow is with whole disk encryption systems. If an end-user loses their key, then organizations want to have a backup method to recover the data from the system. However, key escrow also has applicability for data

encryption within data storage environments. For example, the Backup phase of the lifecycle could make use of key escrow to provide a safe copy of the primary symmetric key in order to prevent against the loss of production data.

Product Interoperability

One major problem encountered by organizations is when they have two or more cryptographic systems. Due to the requirement for use of cryptographic protection of sensitive data, it is increasingly likely that mergers and acquisitions will bring together disparate cryptographic systems as part of the IT integration. Unfortunately, although many systems will use the same types of keys (e.g. AES), it is not always a given that you will be able to take keys from one system and install them onto another. Similarly, the key management interface for either system may not work with the other.

In the end, there are two primary, competing recommendations for addressing this problem. First, organizations may choose to run both cryptographic systems in parallel, so as not to disrupt service. At the same time, documentation and key management processes should be integrated into one streamlined key management program, taking into consideration the competing needs of each line of business.

The other preferred approach is to pick one cryptographic system to become the primary encryption service. However, it is to be noted that, due to Archival requirements, the system that is not chosen may still need to be maintained over time until all residual data has been replaced or deprecated.

Catastrophic Failure

Implementation of a cryptographic system should bring out the disaster recovery specialist in all of us. Stop and ponder for a minute what would happen if all that encrypted data were to suddenly become unavailable. Now consider that in implementing a cryptographic system, you have introduced additional failure points. Suffice to say, use of encryption to protect data means that business continuity and disaster recovery planning needs to be involved and on-board so as to ensure that your organization can continue to function when faced with a complete failure. All it takes is the loss or corruption of one key to trigger a disaster recovery scenario.

CONCLUSION

Management of cryptographic systems can be stressful and complex. Defining the practices associated with each stage of the key management lifecycle, along with regularly exercising those practices, is vital to ensuring a healthy, successful deployment. Preparing for worst-case scenarios, such as a lost key or catastrophic failure, can help an organization handle such events rationally, rather than responding on the fly, with little or no preparation, and with the inevitable result of unnecessary downtime or exposure.

HELPFUL LINKS

- ☐ The Future of Encryption http://www.net-security.org/article.php?id=1113
- ☐ Payment Card Industry Data Security Standard v1.1 https://www.pcisecuritystandards.org/pdfs/pci_dss_v1-1.pdf
- ☐ The Risks of Key Recovery, Key Escrow, & Trusted Third Party Encryption (1998) http://www.cdt.org/crypto/risks98/
- ☐ CERIAS Weblogs » Confusion of Separation of Privilege and Least Privilege http://www.cerias.purdue.edu/weblogs/pmeunier/infosec-education/post-139/confusion-of-separation-of-privilege-and-least-privilege/
- ☐ Protegrity Awarded Patent on Database Key Rotation Method https://www.marketwire.com/mw/release.do?id=836482
- ☐ NIST Key Management Project http://csrc.nist.gov/groups/ST/toolkit/key_management.html
- ☐ NIST SP 800-57 *Recommendation For Key Management—Part 1: General (Revised)* http://csrc.nist.gov/groups/ST/toolkit/documents/SP800-57Part1_3-8-07.pdf

---

Benjamin L. Tomhave is a Senior Security Consultant for BT Professional Services in Reston, VA. He holds a Master of Science in Information Security Management from George Washington University, is a Certified Information Systems Security Professional (CISSP), member of ISSA, member of the American Bar Association Information Security Committee, and member of the IEEE Computer Society. Prior to BT, he worked in a variety of security roles for companies including AOL, Wells Fargo, ICSA Labs, and Ernst & Young.