

DJIGZO EMAIL ENCRYPTION

---

# Djigzo Frequently Asked Questions

---



*Author:* Martijn BRINKERS

May 3, 2010, Rev: 4250



# Contents

<b>FAQ</b>	<b>4</b>
<b>S/MIME</b>	<b>4</b>
What exactly is a certificate? . . . . .	4
What is a root certificate? . . . . .	4
What is an intermediate certificate? . . . . .	5
What is an end-user certificate? . . . . .	5
How can a certificate have ‘child’ certificates? Do ‘child’ certificates share the same public key as the parent? What about the private key? . . . . .	5
Why do certificates expire? How does Djigzo react when it is called upon to do encryption with a recently expired certificate? . . . .	5
What is the difference between a signature and an encryption certificate? Do they have to be different? Should we always sign outbound encrypted emails? . . . . .	6
Should we have a self-signed root certificate or one signed by a certificate authority (CA)? If the latter, then which CA should we use and what should we ask for exactly? What due diligence will the CA perform? How much will it cost? . . . . .	6
What does it mean when a certificate is revoked? Who is able to revoke a certificate? Why is it done? Are we supposed to download a revoked list from our certificate authority? . . . . .	7
What is a certificate Trust List (CTL)? . . . . .	7
Should I use SHA-1 or SHA-256 for CRLs and certificates? . . . . .	7
<b>PDF encryption</b>	<b>8</b>
Djigzo supports two different encryption methods: PDF and S/MIME. S/MIME sounds very logical, but PDF is a strange method. Why did you add this? . . . . .	8
With PDF encryption, are attachments encrypted as well? . . . . .	8
With PDF encryption, how can the recipient reply encrypted? . . . . .	8
<b>Gateway</b>	<b>8</b>
What are the default login credentials? . . . . .	8
Why do I need to install Java Unlimited Strength Jurisdiction Policy Files? . . . . .	9
Djigzo comes with no certificates installed. Why is that? . . . . .	9
What is the root password of the Virtual appliance? . . . . .	9
Incoming encrypted email is not decrypted? . . . . .	9
What is the difference between the Djigzo engine and the Djigzo web admin? . . . . .	9
If I try to login immediately after starting Djigzo I get an exception. Why is that? . . . . .	10
Is it possible to store the administrator credentials and roles on an external LDAP? . . . . .	10
Where should a Djigzo gateway be placed? . . . . .	10
<b>Djigzo for BlackBerry</b>	<b>11</b>

Which BlackBerry smartphones are supported? . . . . .	11
Why is a Djigzo gateway required? . . . . .	11
Can Djigzo for BlackBerry be used with BIS? . . . . .	11
Can all email sent to my BlackBerry be encrypted? . . . . .	11
Can Djigzo for BlackBerry handle email which is S/MIME encrypted by Outlook? . . . . .	11
Should a copy of the private key be available on the Djigzo gateway? .	11
Is Djigzo for BlackBerry compatible with the BlackBerry smartcard reader? . . . . .	12
Are email attachments supported by Djigzo for BlackBerry? . . . . .	12
is HTML email supported by Djigzo for BlackBerry? . . . . .	12
Does Djigzo for BlackBerry validate signatures? . . . . .	12
Why is email sent with Djigzo for BlackBerry relayed via a Djigzo gateway? . . . . .	12
Can I prevent the user from sending non-protected email? . . . . .	12
I always need to enter the key store password when I open a message. Why is that? . . . . .	13
<b>General</b>	<b>13</b>
Suppose we issue a client a private key and the client loses it. What should we do? . . . . .	13
Suppose we issue a client a private key and our relationship with that client ends. What should we do? . . . . .	13
Suppose we issue a client a private key and that key is stolen from the client's computer. What's the worse that can happen? What should we do on our end if we suspect a client's key has been stolen? . . . . .	14
Suppose our Djigzo server becomes compromised. What's the worse case scenario? How should we react? . . . . .	14
If we get a root certificate signed by a CA, can we generate unlimited end-user certificates from that without any intermediates? Is this a sound practice? . . . . .	15
When selecting a password to protect private keys sent to clients, should a unique one be used each and every time? . . . . .	15
What admin roles should customer service personnel be granted? . . .	15
When issuing end-user certificates, a CA email is supposed to be spec- ified. Who should see the mail associated with this email address?	15
Where can I get certificates? . . . . .	15

## FAQ

### S/MIME

#### What exactly is a certificate?

S/MIME uses Public Key Infrastructure (PKI) to securely exchange information over insecure networks using public key cryptography. Public key cryptography makes use of two keys: the *public key* and the *private key*. The private key should be kept secret whereas the public key can be made available to everyone. The public key is used for encrypting and the associated private key is used for decrypting. If a sender wants to send an encrypted message, the sender needs to get hold of the correct public key of the recipient.

An X.509 certificate contains the public key of the certificate owner. The sender needs to select the correct certificate for encryption. The certificate therefore contains some extra information which can be used to identify the correct certificate. A certificate used for S/MIME should at least contain an email address for which the certificate was issued. Most certificates however contain more information like name, company etc.

A certificate in essence is just a binary file with a specific format. With the correct tools, a certificate with any content can be easily made. A certificate can therefore not be automatically trusted because the content, i.e. the identifying information about the certificate owner, cannot be trusted.

There are two ways a sender can decide to trust the certificate: explicit trust or implicit trust using PKI. By explicitly trusting a certificate, the sender decides that this specific certificate is valid and trusted. For example the sender has checked the certificate thumbprint over the phone. The main disadvantage of explicitly trusting certificates is that every certificate must be manually checked and placed on the *Certificate Trust List* (CTL).

Using PKI, an automatic procedure can be used to implicitly check the trust of a certificate. S/MIME uses a hierarchical trust model where trust is inferred bottom-up. The root (the bottom) is blindly trusted (which makes it by definition a root) and all leaf nodes and branches (the end-user and intermediate certificates) are trusted because they are ‘child’s’ of the trusted root (to be precise the intermediate certificates are issued by the root certificate). The main advantage of using the PKI trust model (i.e. implicit trust) is that only the root certificate has to be explicitly trusted. All certificates issued by the root are automatically trusted. A certificate is issued by a CA, which can be a root or an intermediate CA. Because the CA digitally signs a certificate, end-users can check whether the certificate was issued by a trusted root.

Besides identifying information about the owner of a certificate, most certificates contain more information like expiration date, certificate usage etc. For example if the extended key usage is set it should contain at least *anyKeyUsage* or *emailProtection* to make the certificate valid for S/MIME.

#### What is a root certificate?

S/MIME uses a hierarchical trust model. A certificate can be validated by checking whether the certificate was digitally signed by the issuer of the certificate. The issuer certificate itself can be issued by another certificate (which

makes it an intermediate certificate). This procedure should be repeated until a root certificate is reached. Because a root certificate is explicitly trusted and a valid chain can be built from a certificate to a root certificate the certificate is implicitly trusted. Because a root is 'blindly' trusted only root certificates that are really trusted should be added to the root store.

### **What is an intermediate certificate?**

An intermediate certificate is a CA certificate used for issuing other certificates. Sometimes a CA uses multiple intermediate certificates and each intermediate certificate is used for different purposes. For example one intermediate certificate is used for issuing certificates only valid for S/MIME and another one for only SSL. Only certificates with a CA basic constraints extension are allowed to issue other certificates.

### **What is an end-user certificate?**

An end-user certificate is a certificate which should be used for encryption or signing. An end-user certificate for S/MIME should at least contain the email address of the owner. An end-user certificate should not be used for issuing other certificates.

### **How can a certificate have 'child' certificates? Do 'child' certificates share the same public key as the parent? What about the private key?**

A root and intermediate certificate are used for issuing other certificates. The certificates issued by a root and intermediate certificate are 'child' certificates of the root and intermediate.

Every certificate has its own public key. Public and private keys are not shared. A private key is not part of a certificate. When a certificate request is created, a private key and public key is generated. The public key is added to the certificate. Although the certificate and private key are associated, they are separate entities. For transport, a certificate and private key are often stored in a password protected *.pfx* file.

### **Why do certificates expire? How does Djigzo react when it is called upon to do encryption with a recently expired certificate?**

There are different reasons why certificates have an expiration date. For example when a root certificate is created the protection level of the certificate (for example the length of the public key) is set to a level that should be strong enough to last the lifetime of the certificate. Because of increasing computer power the level of protection should also be increased over time. By making sure a certificate expires before the level of protection falls below an acceptable level, the certificate protection level is always strong enough. Another reason why an end-user certificate has an expiration date is that when a certificate should no longer be used, because for example an employee has left, the certificate automatically expires after some time.

Commercial certificate issuers have a business reason why certificates expire (most of them are only valid for one year). They ensure themselves of a constant income when they only issue certificates that expire within one year.

Djigzo won't automatically use an expired certificate. If an expired certificate should be used by Djigzo because the administrator has determined that the certificate can still be used, the certificate should be "white-listed" by adding the certificate to the CTL. Because issuing new certificates is cumbersome certificates should not expire too soon. We therefore advise to make end-user certificates valid for 5 years.

### **What is the difference between a signature and an encryption certificate? Do they have to be different? Should we always sign outbound encrypted emails?**

A certificate can contain a *key usage* extension which restricts the certificate usage. For S/MIME encryption, if a key usage is specified it should at least contain *keyEncipherment*. For S/MIME signing, if a key usage is specified it should at least contain *digitalSignature* or *nonRepudiation*. With a different certificate for encryption and signing, signing of messages can be done on the senders email client and decryption can be done on the encryption gateway. Sometimes policies require signing of messages with a smartcard on the desktop because the signature is considered a legally binding signature. If signing of messages on the senders email client is not required there is no need to use a separate signing and encryption certificate.

Signing a message provides you with authentication and message integrity. The receiver can check who wrote the message (authentication) and whether the message was changed after signing.

### **Should we have a self-signed root certificate or one signed by a certificate authority (CA)? If the latter, then which CA should we use and what should we ask for exactly? What due diligence will the CA perform? How much will it cost?**

The problem with using your own root certificate is that the root certificate is not automatically trusted by external users. Every external user should import the root certificate into the root certificate store. Getting an intermediate CA certificate issued by a universally trusted root has the advantage that all issued certificates are automatically trusted. The main disadvantage however is that this is expensive. Most CAs do not provide the private key, they only allow you to issue certificates via an externally accessible portal. Djigzo currently does not interface directly with external CA providers. Certificates from external CAs should therefore be manually imported. We do have plans however to directly support external CAs. This allows you to automatically create certificates issued by an external CA. Some widely known commercial certificate issuers are: Verisign, Comodo, GlobalSign and StartSSL. The costs of having your own CA issued by a trusted root depends on a lot of details like volume etc.

## What does it mean when a certificate is revoked? Who is able to revoke a certificate? Why is it done? Are we supposed to download a revoked list from our certificate authority?

Sometimes a certificate should no longer be used. For example the *private key* has been compromised or an employee has left the company. Certificates can be revoked by putting the certificates on a Certificate Revocation List (CRL). A CRL is issued and signed by the certificate authority (CA) that issued the certificate. A CRL is periodically updated. Most certificates contain a “CRL distribution point” with the URL from which the CRL can be downloaded. Djigzo periodically downloads all the CRLs from all the CRL distribution points of all certificates in the certificate store. Newly downloaded CRLs replace the previously downloaded CRLs. CRLs can also be manually added to the CRL store. However, this is only needed when a CA issues a CRL but the certificate does not contain a CRL distribution point.

## What is a certificate Trust List (CTL)?

A Certificate Trust List (CTL) is a list of “white-listed” and “black-listed” certificates. A CTL is created and updated by the gateway administrator and can contain certificates from different issuers. In most situations trust management using PKI will be sufficient. Sometimes however, the administrator needs more control over this automatic process. Some examples when a CTL can be helpful:

- a) A certificate should no longer be used because it was compromised. The certificate issuer however does not publish a CRL. By *black-listing* the certificate the certificate will no longer be valid.
- b) A certificate is not valid because the root is missing. The administrator however knows that the certificate is valid (for example the thumbprint has been checked over the phone). By *white-listing* the certificate the certificate will be valid for encryption.
- c) A certificate is not valid because the certificate has expired. However, the administrator is 100% certain that the certificate is still ‘valid’. By *white-listing* the certificate the certificate will be valid for encryption.

Using the CTLs white-list, trust can be managed on a more ad hoc bases. This is similar to how trust is managed with PGP. Instead of ‘inheriting’ trust from other trusted certificates, with a non-PKI approach using white-listing every certificate should be explicitly trusted.

## Should I use SHA-1 or SHA-256 for CRLs and certificates?

SHA-1 and SHA-256 are hash algorithms used for digital signing. SHA-256 is more secure than SHA-1 and should therefore be preferred over SHA-1. However, older versions of Windows do not support SHA-256. SHA-256 is supported on Windows XP SP3 (service pack 3) and newer versions of Windows. If older versions of Windows should be supported SHA-1 should be used for certificates and CRLs.

## PDF encryption

**Djigzo supports two different encryption methods: PDF and S/MIME. S/MIME sounds very logical, but PDF is a strange method. Why did you add this?**

The problem with S/MIME (or PGP for that matter) is that the recipient needs an S/MIME capable email client (most email clients however support S/MIME so this is not really a problem) and a certificate with private key. Even though importing a certificate with a private key is easy, some recipients might find importing a *.pfx* file too problematic. If the sender and recipient only need to exchange an encrypted email once or only a few times over a longer period, installing a certificate and key might be more trouble than it is worth. To accommodate for those situations a PDF encryption module has been added to Djigzo.

When a message is PDF encrypted the message, including all attachments, is converted to a password encrypted PDF file. The encrypted PDF is then sent to the recipient. The password for the PDF can be manually set by the administrator or can be randomly generated. The randomly generated password can be sent to the recipient via an SMS Text message (using the built-in SMS gateway) or can be sent back to the sender of the message (the sender should then securely transmit the password to the recipient). By using a separate channel for sending passwords, PDF encryption is almost as secure as full S/MIME encryption (provided that the password is long enough to withstand a brute force attack). PDF encryption is straightforward from an end-users perspective because a standard PDF reader can be used to open the encrypted message.

**With PDF encryption, are attachments encrypted as well?**

All attachments are added to the PDF document before the PDF is encrypted. The attachments are therefore encrypted as well.

**With PDF encryption, how can the recipient reply encrypted?**

The encrypted PDF document contains a reply link (only if the gateway enables the reply functionality). When the recipient clicks on the reply link, the recipients web browser opens an online page (running on a Djigzo gateway) on which the recipient can write the reply message.

## Gateway

**What are the default login credentials?**

**For the Web Admin:**

user: admin  
password: admin

### **For the Virtual Appliance console and SSH:**

user:      djigzo-admin  
password:  djigzo

### **Why do I need to install Java Unlimited Strength Jurisdiction Policy Files?**

Due to import control restrictions by the governments of a few countries, the jurisdiction policy files shipped with Java SE from Sun Microsystems specify that “strong” but limited cryptography may be used. An “unlimited strength” version of these files indicating no restrictions on cryptographic strengths is available for those living in eligible countries (which is most countries). Recent versions of OpenJDK (for example OpenJDK that ships with Red Hat 5.4) no longer limit the encryption strength. Installation of the “unlimited strength jurisdiction policy files” is therefore not required in those cases. The “unlimited strength jurisdiction policy files” should only be installed if a warning is shown in the Djigzo admin page.

### **Djigzo comes with no certificates installed. Why is that?**

Which certificate authorities should be trusted by the gateway is something the administrator should decide. Djigzo therefore does not come pre-configured with any root certificates. From Djigzo’s website two *.p7b* files can be downloaded: one with a collection of root authorities and one with a collection of intermediate authorities. These are just a collection of some of the mostly used CAs.

### **What is the root password of the Virtual appliance?**

The Virtual Appliance uses Ubuntu Linux 8.04 LTS which is maintained until April 2013. By default Ubuntu does allow the root to login. *sudo* should be used for all commands requiring root access.

### **Incoming encrypted email is not decrypted?**

Make sure that domains for which you receive email have been marked as internal domains. If not, Djigzo tries to encrypt the message because it thinks the messages is sent to an external domain. Also make sure that a correct certificate with private key is available.

### **What is the difference between the Djigzo engine and the Djigzo web admin?**

The Djigzo engine is the back-end part that encrypts and decrypts the email, sends SMS Text messages, stores certificates and keys etc. Djigzo web admin is the web application used to control the engine (the front-end part). The web admin uses SOAP calls to control the engine. It is therefore possible to install the engine and the web admin on separate servers.

## If I try to login immediately after starting Djigzo I get an exception. Why is that?

The back- and front-end are started separately. Starting and initializing the two processes takes some time. If the front-end startup process was finished and the admin tries to login before the back-end startup process was finished, an error will be shown. This only happens during startup when the admin tries to login before the back-end process has finished initializing. The admin should try to re-login after a few seconds.

## Is it possible to store the administrator credentials and roles on an external LDAP?

See the Web LDAP authentication guide for instructions on how to authenticate against an LDAP server.

## Where should a Djigzo gateway be placed?

The most typical setup is where a Djigzo gateway is placed between the Internet and the internal server. If a gateway level virus scanner is used, Djigzo should be placed between the Internet and the virus scanner (see figure 1). This ensures that the virus scanner never sees any encrypted email.

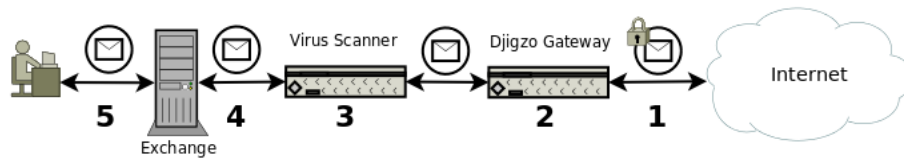


Figure 1: Djigzo with Virus Scanner

Djigzo can also be integrated with any existing content scanner (see Figure 2). With a content scanner, encryption can be forced based on message content (for example when the message contains a Social Security Number)

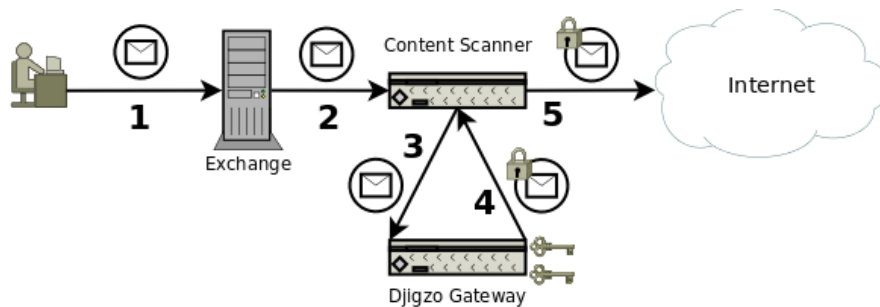


Figure 2: Djigzo with Content Scanner

## Djigzo for BlackBerry

### Which BlackBerry smartphones are supported?

All BlackBerry smartphones are supported including touch screen devices. The only requirement is BlackBerry OS  $\geq$  4.5 (contact Us if support for lower OS versions is required).

### Why is a Djigzo gateway required?

The BlackBerry infrastructure blocks access to certain attachments. When a message is S/MIME encrypted, the complete message including attachments is encrypted and the encrypted binary “blob” is then attached to a new message. The message with the S/MIME attachment then gets an S/MIME specific “Content-Type” to indicate that the message is an S/MIME message.

The problem is that the BlackBerry infrastructure blocks the S/MIME attachment. By rewriting the message headers (for example the “Content-Type” header) the S/MIME message will no longer be blocked by the BlackBerry infrastructure.

Strictly speaking a Djigzo gateway is not required. Any email server which is capable of rewriting certain message headers can be used instead. For more background information on the required header rewriting see the *Djigzo for BlackBerry Reference Guide*.

### Can Djigzo for BlackBerry be used with BIS?

Yes Djigzo for BlackBerry is BIS compatible.

### Can all email sent to my BlackBerry be encrypted?

Yes that’s possible. If *Encrypt Mode* for the BIS email address is set to *Force* and a certificate is available for the BIS email address then all email sent to the BIS email address will be encrypted.

### Can Djigzo for BlackBerry handle email which is S/MIME encrypted by Outlook?

Yes. Djigzo for BlackBerry is S/MIME compatible and can therefore handle email encrypted by any S/MIME compliant email client (like Outlook, Lotus Notes etc.) as long as the private key for decryption is available on the BlackBerry smartphone. Email encrypted with Outlook (or any other S/MIME client) however, still requires that the headers are rewritten to make sure that the S/MIME attachment is not blocked. See *Why is a Djigzo gateway required?* for more information.

### Should a copy of the private key be available on the Djigzo gateway?

No not necessarily. If the gateway receives an email from an external sender and the email is already S/MIME encrypted, the gateway only needs to rewrite

the headers. There is no need to decrypt the message. A private key therefore is not required on the gateway.

If all email forwarded to the BlackBerry smartphone should be encrypted, even when the original message was not encrypted, the public certificate of the BlackBerry smartphone should be available on the gateway. The gateway will use the public certificate to encrypt all email forwarded to the BlackBerry smartphone.

### **Is Djigzo for BlackBerry compatible with the BlackBerry smartcard reader?**

Yes. Djigzo for BlackBerry uses the BlackBerry's built-in cryptographic functionality. Private keys stored on smartcards can therefore be used as long as the smartcard is compatible with the BlackBerry smartcard reader.

### **Are email attachments supported by Djigzo for BlackBerry?**

Yes. The complete message (body and attachments) is encrypted. Attachments for which a content handler is registered, can be directly opened (for example .doc and .xlt files will be opened with *Documents to Go*). Attachments can be saved on the device or on the SDCard.

### **is HTML email supported by Djigzo for BlackBerry?**

Yes. HTML email will be shown with all mark-up (like color, images etc.). If the email contains an alternative text part, the user can switch between the HTML and text version.

### **Does Djigzo for BlackBerry validate signatures?**

Yes. If the email is digitally signed, the signature will be validated using all the available intermediate and root certificates. If the signature is not correct (for example the issuer of the signing certificate is not trusted) a warning will be shown.

### **Why is email sent with Djigzo for BlackBerry relayed via a Djigzo gateway?**

Email sent with Djigzo for BlackBerry is protected with S/MIME and sent to the configured relay email address (S/MIME tunnel). The Djigzo gateway listens for email sent to the relay address and checks whether the message was signed with an approved certificate and whether the sender is allowed to relay email. The email is then forwarded to the final recipient. The main advantage of relaying email via the Djigzo gateway is that encryption management (certificates, PDF passwords, domain settings etc.) can all be done on the gateway.

### **Can I prevent the user from sending non-protected email?**

The *Clear mail policy* can be set to *Deny*. If the *Clear mail policy* is set to *Deny* and the user tries to compose a normal non-protected email, a warning will be

shown and the compose window will be forcefully closed. If *Clear mail policy* is set to *Warn* only a warning will be shown. The user is still allowed to compose a non-protected email.

### **I always need to enter the key store password when I open a message. Why is that?**

The *Private key security level* of the decryption key determines when and how often the key store password should be entered. If the *Private key security level* is set to *Medium* or *High*, a password must be entered when the private key is accessed. With the *Low* security level, a password is not required when the private key is accessed. You can change the *Private key security level* of a private key by opening the BlackBerry certificate options (options→Security Options→Certificates), select the certificate for which the security level should be changed and then from the content menu select “Change Security Level”.

## **General**

### **Suppose we issue a client a private key and the client loses it. What should we do?**

The two main questions which should be answered are: *is the key required to decrypt old email?* and *is the key compromised?*

In most cases the client should still be able to decrypt previously received email. The client therefore needs a copy of the private key. If the client did not backup the previously received key (i.e. a copy of the *.pfx* file) a new copy of the key can be securely sent to the recipient. Previously received encrypted email can be opened again after the key has been imported into the email client or operating system.

Whether or not the certificate should be revoked (i.e. placed on the CRL) depends on what is actually meant with ‘...the client loses it’. If for example the key was lost because the system had to be reinstalled because of a system crash then the key is technically not compromised. If however the key was stored on a laptop and the laptop was stolen, the key should be considered to be compromised. The certificate should be placed on the CRL and a new certificate and key should be generated for the client.

### **Suppose we issue a client a private key and our relationship with that client ends. What should we do?**

Technically speaking, nothing should be done. If the relationship was completely ended and no email should ever be sent again encrypted to the client, the client certificate can be revoked and the certificate can be deleted. By adding the certificate to the CRL the client can no longer use the certificate. Whether or not a certificate should be revoked when the relationship ends is dictated by the companies policy.

The main disadvantage of revoking every certificate after a short period is that the CRL grows with every newly revoked certificate. A certificate should remain on the CRL at least until the certificate has expired. If client relationships

are relatively short it's better to issue certificates with a short validity interval (for example expire within 1 year instead of 5) then to place the certificate on the CRL after use.

**Suppose we issue a client a private key and that key is stolen from the client's computer. What's the worse that can happen? What should we do on our end if we suspect a client's key has been stolen?**

S/MIME provides the following cryptographic security features: authentication, message integrity and non-repudiation. If a key is compromised (for example the laptop on which the key was stored was stolen) all of the three features no longer hold for messages encrypted and signed with that certificate. In other words, any email sent to the client can be read by anyone in possession of the key (authentication) and anyone in possession of the key can sign a document pretending to be the rightful owner (message integrity and non-repudiation).

If a key is compromised, or suspected to be compromised, the certificate should be revoked by placing the certificate on the CRL. This ensures that the gateway will no longer use the certificate for encryption and that all email clients will report the certificate to be revoked.

Instead of issuing 'soft keys' (i.e. a *.pfx* which should be installed on the clients computer) a more secure way of providing keys and certificates to clients would be to use a USB token. A key stored on a USB token is password protected and cannot be copied. The main disadvantage of a USB token is that it is not as cost effective as sending a *.pfx* file and that additional software must be installed.

**Suppose our Djigzo server becomes compromised. What's the worse case scenario? How should we react?**

If the Djigzo server is completely compromised all incoming and outgoing email sent through the gateway after the break-in could have been intercepted by the attacker. All keys should also be considered to be compromised and all issued certificates should therefore be revoked and everyone involved should be warned that the root certificate should no longer be used. Because all email could have been compromised all involved parties should be notified of the security breach (whether or not this is required depends on the companies policies and state and federal regulations).

Djigzo stores all private keys in a PostgreSQL database. This means that if the server is compromised all keys can be copied. To provide more security, a Hardware Security Module (HSM) can be used. All cryptographic functions involving private keys are handled by the HSM. Because all the private keys are stored on the HSM and all the private key handling is done inside the HSM all private keys are protected against copying. The main disadvantage of an HSM is that it's very expensive.

**If we get a root certificate signed by a CA, can we generate unlimited end-user certificates from that without any intermediates? Is this a sound practice?**

A root certificate is never signed by another CA. A root certificate is by definition self-signed. If your intermediate certificate is signed by a root certificate, it is best to issue another intermediate CA and securely store the original. The reason for this is that if you ever need to revoke the (second) intermediate certificate you can always issue another intermediate certificate. Sometimes however certificate issuers do not allow intermediate certificates from issuing other intermediate certificates (the path length constraint of a certificate can specify a maximum length of a certificate chain).

**When selecting a password to protect private keys sent to clients, should a unique one be used each and every time?**

It is more secure to always use a new password. If password protected private keys are sent to multiple clients all within the same company and the same password is used for all keys, in principle one recipient can open the password protected private key file of another recipient (access to the other recipients email box is still required though). If multiple keys are sent to just one recipient the same password can be used if required.

**What admin roles should customer service personnel be granted?**

The required roles largely depends on the tasks done by the customer service personnel.

**When issuing end-user certificates, a CA email is supposed to be specified. Who should see the mail associated with this email address?**

The password protected private key file is sent by email to the external client. The actual sender of this email is set to the CA sender. The reason this is required is that it allows the administrator to specify the security settings of the CA sender. Suppose that the preferences are set that all email sent by the gateway should be encrypted. If a key and certificate is generated and sent to an external client, the email containing the password protected private key file will also be encrypted. The recipient however cannot open the encrypted email because the private key required for opening the email is inside the encrypted email. The preferences of the CA sender should therefore be set to never encrypt. The sender of the email containing the password protected private key will be equal to the CA sender (i.e. the from header of the email will be set to the CA sender).

**Where can I get certificates?**

Djigzo has a built-in CA which can be used to issue certificates for internal and external users. A certificate and private key can be sent encrypted to an external

user. The external user can use the certificate with any S/MIME capable email client. Some external commercial and non-commercial certificate providers are:

StartSSL [www.startssl.com](http://www.startssl.com) (free)  
CACert [www.cacert.com](http://www.cacert.com) (free)  
Comodo [www.comodo.com](http://www.comodo.com) (free for personal use)  
Verisign [www.verisign.com](http://www.verisign.com)