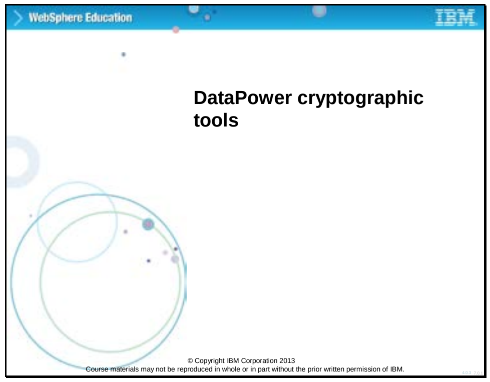
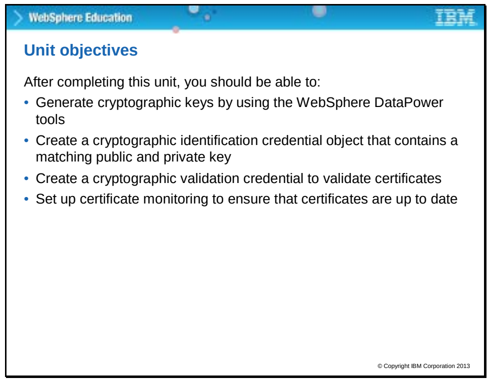
Slide 1

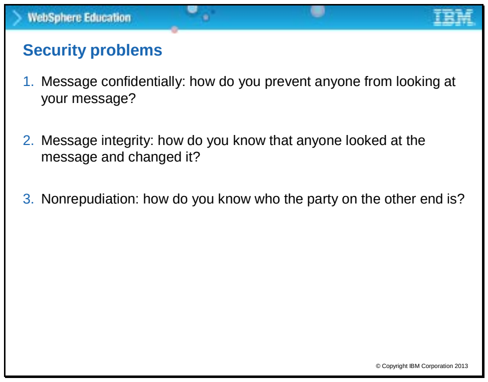

WebSphere Education
IBM
DataPower cryptographic tools
© Copyright IBM Corporation 2013
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Slide 2



This presentation talks about how to use the DataPower cryptographic tools to create public-private key pairs. In addition, students learn how to create cryptographic objects that are used in SSL, also actions such as sign, verify, encrypt, and decrypt.

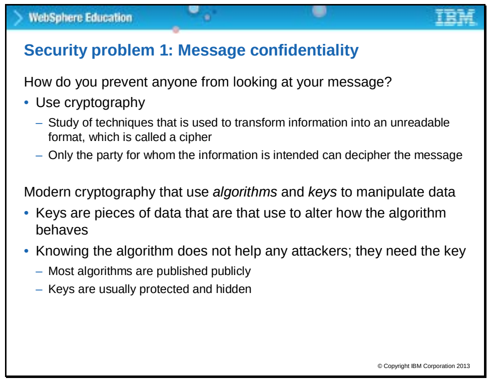
Slide 3



Security problems

What are the fundamental problems of security? This slide sums them up under three headings; confidentiality, integrity, and non-repudiation. This presentation looks at each of these points in turn.

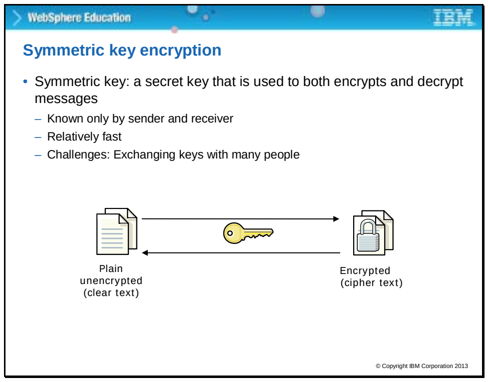
Slide 4



Security problem 1: Message confidentiality

The first problem is how to stop any third party from looking at your message. In fact, the problem is not really how to stop them looking; it is how to stop them understanding what they see. Cryptography, meaning secret writing or hidden writing, is the technique of making information unintelligible while it is in transit. There are two ways the technique can be used. First, two parties might agree on the meaning of a word or a phrase. So for example, “the tree is green” means “sell half the stock”. Obviously, while this system can have some applicability; it is far too restrictive in the variety of information that can be transmitted. It is typically used to code operations or development projects. The Manhattan Project is probably the best-known example. Some 35 years ago a different system of encryption was put forward. The project algorithms were seeded by a key. While there are different ways of doing the processing, one of the most widespread is the use of a public-private key pair. The algorithm is seeded by using the public key, and the encrypted result cannot be reverse-engineered. The only way back to the original document is by seeding the same algorithm with the private key. An example of seeding is covered in a couple of slides. First, view symmetric encryption.

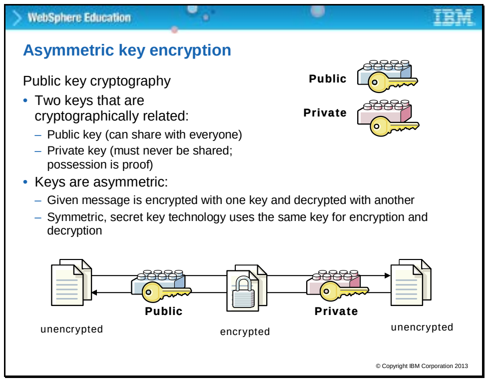
Slide 5



Symmetric key encryption

In symmetric key encryption, there is only one key that is used both for encryption and for decryption. Symmetric key encryption is like locking a box with a key and then unlocking it with the same key. To be effective, the key is secret, known only to the two sides of the communication path. The problem therefore is how to exchange the key over a channel that is open that is to say, not secure before the key is used.

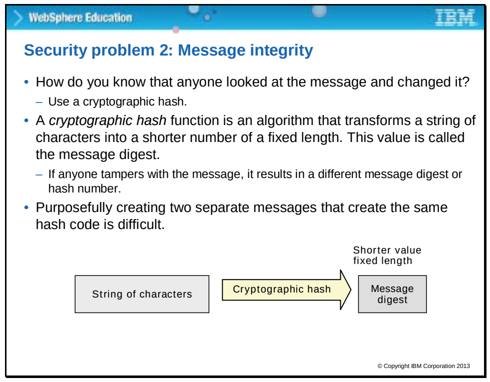
Slide 6



Asymmetric key encryption

In asymmetric key encryption, there are two keys, one to encrypt, and a different one to decrypt. There is an analogy here with the night deposit box at the bank. Each person who must deposit has a ‘public key’, one that opens the front door of the night deposit. The bank has a private key that opens the other side of the night deposit box. It does not matter that you have the same key as someone else. The only thing that you can do is deposit; you cannot remove anything through the front of the box. Likewise, the public key is freely available for encryption because when encrypted the message cannot be decrypted with the same key.

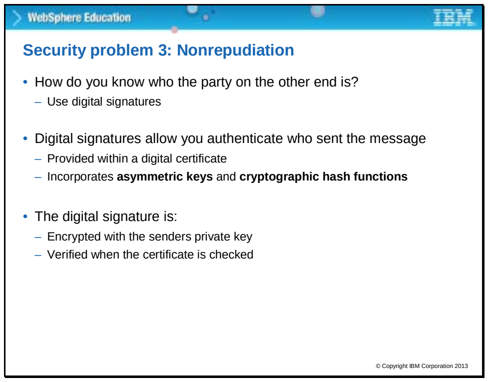
Slide 7



Security problem 2: Message integrity

The second problem that was stated at the beginning of this unit is the question of integrity. Is the message that you are receiving the same as the one that was sent? The process to verify that the message is the original message is to create a hash value for the message. You can think of a hash in this way: take each letter of a word, find the numeric position in the alphabet, and add the numbers. For example, if “John” is the word, the total is “J + O + H + N”, or “10 + 15 + 8 + 14“, which gives 47. If some hacker now modifies the message to pretend that “Joan” sent the message, the addition becomes “J + O + A + N”, or “10 + 15 + 1 + 14“, which gives 40. With a little thought, you can invent different names that add up to the same value, for example “Mary” and “George”. The actual hash algorithm is obviously more complex than the simple example shows. However, the principle is the same, which is why the third bullet does not state that it is impossible to create two different messages with the same hash value! But it is difficult.

Slide 8

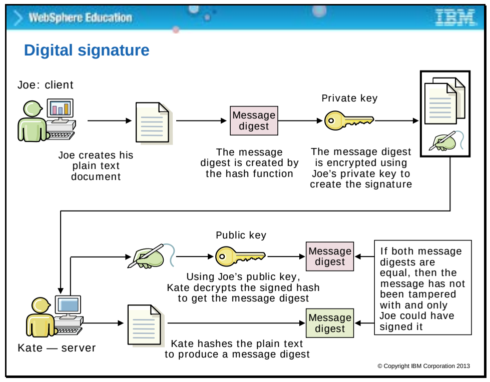


Security problem 3: Nonrepudiation

The third problem that is stated at the beginning was non-repudiation. I send you a message and then I deny sending it, how can you prove that it was from me? In fact, I do not have to deny sending it! You are able to prove that I sent the message for security reasons: a hacker might be trying to impersonate me. The solution to the impersonation is a digital signature. The digital signature is created by applying a private key to the message, or more usually, to the hash of the message. The validity of the message can then be verified by using the public key.

Take a closer look at the steps here.

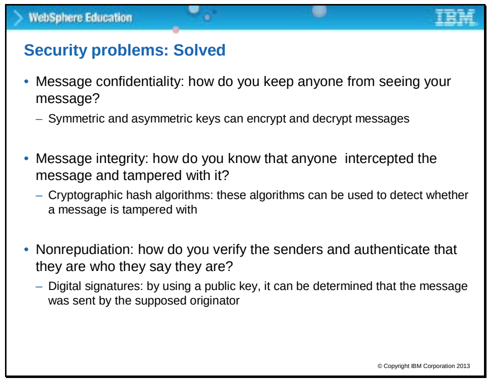
Slide 9



Digital signature

Joe, the client, creates a message digest from the document he intends to send to Kate, then applies his private key to it. Kate receives the message and does exactly what Joe did – she creates a message digest from the document by using a hash function. She must use the same hash function Joe that is used, or this does not work. She then takes Joe’s encrypted message digest and decrypts it using Joe’s public key. If the digests match, the message is authentic. What happens if a hacker intercepts the message? They can decrypt the digest, because they need the public key to do that. But here is the problem: if they change the message, the hash must be recalculated, and the recalculation can be done with Joe’s private key. Remember that asymmetric encryption is a one-way thing, you cannot decrypt and encrypt with the same key. Jane would therefore detect that the hash encrypted by Joe did not correspond to the message hash she created, and she would reject the message

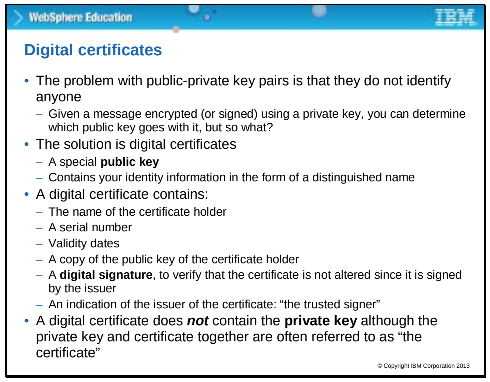
Slide 10



Security problems: Solved

In conclusion, to prevent anyone that understands your message, you encrypt it. Note again that the slide says “seeing”. Seeing refers to “seeing the clear text message”. To verify integrity you use a hash value of the original message, and to validate the sender, you create a digital signature by using the hash value.

Slide 11

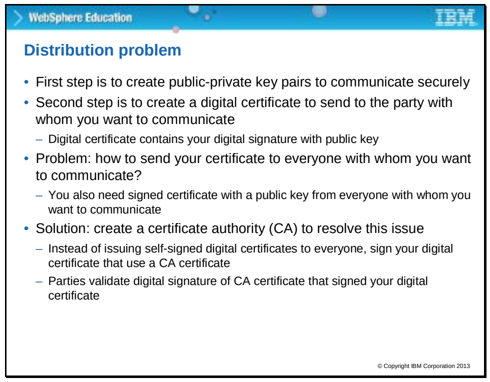


Digital certificates

The digital signature that is covered on the previous slide is used as a part of a complete object – the digital certificate. The thing here is that when you see a public key or a private key, there is no information that tells you who they belong to. The digital certificate wraps more information together that identifies the certificate holder. Information such as the name and a specific serial number (like an id number), and also the date from which the certificate is invalid. It also contains the public key. The first statement of the second bullet might be misleading, that the digital certificate is a 'special public key'. The certificate contains the public key; it is not a key in itself.

How do you know that the digital certificate itself is not just a hacker's invention? Good question! And now is where you reach the realm of trust: there must be some point where you can say "I trust this". A certificate authority provides the trust point. VeriSign is an example of a certificate authority. They take it upon themselves to validate a company, and they issue the certificate. When you receive the certificate, you can be sure about the identity of the sender because you can say "I trust VeriSign". The certificate is therefore self-signed – by the certificate authority. This signature is commonly known and can easily be verified.

Slide 12

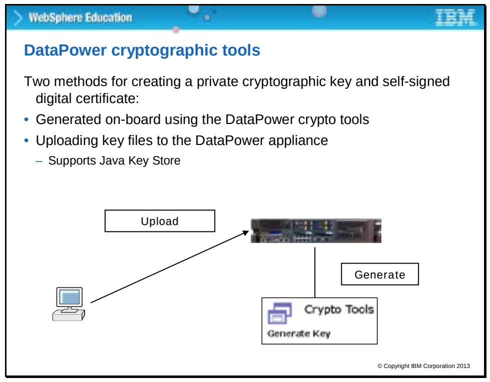


Distribution problem

What you saw on the previous slide is linked to the problem of distribution. Although you might distribute self-signed certificates (and in many cases the distribution is a good solution), it is much better to have a third party that everyone can trust. The certificate authority issues you a CA certificate, and you use the certificate to create your own digital certificates.

There are about 30 CA certificates that are shipped with DataPower. The certificates are known as root certificates.

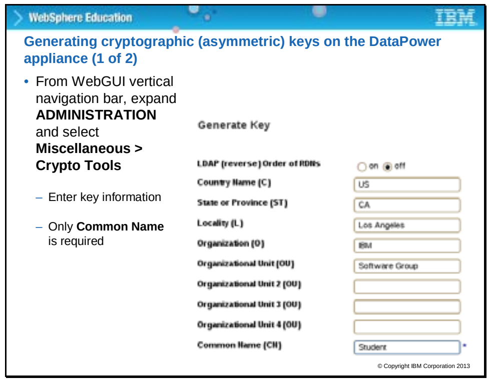
Slide 13



DataPower crypto tools

Self-signed certificates are often a suitable solution. For example, for communication within a company, there is usually less need for the complete assurance of a third-party CA certificate. DataPower has two ways to create self-signed certificates. First, you can use DataPower tools to create the certificate on-board. Second, you can upload keys to the appliance by using a Java keystore.

Slide 14

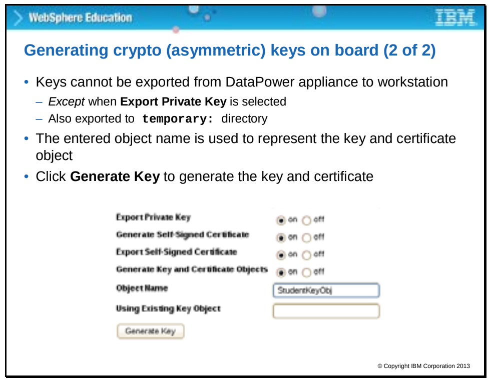


Generating crypto (asymmetric) keys on board (1 of 2)

Here is how to create asymmetric keys on DataPower. You expand the Administration section of the left navigation bar and select Crypto Tools under the Miscellaneous heading. On the dialog that opens there are three tabs – Generate Key, and Export and Import Crypto Object. The dialog is long, and so it is split over two slides. This slide shows the top half.

The most important thing you put in here is the common name, the name that identifies the key. For more precise identification, you can optionally include information that ranges from the country (the largest division) down to four indications of unit (the finest division). You cannot see the RSA key length on the slide: the field is off to the right – but you can specify a key length of 1024 bits, 2048 bits, or 4096 bits. Keys have validity periods, and by default the period is set to one year.

Slide 15



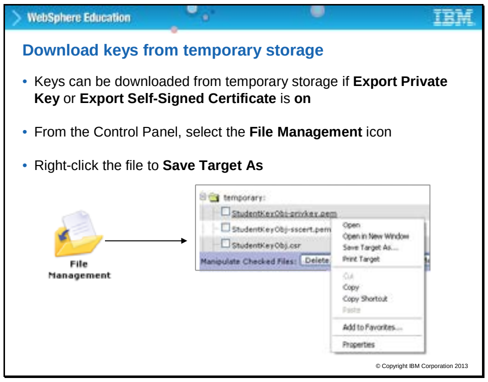
Generating crypto (asymmetric) keys on board (2 of 2)

You have one opportunity to export a private key from DataPower, at the moment you create it! First, you toggle the Export Private Key button to ‘on’ before you click the Generate Key button at the bottom of the dialog. This action writes a copy of the key to the temporary storage directory. You can then pick it up from there and export it. This is generally a good thing to do. You want to keep a copy of your certificates and private keys as backup in case of major problems with the appliance. If you do not turn on this option, then the key goes only to the cert directory. The cert directory is an encrypted directory; you do not have a choice about that action. When your key goes in there, you cannot retrieve it.

You might wonder how temporary is ‘temporary’? The copies sit in the temporary directory until the domain is reset or the appliance is recycled. Restarting the domain does not empty the temporary directory. It simply restarts the services.

At the same time, you can generate the self-signed certificate and export that. This process is in addition to creating the key and certificate objects that are associated with the files, and that you use in the DataPower services you create. You would in this case provide the name that identifies the object.

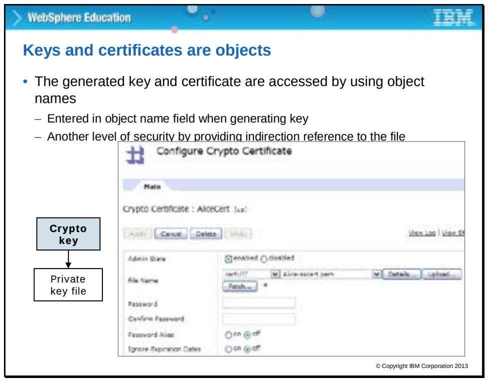
Slide 16



Download keys from temporary storage

When you click the button to generate the key, you get a dialog that asks you to confirm that you want a 1024-bit RSA key pair and a CSR generated. CSR is a certificate signing request that you would send off to your certificate authority. After generating the key, you go to the temporary directory, and (assuming that you asked to export the private key) you can right-click the private key file and select 'Save Target As...'. There are three files that are listed in the screen capture: the private key, the self-signed certificate, and the certificate signing request file. The file names are composed from the information you gave in the field 'object name'. You saw this one on the previous slide – together with the indication as to what the file represents, the private key or the self-signed certificate, or privkey and sscert. By the way, the name you can see on the slide is StudentKeyObj. If this name corresponded to the information that you saw on the previous slide, it would say AliceKeyObj.

Slide 17



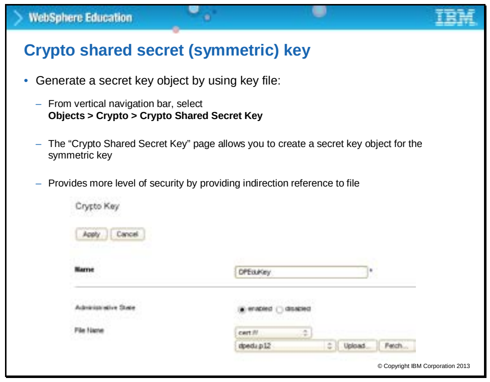
Keys and certificates are objects

You do not manipulate the key and certificate files directly, but rather the objects that point to them. You do not memorize file names that might be complex; you can give a key object name and a certificate object name that can be easy to remember. The name is the name that you enter in the object name field that you saw two slides ago. On this slide you can see a crypto certificate called AliceCert. This object points to the actual file called Alice-sscert.pem, in the cert directory. You can click the details button and examine things such as the distinguished name, or the validity dates. You cannot change the values!

Now is the second time that you are seeing the ‘Password Alias’ field. It was not discussed earlier, but now is an appropriate time to present some information about this field. You can set up a password for a key, and each key might have a different password. You can then set up a password alias that points to the actual password. You might use the same password alias for different keys. You do not require an alias if you set up the password.

You can choose to ignore any expiration date that is in the key. Normally, if the key or certificate is out of date there is a warning about compromised validity. You might want to be able to use the objects without these warnings.

Slide 18

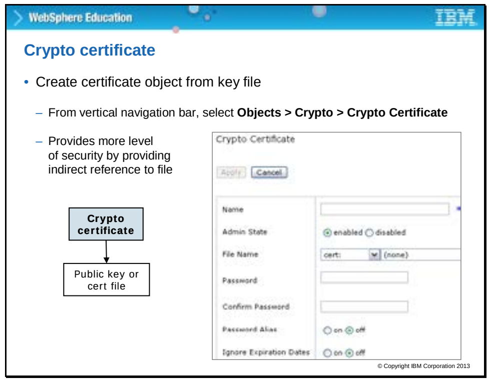


Crypto shared secret (symmetric) key

The shared secret key is another term for a symmetric key. Remember that the key is a key that is identical for the sender and the receiver. Whereas the asymmetric key is a technology that was invented relatively recently (around 35 years ago), the symmetric key technique is known and used for a long time, possibly even thousands of years. The problem is how to distribute the key in a secure manner. It also requires a different key for each sender-receiver pair.

There is no tool on DataPower to generate symmetric keys. It generates asymmetric keys. You would use some other external tool to generate the key, and then upload it to DataPower.

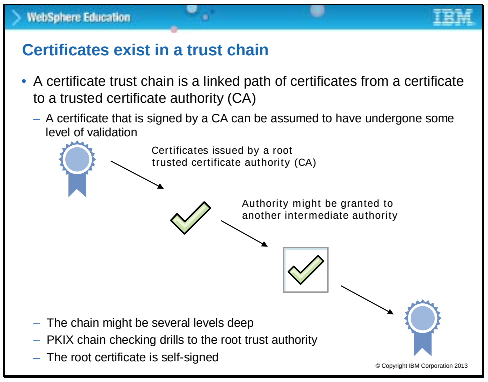
Slide 19



Crypto certificate

The crypto certificate object provides the indirect reference to the certificate file. You give the object a name. Then, choose the file to which to object is making reference. You can add a password if you want, and you can choose to ignore the validity dates to avoid warning messages.

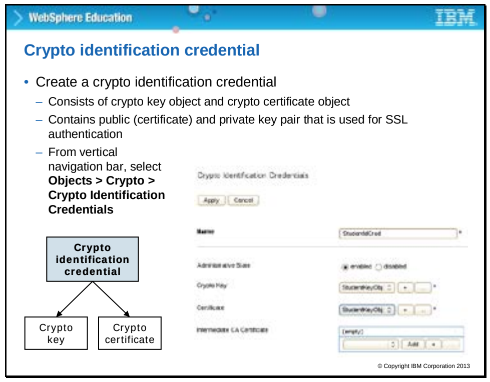
Slide 20



Certificates exist in a trust chain

The certificate that you use might not itself be issued by a certificate authority. For example, an enterprise might go to a trusted authority such as VeriSign to get a root certificate, and then use that as the basis for internally produced certificates. Thus the certificate that the customer might be using is signed by the enterprise, not by VeriSign. The trusted authority is known as an intermediate authority. This might end up being a series of certificates that are known as a chain of trust. Here VeriSign certifies the root certificate, that one is used to certify the enterprise certificate, that on is used to certify the certificate of the software department of the enterprise, and so on. There is a thing that is called the PKIX chain, or Public Key Infrastructure X.509 chain, which can be used to recursively drill down to the original certificate. The original certificate, issued by the root trust authority, is self-signed. It represents the point at which trust is assumed; in other words, do you trust VeriSign (or any other certificate authority) when they say you should trust a specific certificate?
As was mentioned earlier, DataPower ships with about 30 certificates that are signed by IBM. VeriSign vouches for IBM, and IBM vouches for the certificates on DataPower, so do you trust those certificates? The answer is a business decision. For many enterprises this trust would not be a problem. Some military clients systematically delete all the certificates on DataPower when they get the appliances; they then load their own certificates that they are confident are valid.

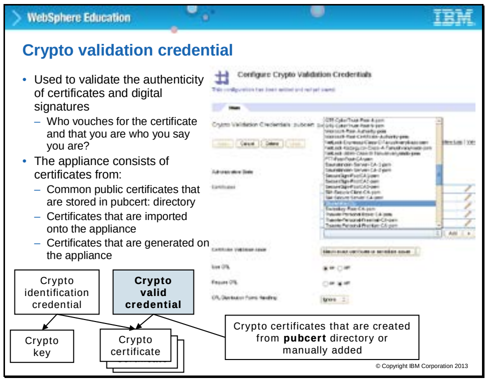
Slide 21



Crypto identification credential

This presentation has been about the lowest level of objects: the key object, and the certificate object. There is another level of object above the current objects, the crypto identification credential, which points to two things: the crypto key, which is the private key, and the crypto certificate, which contains the public key that is associated with the private key. You can think of the crypto identification credential as the object that groups together related keys. The screen capture shows how it is defined, by pointing not to the files themselves but to the objects that point to the files

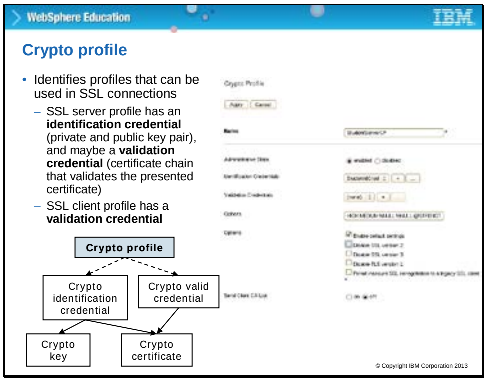
Slide 22



Crypto validation credential

If a certificate is presented to DataPower during some request from a client, how is DataPower going to validate it? There are two options. First, DataPower can match the exact certificate. In this case, the certificate is been presented must be available on the appliance. Second, a complete check might be required, by using a PKIX chain back to the root certificate. This process can be defined in a validation credential. This credential does not have anything to do with keys; it is concerned with certificates. It defines which certificates can be used to validate a presented one. You can also decide whether you want to use a CRL, or Certificate Revocation List. There is an opportunity to take a closer look at this list in a few slides.

Slide 23



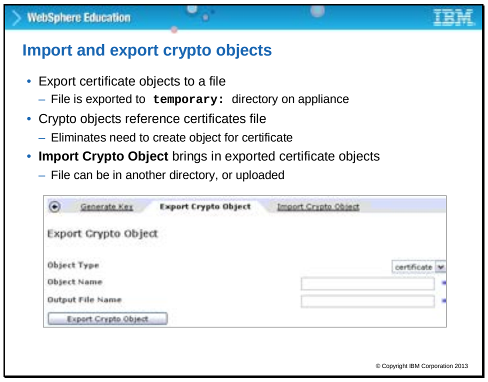
Crypto profile

There is yet another level above the crypto identification credential and the crypto validation credential: the crypto profile object. The object points either to a crypto identification credential object, or to a crypto validation credential object, or to both. It depends on the situation. There are two types of profile, depending on which direction you are coming from. The server profile says that the DataPower appliance is acting as a server; some client is calling it. The client profile says that the DataPower appliance is acting as a client; it is calling another server.

The crypto profile that is shown in this slide is used by the DataPower appliance during an SSL connection when the appliance is acting as the SSL server. Since no validation credential object is specified, it implies that the DataPower appliance is not requesting a certificate from the client.

You come back to the crypto profile when you look at SSL in the next unit.

Slide 24

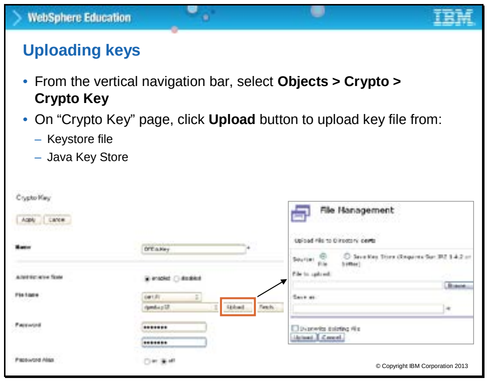


Import and export crypto objects

A few slides back you looked at generating asymmetric keys. You might remember that there were three tabs: generate keys, export crypto object, and import crypto object. This slide looks at these last two tabs.

The dialogs are simple, and similar. Export places a copy of the exported file in the temporary directory. You can then open the directory and select to save the file. An import can be from a remote location, and so there is a button that allows you to upload files.

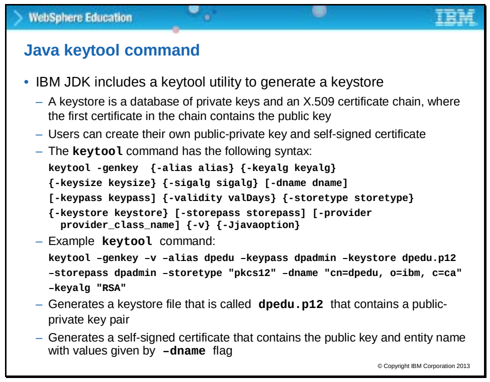
Slide 25



Uploading keys

You might have keys that are generated in a different environment that you upload to the DataPower appliance. You would do the upload for example if you generated keys by using the Java keystore. This Java keystone is a Java security object that holds keys and certificates. From the crypto key page, you would click the upload button and browse to where ever you place the file.

Slide 26

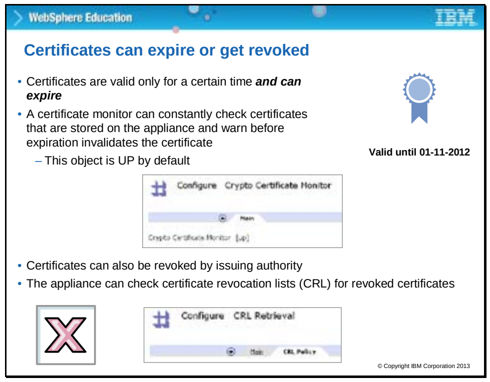


Java keytool command

The syntax is an example of a command sequence that generates a keystore. The Sun JDK has a keytool utility, and also the IBM JDK. The syntax is shown in the middle of the slide, and a typical command sequence at the bottom. Notice that the file name is defined by using the –keystore directive. The distinguished name is created with three values, common name, organization and country, and the algorithm is used to generate the key is the RSA algorithm.

You saw RSA earlier. The presentation does not cover security algorithms. Just as an side note, you might be interested to learn that the security algorithm RSA is the initials of the three people who developed the first algorithm that might be used for asymmetric encryption, Rivest, Shamir, and Alderman.

Slide 27



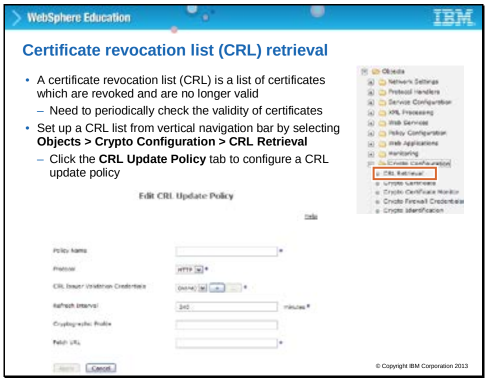
Certificates can expire or get revoked

A few slides back you saw mention of a CRL. Here is some more discussion of what a CRL is.

Certificates typically are only valid for a limited period. From the DataPower perspective, you would not want to suddenly find yourself with execution problems because an essential certificate expired yesterday! It can be that your server environment becomes inaccessible because certificates are no longer valid, which means that you would suddenly stop doing business. DataPower therefore has a certificate monitor that constantly checks expiration dates of all the certificates that are stored on the appliance. This monitor function gives timely warning to the administrator, for example, starting one month before the expiration date. Some slides later in this presentation provide a closer look at how CRLs are set up.

The same principle holds for certificates that other environments present to DataPower. Maybe a company ceased to exist and its certificates are no longer used, or maybe it is discovered that a certificate was set up fraudulently. How can the appliance know whether they are still valid? It determines their validity by consulting a Certificate Revocation List, or CRL.

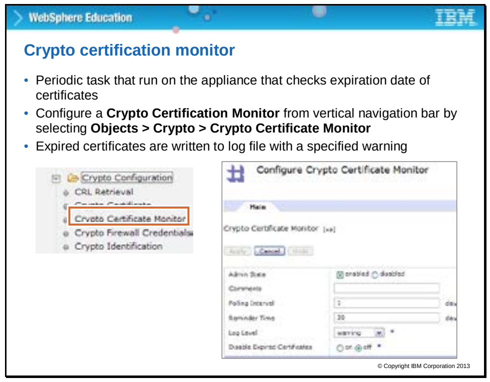
Slide 28



Certificate revocation list (CRL) retrieval

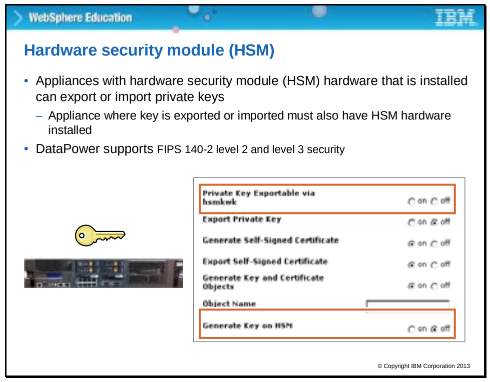
Here is the dialog for setting up a CRL, which is only available from the default domain, and you need administrative privileges to add or modify a policy. You give the retrieval policy a name. Then, indicate whether you retrieve the list from an LDAP server or some other place by selecting the appropriate protocol. You set up one to go to VeriSign, one to go to another authority. Where ever you are getting your certificates from. You can set up a refresh interval according to how often you think you must get a new revocation list.

Slide 29



Crypto certification monitor
Here is the dialog to set up the certificate monitor, which periodically checks all your certificates for expiration. You set up the polling interval, the reminder time, and the log level. The polling interval is typically one day because expiry cannot be set to smaller time intervals than the day. You can set expiry to be on April 1, but not on April 1 at 11 in the morning! Reminder time indicates how long before the expiry date you want to start getting warned. You get warnings every day from this point, since if you do not renew a certificate. The next day the monitor is going to still see that it is going to expire and will report it. The log level can be set to give a specific level of logging of the impending expiry. Finally, you can decide what to do with certificates that expired. Do you want to allow them to be used, while the warnings continue to be issued, or do you want them disabled until someone does something about it?

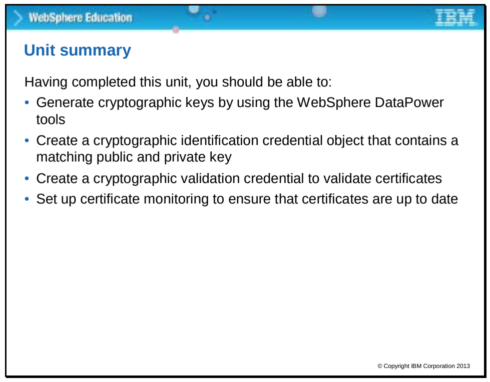
Slide 30



Hardware security module (HSM)

When you purchase an appliance, you can request that an extra module called the hardware security module. The security module provides some extra flexibility that regards key storage on the appliance. If you have this module installed, there are some extra fields that are added to the dialog for generating keys that allow you to specify that you want the key that is generated on the HSM. The process allows you to export it at any time, not just at key generation time.

Slide 31


WebSphere Education
IBM
Unit summary
Having completed this unit, you should be able to:
• Generate cryptographic keys by using the WebSphere DataPower tools
• Create a cryptographic identification credential object that contains a matching public and private key
• Create a cryptographic validation credential to validate certificates
• Set up certificate monitoring to ensure that certificates are up to date
© Copyright IBM Corporation 2013

Slide 32

Checkpoint questions

1. Match the corresponding asymmetric and symmetric key encryptions:

Description	Definition
1. Symmetric key	A. Uses two different keys for encrypting and decrypting
2. Asymmetric key	B. Uses the same key

2. What does a digital certificate contain?
   - A. Your name, a serial number, expiration dates, a copy of the certificate holders public key, a digital signature
   - B. Your name, credit card number, expiration date, pin code
   - C. Digitalized company certificate with verification turned on

3. True or False: Keys that are generated on-board cannot be exported.

Slide 33

Checkpoint answers

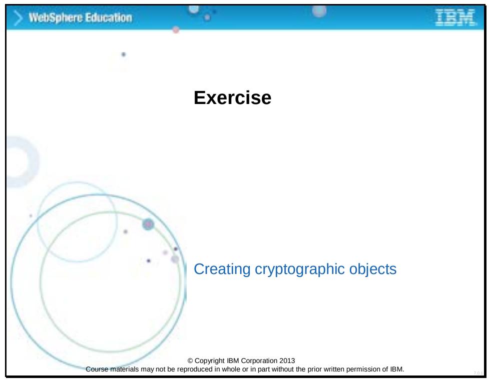
1. Match the corresponding asymmetric and symmetric key encryptions:

Description	Definition
1. Symmetric key	A. Uses two different keys for encrypting and decrypting
2. Asymmetric key	B. Uses the same key

2. A. What does a digital certificate contain?
   - ✓ A. Your name, a serial number, expiration dates, a copy of the certificate holders public key, a digital signature
   - B. Your name, credit card number, expiration date, pin code
   - C. Digitalized company certificate with verification turned on

3. False. Keys can be exported to the **temporary:** directory if the **Export Private Key** radio button is selected when generating a key on the appliance.

Slide 34


WebSphere Education
Exercise
Creating cryptographic objects
© Copyright IBM Corporation 2013
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Slide 35

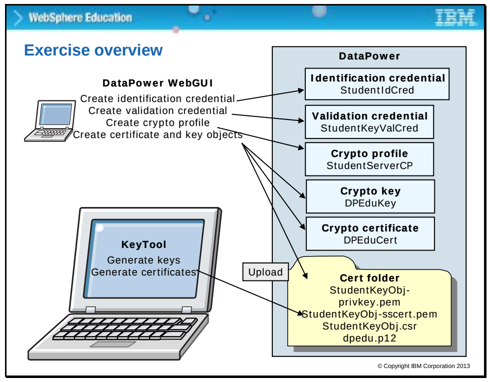
WebSphere Education IBM

Exercise objectives

After completing this exercise, you should be able to:

- Generate cryptographic keys by using the WebSphere DataPower cryptographic tools
- Upload key files to the WebSphere DataPower SOA Appliance
- Create a cryptographic identification credential that use a cryptographic key object
- Validate certificates that use a validation credential object

Slide 36



WebSphere Education
IBM
Exercise overview
DataPower WebGUI
Create identification credential
Create validation credential
Create crypto profile
Create certificate and key objects
DataPower
Identification credential
StudentIdCred
Validation credential
StudentKeyValCred
Crypto profile
StudentServerCP
Crypto key
DPEduKey
Crypto certificate
DPEduCert
KeyTool
Generate keys
Generate certificates
Upload
Cert folder
StudentKeyObj-privkey.pem
StudentKeyObj-sscert.pem
StudentKeyObj.csr
dpedu.p12
© Copyright IBM Corporation 2013