www.mainframecrypto.com
gregboyd@mainframecrypto.com
Tel: 240-772-1539

Missing Newsletter?

For those of you that were wondering, there wasn't a July issue of the Mainframe Crypto Newsletter.

While I had planned on publishing these quarterly, the 3rd quarter of 2015 proved to be a challenge. The major issues was the fact that the wife and I decided to downsize our living quarters. Now that the kids have moved out, the house that we raised them in was much too large for just the two of us so we sold it in July, which meant that most of May and June we were packing and boxing up 25 years of 'stuff'. And I just ran out of time to get the newsletter published.

As part of the packing process I found a keychain with multiple keys … many of which I have no idea what they go to. And of course that has application in the cryptographic world. (Otherwise, I wouldn't be talking about our move in this newsletter.)

If you've heard me speak, then you know that I talk about key management as the hard part of cryptography. The algorithms are published and well-known. And the standards and protocols (like TLS) define the packaging structure. It's making sure that the keys are available at the right time and in the right place that is hard. Without key security and key integrity you won't have an effective cryptographic system.

In this Newsletter we'll talk about key management. First we'll cover solutions from IBM and secondly, since most customers don't have a key management product, we'll talk about what you can do to manage your keys without a package.

IBM Key Management Solutions

IBM offers several key management solutions. Two have been around awhile and the third is relatively new. Each product was originally intended to address specific requirements, but each has had functionality added to make it more robust and comprehensive.

Trusted Key Entry Workstation

The oldest key management solution from IBM is the TKE or Trusted Key Entry Workstation. The TKE is a stand-alone workstation that primarily is intended for secure key loading. It provides the most secure method for loading key material into the crypto hardware. There are three techniques for loading master keys into the hardware: Passphrase Initialization (PPINIT), ISPF ICSF panels and the TKE. With PPINIT your master keys are created via

software routines, so your master keys will exist, in the clear, in the TSO address space during the process. For the panels, your key parts (that get XOR'd together in the hardware) will exist in the TSO address space of the key officers before they are loaded into the hardware.

But with the TKE you create your keys inside the secure tamper resistant boundary of a crypto card inside the TKE. That key is then pushed up to the crypto card on the host system after establishing a secure communication session between the TKE and ICSF on the host. The crypto cards, both in the host and in the workstation, have their own public/private key pair assigned and a secure session is established between the cards for securely transmitting the key material from the TKE to the host cards.

The TKE is built on the same platform as an HMC (Hardware Management Console), but the TKE also has a Crypto Express card installed. Like the HMC, the TKE runs a closed operating system and each can only perform their intended function. The HMC can only manage the CEC and the TKE can only manage key material and the crypto environment.

Originally, the TKE was designed solely for secure key entry. It's one and only purpose was to securely get keys into the Crypto Express cards on the host. Over time, though there has been additional functionality added. The installation process for the TKE is quite complex and there is now an installation wizard that walks you through the required steps. The TKE also now includes a migration wizard, which helps with the migration from one hardware platform to another. The migration wizard can capture information about your current host crypto environment and then push that information to the new hardware platform. This wizard significantly reduces the amount of time that it takes to migrate to a new CEC when upgrading the hardware platform.

With recent versions of ICSF and the crypto hardware, IBM has started adding functionality that requires a TKE to enable the new capabilities. For example, with HCR77A0 and the CEX4S cards, IBM will support a 24-byte DES-MK. A longer DES-MK provides more security for wrapping operational keys, but the only way to enable the 24-byte DES-MK is by using the TKE to set the appropriate hardware switches in the crypto cards. Farther back, IBM introduced support for something called PIN Decimalization Tables. (PIN Decimalization is associated with protecting PIN keys.) The only way to load a PIN Decimalization Table is via the TKE. Finally, the new PKCS #11 secure key support requires that you use a TKE to load the PKCS #11 master key (P11-MK). A TKE is required if you want to take advantage of any of these functions.

The TKE however is not designed for the management of a large

volume of operational keys. It does not support clear keys either.

EKMF/DKMS

The second key management tool from IBM is EKMF, Enterprise Key Management Foundation also known as DKMS. DKMS, depending on who you talk to is the Distributed Key Management System or maybe the Danish Key Management System because it was developed by the IBM Crypto Competence Center in Denmark. DKMS was initially developed for European banks to manage the large volume of keys that are required to manage PIN keys. When the European banks started embracing EMV (Europay Mastercard Visa) standards, DKMS was expanded to support the public/private keys that are required to authenticate the smart credit cards.

Several years ago, IBM realized that key management was an important requirement for its customers and it realized that DKMS provided an already available solution. At that time, DKMS was repackaged as EKMF or the Enterprise Key Management Foundation. That foundation includes several other products developed by the Crypto Competence Center.

Like the TKE, DKMS also relies on a stand-alone workstation that contains a secure crypto express card. The DKMS application is designed to allow key custodians to generate key requests using a set of pre-defined templates. Those key requests are queued and eventually a key officer (or likely multiple key officers) then approve and actually execute the generation of the keys. As with the TKE, the keys are generated inside the secure card in the DKMS workstation and eventually pushed up to the host system. The host system may be z/OS but DKMS can also push keys to other operating systems as well as to some competitive crypto products.

The templates are pre-defined by the organization, allowing it to implement only the key types and key lengths that meet the organization's standards, thus enabling the organization's specific crypto policies.

DKMS does requireDB2 to be the database for the meta-data associated with the keys.

DKMS is designed to manage a large volume of keys, however like the TKE it does not support clear keys. It also does not support the loading of master keys. It does provide a secure offline repository for signing keys.

ISKLM (or EKM or TKLM)

The third key management solution from IBM was developed by the IBM Software Group to support the encrypting hardware technology. The first generation of this solution was called EKM or Enterprise Key Manager. It was a no-charge product that provides a way to manage keys and communicate those keys to the tape devices that

support encryption. EKM had a command line interface and it provided a communication link between the device and the keystore. That keystore could be a Java keystore (JKS) or RACF or ICSF or RACF and ICSF working together to manage and protect the key material.

Eventually EKM was replaced by TKLM or the Tivoli Key Lifecycle Manager, which was a priced product. TKLM provided a GUI interface, but it also required Websphere as well as DB2 to be the key management database. TKLM also provided formal support for encrypting DASD devices, where EKM only supported tape devices. TKLM was designed to run on multiple platforms, not just z/OS. Because of the cost of TKLM as well as the pre-req products, some customers chose to remain on EKM and TKLM was not widely accepted by many z/OS customers.

Eventually, IBM Software Group came out with SKLM or ISKLM, the IBM Security Key Lifecycle Manager. ISKLM is also a priced product, but it does not require Websphere or DB2 as pre-reqs. It also returned to using a command line as opposed to a GUI interface. ISKLM is a pre-req for using the encrypting TotalStorage devices on z/OS.

ISKLM does comply with the OASIS Key Management Interoperability Protocol (KMIP) standard, a security standard for the communication of key material. ISKLM, as well as EKM and TKLM, does not perform cryptographic operations, it simply manages and communicates key material. When a drive needs a key, ISKLM will communicate that request to a cryptographic system that will generate a symmetric key and encrypt that key under the appropriate public/private key pair(s) which are then securely transmitted back to the device that made the request.

Each of these key management products fills a niche for managing key material, but none is a comprehensive solution for managing all key types for all crypto end-points. If you are using the IBM TotalStorage encrypting devices, you must install ISKLM to provide the communication between the device and the keystore. If those are the only keys that you need to manage, then it's probably sufficient to only install ISKLM.

However, if you are using ICSF to perform any kind of symmetric work then there will be additional keys to manage. If it's only a handful of keys, you can probably manage those manually, using operational procedures and naming conventions (more about that in a minute). As already mentioned, PIN banking applications require a large volume of symmetric keys. Or if you're using a product like the Infosphere Guardium Data Encryption Tool for DB2 and IMS then the number of keys required can grow quickly. (Potentially you could have a unique key for each

table, or at least for each application, and you will need to rotate keys periodically and maintain a history of those keys.) DKMS is the best IBM solution for managing that large volume of symmetric keys.

Finally, a TKE is required for the most secure means of loading master key and it's also required to exploit some of the crypto functionality on the CEX cards (like 24-byte DES-MK support).

Hopefully IBM will bring these three solutions together into one product, running on one platform, however it's not a trivial transition. There are lots of considerations to ensure the security of the key material, no matter what type of key. A key management system must secure the key material both in storage and as it is being transmitted to an end-point. And it must be delivered to the end-point in a form that the end-point can use. And it must manage the meta-data associated with the key. Since the security of your data is provided by the key itself, not the algorithm, an attacker is not going to attack the algorithm, but try to gain access to the key material, either in the keystore or as part of the key management process.

Key Management without a Tool

While any customer that is using IBM TotalStorage encrypting hardware must have ISKLM (or EKM or TKLM) installed, those customers are not typically using ISKLM to manage other keys in the environment, because the tool is not really designed to manage other keys.

There are a number of customers that have a Trusted Key Entry installed but typically they are not using the TKE to its fullest capabilities. The TKE is its own separate system that must be implemented and managed properly. Plus customers are typically using it only to manage master keys, not operational keys.

Finally, DKMS (or EKMF) does not have wide penetration in the US. So customers are generally relying on operational procedures to manage their keys. Fortunately, ICSF provides some of the basics for implementing those key management procedures, and it starts with the label that is associated with a key. Each key has a 64 character label that is used to reference the key.

That label can use a convention like data set names, where the HLQ can associate the key with a user or an application (like a group). It's important that the label indicate ownership, so you don't end up with keys that no one knows who is using them. And of course that label provides the security framework because you can define your security profiles according to the label.

The label can also be used to manage the key lifecycle by including a date as part of the key label. Either the creation date of the key or the projected expiration date can be one level of the label and that can be a trigger for knowing when

the key needs to be rotated and/or retired.

All this implies a somewhat manual process for tracking and managing keys, but without a key management package that is the way most customers manage their keys today.

Starting with the new KDSR format keystores and HCR77B0, ICSF now has a date and time last used field in the record, so you can keep track of whether a key is being used. There is also new support for a start and end validity date. ICSF will honor the start date and not allow a key to be used prior to that date. Similarly it will not use a key after the end date has been reached.

However, none of the tools described above have been updated to use these new fields. And until there are key management tools available to query and act on these fields, it makes sense to continue to use the key label to manage keys.

Did You Know?

There is a lifecycle associated with keys … it's not just create and then use a key.

The National Institute of Standards and Technology have identified (in NIST Special Publication 800-57, Part 1 'Recommendation for Key Management – Part 1: General') six 'states' that a key might have during its lifetime:

Pre-Activation - key has been generated, but not yet in use
Active - key in use
Suspended - a key might be suspended because of a suspected compromise or because the owner of the key is 'out' (for example on leave) and thus not signing or authorizing crypto operations
Deactivated - no longer being used to protect data, but still available to decrypt data
Compromised - when a key has been accessed by an unauthorized entity; a compromised key should not be used to protect data going forward, but may still be used to decrypt data
Destroyed - the key no longer exists, although metadata about the key may still be available

Key management documentation from NIST

In addition to Part 1 of SP800-57, there is a Part 2 that covers Best Practices for Key Management Organization and Part 3 covers Application-Specific Key Management Guidance
NIST also provides documents on generating keys and building key management systems.

What Mainframe Crypto can provide

Getting started with crypto – If you're just starting to leverage the crypto infrastructure, we can provide guidance in configuring the environment as well as developing the processes and procedures to manage the infrastructure. We can provide guidance on the configuration settings that

might impact performance as well as security.

If you're already leveraging the crypto technology, maybe it was implemented some time ago and by a staff member that has moved on to new opportunities. We can help you review and document the configuration, and understand some of the decisions that were made in the initial implementation as well as determining whether those decisions are still appropriate.

If you're dealing with audits and questions from your security team, we can help you document the policies and processes that you are using to provide a secure environment. As appropriate, we can help you "tighten up" those processes to meet your audit requirements.

If you need to implement new crypto solutions we can help extend the current crypto environment to support the new products or applications. That may include evaluating the current workloads to ensure that you have the capacity to support the new workloads.

If you need help in implementing crypto within your internally developed applications, we can help with understanding and implementing the APIs.

NewEra zExchange

The zExchange, sponsored by NewEra Software, continues to offer the monthly crypto sessions in 2015. (Plus a whole lot of other valuable session on z Systems.) Already this year in the crypto sessions, we've covered the hardware through the new IBM z13 and ICSF through the latest release of ICSF, HCR77B0. The September session we reviewed crypto performance expectations.

The sessions are typically on the last Wednesday of the month at 1PM. You can enroll at http://www.newera-info.com/z-OS-Crypto.html.

This is the fourth issue of the Mainframe Crypto Newsletter. Its goal is to help you learn about new crypto technology and realize the full-function of the crypto technology that is available on IBM System z. You are receiving this newsletter because you either a) signed up for it on my website, www.mainframecrypto.com or b) you signed up for one of my Crypto Webcasts on the NewEra Software zExchange. Either way, Thank you!

If you would like to continue receiving this newsletter, please sign up at my website www.mainframecrypto.com (from any page except the home page look for "Get Greg's Newsletter"). I plan to phase out the use of the webcast list over time, so the only way to be sure to continue receiving this newsletter is to subscribe.