# Global Information Assurance Certification Paper

# Encrypting Mail in a Windows Network

## GIAC Certified Windows Security Administrator (GCWN)

## Practical Assignment (v5.0, Option 1: Solving a Windows Problem)

**David Pérez Conde**

**August 2004**


## ABSTRACT

This paper constitutes the practical assignment (v5.0, option 1) that I submitted as one of the requirements to obtain the GCWN certification (GIAC Certified Windows Security Administrator).

It is divided in three parts: first, the issue of sending secure e-mail in a Windows network is discussed, then, some solutions are evaluated and compared, and finally, a implementation guide is provided for the selected solution.

# Table of Contents

# 1 Securing E-Mail in a Windows Environment

Electronic mail (e-mail) is a form of communication we could not live without anymore in both the personal and the corporate worlds, specially in the latter. However, the security of the e-mail system is often overlooked.

The term security refers here to three concepts: availability, integrity and confidentiality of the e-mail messages.

Availability means the quality of being ready for use. If the e-mail system is unavailable, the users (people and/or automatic processes) cannot receive and/or send e-mail messages, and vice versa.

Integrity refers to the authenticity of the messages, both of the contents and of the sender. Can a recipient of an e-mail message be sure that the mail was in fact sent by the purported sender and that the contents were not changed in transit?

Last, but not least, there is the problem of the confidentiality of the information contained in the messages. Can a sender be sure that no one else but the intended recipients will be able to access the information of a message?

While the loss of availability is usually recognized as a great risk and it is often addressed by making the system (including network paths, servers and servers' components) redundant, the other two problems are too often dismissed as "not that important". Only a proper risk analysis can determine the level of risk that is acceptable in each particular situation, but there are certainly cases where protecting the integrity and confidentiality of e-mail messages is not only desirable but a must.

Let us imagine a company that has its headquarters located in Palo Alto, California, where research and development is conducted, and a remote factory in Malaysia. What would happen if a competitor sent a mail message to the factory, pretending to come from R&D, asking them to modify certain parameters of the manufacturing process that would in fact ruin the product? Hopefully the factory would notice that these changes would adversely affect the product and would try to confirm the order by calling R&D by phone and thus discover that the message was forged, before implementing the changes. Or, what would happen if an e-mail message containing the details of the design of their new product was intercepted by a competitor in transit between R&D and the factory?

Changing scenario, let us imagine a healthcare organization that routinely exchanges medical records, including diagnostics and treatment instructions from doctors, between different hospitals and clinics via e-mail. What could happen if someone managed to modify one of those messages in transit and for example changed the blood group of a patient that is about to be operated?

Would this risk be acceptable? Or, what would happen if the medical record of a patient were disclosed because someone managed to intercept an e-mail message containing that information? Would that risk be acceptable?

Fortunately, cryptography offers a good solution[1] to both problems: the contents of e-mail messages can be encrypted if confidentiality is needed, they can be digitally signed if integrity needs to be assured, or both if both characteristics need to be protected.

When the contents of the message are encrypted, only those in possession of the appropriate key can decrypt it and therefore only them can access the information contained in the message. That solves the confidentiality problem.

If a message is digitally signed, a hash of the contents of the message is encrypted and sent to the recipient together with the message (which itself can be sent in clear text or encrypted). If a recipient decrypts the hash using a key that can only correspond to a particular sender and the hash matches a new hash calculated by the recipient over the received contents, then the receiver knows two things: the contents were not altered in transit and the sender was the owner of the particular key that he or she used to decrypt the hash. That solves the integrity problem.

Of course, the whole cryptographic setup is based on the assumption that at least some of the keys will be kept secret. In symmetric encryption, the key must be shared by the sender and the receiver and kept away from anybody else. A huge problem with this kind of setup is the distribution of the keys to those that need them while at the same time making sure that nobody else gets them. Public key cryptography addresses that problem because only public keys need to be distributed which, by definition, are public and therefore it does not matter if some attacker intercepts them. Then, there is the problem of being able to verify the authenticity of the public keys, which is addressed by the use of signed keys, either in the form of digital certificates signed by certification authorities, or directly by the end users, because they have been able to verify the authenticity by some out-of-band method, like direct personal contact with the owner of the key. The first approach is the basis of what is known as a Public Key Infrastructure (PKI), while the second approach is the one taken by the OpenPGP standard.

The optimal solution would be a product or set of products that allowed users to send and receive signed and/or encrypted e-mail messages, with the following characteristics as a plus:

- easy deployment (installation and configuration) of the products,
- easy key distribution,

1 It is outside the scope of this document to define and explain the cryptographic concepts behind a public key infrastructure (PKI) solution.. The reader is assumed to be familiar with the following concepts: symmetric and asymmetric encryption, public and private keys, digital certificates, digital signatures, certification authorities and public key infrastructure (PKI).

- easy key management, and
- easy usage by end users.

This document analyzes three different combinations of products that could be used to sign and/or encrypt e-mail messages using public key cryptography, thus managing[2] the risk of loss of integrity or confidentiality of e-mail messages.

The problem is not particular to Microsoft Windows products or any other particular platform, nor to specific configuration setups. Instead, the problem is common to any system being used to send or receive e-mail messages. However, the scope of this analysis is limited to the Windows platform for obvious reasons[3] and only some notes on interoperability with other platforms are included. Also, only a specific set of Windows operating system (OS) versions is considered in order to achieve the necessary brevity of the paper. In particular, a Windows network composed of servers running "Windows Server 2003 Standard Edition" and clients running "Windows XP SP1" is assumed. Nevertheless, an effort has been made to point out which of the results are applicable to other Windows OS releases and which are not.



2 The word "manage" in this context means reducing the risk to an acceptable level if the risk cannot be completely eliminated.

3 This document is intended to fulfill part of the requirements for the author to obtain the GCWN certification: GIAC Certified Windows Security Administrator.

# 2 Product Evaluation

This section evaluates and compares different solutions to encrypt and digitally sign e-mail messages.

## *2.1 Product Selection*

There are many products available that offer encryption and digital signature of electronic mail. Since testing and comparing all of them would be a daunting task and the analysis would certainly not fit in the maximum length requirements of this paper, only three combinations of products were selected for testing and comparison. Two of these combinations represented two very different approaches while the third was a mixture of them.

In all cases the mail server was assumed to be MS Exchange Server 2003 running on Windows Server 2003 Standard Edition operating system and the clients were considered to be running Windows XP SP1.

### 2.1.1 Set #1: Microsoft Products Only

The first set of products are all from Microsoft:

- Microsoft Outlook 2003
- Microsoft Certificate Services (Windows Server 2003)
- Microsoft Active Directory (Windows Server 2003)

Microsoft Outlook 2003 is Microsoft's main mail client. It is the program that the user interacts with to send or receive mail messages.

Microsoft Certificate Services is Microsoft's implementation of a Certification Authority. Its purpose is to produce digital certificates

Microsoft active directory is a hierarchical database that stores much information about the elements of a Windows network. Among many other things, it can serve as a repository to store and exchange certificates.

This solution is based on the concept of a PKI, where trust is managed in a hierarchical way. Trust in a particular public key is decided according to whether a chain of certificates to a trusted root certification authority can be established.

### 2.1.2 Set #2: Free Software Products Only

The second set of products are all free software:

- Mozilla Thunderbird
- Gnu Privacy Guard (GnuPG or GPG)
- Enigmail

Mozilla Thunderbird is a free mail client. It is available for many different platforms, including Windows and in particular it runs on Windows XP SP1.

Gnu Privacy Guard (GnuPG or GPG) is a free (GPL) implementation of the OpenPGP standard (RFC2440) [RFC01] for encryption and digital signature of e-mail messages and files in general.

Enigmail is a free (GPL) [GNU02] plug-in for the Mozilla mail clients family, including Firebird, that allows the user to easily access the functionality provided by GnuPG from the mail client to encrypt, decrypt, sign and verify messages.

The trust model of OpenPGP is very different from that of the previous solution. Trust in OpenPGP is not based in certification authorities at all. Instead, it is based in peer-to-peer relationships. For each pair of users to establish encrypted communications they will first have to exchange their public keys and the validity of those public keys can only be checked by some out-of-band communication, like exchanging them in person or verifying their fingerprint over the phone (assuming both ends can authenticate each other's voice). Key distribution in this schema becomes a huge problem. There are public key servers where OpenPGP public keys are published, but again, there is no guarantee of the keys obtained from them except with some additional out-of-band checking.

### 2.1.3 Set #3: Mixture of Microsoft, and Free Software

The third set of products is a mixture of the previous types of software (Microsoft's and free software):

- Microsoft Outlook 2003
- Gnu Privacy Guard (GnuPG or GPG)
- GDATA Outlook plug-in for GnuPG

This set of products combines Microsoft's mail client with the encryption capabilities of GnuPG. For that integration, a plug-in for Outlook is needed to interface between the two products. A plug-in [GDA01] from a company called GDATA [GDA02] was chosen for this task.

## *2.2 Test Environment*

All tests were performed with three virtual systems using VMware Workstation [VMW01]. One of

the systems was configured without a network card, so it was a completely isolated system, and the other two were connected to an isolated virtual LAN.

No communication was allowed to enter or leave the isolated network except for the transfer of installation files of some of the products from the host system to the guest All Microsoft products were installed from original installation CDs and all the other products were downloaded from the Internet and installed in the virtual systems using a removable usb storage device.

The isolated system was installed with Windows Server 2003 Standard Edition and was used for creating an off-line root CA using Microsoft Certificate Services.

The two systems connected to the isolated LAN were configured as server and client respectively.

The server would run Windows Server 2003 Standard Edition, and was configured as the only domain controller of a single Windows 2000 domain in a single forest. The client would run Windows XP SP1 and was configured as a domain member computer.

Active Directory (AD) was installed in the server (needed for acting as a domain server), which was also configured as an isolated DNS server because DNS is required for AD operation.

MS Exchange Server 2003 was also installed in the server. It was configured to speak with native clients such as MS Outlook but also to serve incoming mail using the IMAP4 protocol. It was also configured as an outgoing mail server via SMTP.

Three non-privileged domain users (user1, user2 and user3) were defined and each was assigned a Exchange mailbox. Only one of them, "user1" was given administrator privileges in the client[4], so that any differences between privileged and non-privileged users could be noted.

It is important to note that he test environment was configured with the only requisite that the chosen set of products could be tested against the problem in question. No effort was made, for example, to harden either of the boxes nor to tune the configuration of the Exchange server for security. Obviously, in a real network there would be many other issues that would need to be addressed apart from of the encryption and digital signature of e-mails. This was done purposely as in the author's opinion this is an optimal approach to successfully deploy security solutions or configuration changes: first, test each element separately, making sure it actually provides the expected benefits and doesn't introduce new risks; then, test all of the elements together to check how each element interacts with the rest; then, devise a plan for deployment; then, test the plan, and finally, execute it while taking great care to always have a way to roll back any changes in case something went wrong.

4 It was added to the local group Administrators of the client.

## *2.3 Test Results*

In this section the results of the tests are presented. Each solution was evaluated on the following aspects:

- **Installation and configuration**: Were the products easy to install and configure? Would the installation process scale?
- **Distribution of Keys:** Was it easy to distribute the public keys? Would the distribution process scale?
- **Usage:** Was it easy to use the whole solution?
- **Protection level:** Did the solution offer the expected level of protection?

### 2.3.1 Results for Set #1

The first set of products was composed of MS Outlook 2003, MS Certificate Services, and MS Active Directory.

An offline standalone root CA was installed in the isolated server and an online subordinate Enterprise CA was installed in the domain server. Certificates were published using both IIS (HTTP) and AD (LDAP).

#### *2.3.1.1 Installation and configuration*

Active Directory was installed automatically when the server was promoted to domain controller so installation was not an issue.

The installation of the root CA was pretty straightforward. However, editing the CDP and AIA extensions to reflect the LDAP path where CRLS would be published was a little tricky. It is the case that you need to know exactly which portion of the default URL you need to change for the CRL to be imported to the correct path. See the implementation guide in section 3 for more details about this topic.

Distributing the certificate of the root CA to all systems in the Enterprise forest was made by importing it into AD. With a single command, the certificate was automatically installed in the Trusted Root Certification Authorities Store of each and every system of the Enterprise. That is something really powerful when you have to manage thousands of systems. Another possibility would have been to distribute it using Group Policy, which does not apply to the whole Enterprise at once but still can be applied to every domain in the forest.

Installing the online enterprise subordinate CA was also pretty straightforward. It automatically

integrates with AD and IIS to publish the certificates and CRLs and to allow user enrollment via web.

The process of enrollment via web involves the users connecting to a web page using their web browser, clicking a few links to request a certificate and then clicking another link to install it in their local store. It is quite simple, certainly better than having to distribute thousands of certificates one by one, but it requires some user involvement and level of knowledge that some users simply don't have. Microsoft Windows Server 2003 Enterprise Edition supports, according to the documentation, auto-enrollment for users, that is, fully automated and transparent user enrollment. Unfortunately this feature is not available in the Standard Edition, which was the one at hand for the tests.

A problem was encountered that prevented non-privileged users to get their certificates. It was necessary that the first user to enroll from a particular workstation was a local administrator so that some activeX controls could be downloaded and installed. After the administrator had followed the process, the rest of users could enroll without problems. It is a known bug described in Microsoft's knowledge base article KB818026 [MSF01].

Installing MS Outlook 2003 was very straightforward. Following the instructions of the wizard was enough. After installation, the first time a user executes Outlook a configuration wizard appears that asks the user for the address or DNS name of the mail server, the type of server (Exchange) and the user name associated with the mailbox. It takes the user less than a minute to have it working.

Once the CAs had been set up properly and users had obtained their personal certificates, the only extra configuration required for users to send encrypted and/or signed mails is to click a couple of buttons before sending the message or setting up a couple of default configuration options (encrypt mails by default / sign mails by default).

#### *2.3.1.2 Distribution of Keys*

For user "A" to be able to send an encrypted message to user "B", user "A" needs to know the public key of user "B".

Using a Microsoft online Enterprise CA and AD this is not a problem because every time a certificate is generated for a user by the CA, a copy of the certificate, containing the public key of the user signed by the CA, is stored in AD. Then, any user wanting to send an encrypted message to another user can get the certificate of the recipient from AD. This is done automatically by MS Outlook, making the whole process transparent for the user.

The problem remains though for exchanging ciphered mail with people outside the Enterprise.

#### *2.3.1.3 Usage*

Usage is really simple and easy once everything has been set up.

#### *2.3.1.4 Protection*

This solution does address the issue identified in section 1. The level of protection obtained using this solution should be analyzed by cryptanalyst's, which the author of this paper is not. However, it may suffice to say that messages are protected by encryption and signing algorithms that are public and which are currently considered strong by the community.

### 2.3.2 Results for Set #2

The second set of products was composed of Mozilla Thunderbird 0.7, Gnu Privacy Guard 1.2.5 and Enigmail 0.85.

The three products were installed in the client workstation (Windows XP SP1).

#### *2.3.2.1 Installation and configuration*

First, Mozilla Thunderbird 0.7 was downloaded from the following URL,

http://www.mozilla.org/products/thunderbird/

and installed following the instructions of the typical installation wizard that walks you through the whole process.

Configuring it for accessing the mailboxes in the Exchange server was easy using the IMAP4 service of Exchange and configuring Thunderbird to use it.

Then ,Enigmail 0.85 was downloaded from:

http://enigmail.mozdev.org/download.html

and installed following the instructions included in the same download page. It comes in two .xpi files (enigmime and enigmail) that must be installed from inside Thunderbird using “Tools-> Extensions-> Install”. The Enigmime module needs to be installed only once by a system administrator. The Enigmail module must be installed by each user.

Finally, GnuPG 1.2.5 was downloaded from:

http://www.gnupg.org/(en)/download/index.html

and installed under C:\Program Files\GnuPG by simply moving its files to that directory. Then, a registry key was created to point to a directory in the profile of the user:

HKLM\SOFTWARE\GNUPG\HomeDir=%USERPROFILE%\Application Data\GnuPG

Unfortunately, users must create this directory in their own profiles by hand before they can use GnuPG

#### *2.3.2.2 Distribution of Keys*

Again, For user "A" to be able to send an encrypted message to user "B", user "A" needs to know the public key of user "B".

In OpenPGP there is no such thing as certification authorities. There are public servers were keys are stored and exchange but ultimately the decision of trusting a particular key corresponds to the end user.

Engimail allows the user to create a key pair and then send the public key to any mail recipient or to a public key server.

It also allows the user to import public keys sent to him or her in an e-mail.

The management of the keys becomes an important issue as soon as the number of recipients grows to a big number. It simply does not scale. This is one of the main problems of this kind of solution.

#### *2.3.2.3 Usage*

Usage is a bit more complex than in the first solution only because of the key management problem. In all other respects this solution is as easy to use as the other.

#### *2.3.2.4 Protection*

This solution also addresses the issue identified in section 1. Again, the level of protection obtained using this solution should be analyzed by cryptanalyst's, which the author of this paper is not. However, it may suffice to say that messages are protected by encryption and signing algorithms that are public and which are currently considered strong by the community.

### 2.3.3 Results for Set #3

Unfortunately, it was discovered during the tests that the Outlook plugin from GDATA is known to work with Microsoft Outlook 2000 but it is not to be known to work with Microsoft Outlook 2003 and in fact it does not work. In appearance everything is fine, plugin buttons appear in the Outlook GUI and no error message pops up, but messages supposedly sent encrypted are actually sent empty, that is with an empty body (headers are sent OK).

The plugin is not actively maintained anymore by GDATA nor by any other organization or individual so it does not seem probable that a version for Windows Server 2003 appears any time soon.

Therefore, this solution was abandoned as non-working and therefore not valid.

## *2.4 Conclusions*

The two main solutions analyzed (set #3 was discarded) properly address the problem at hand and offer a very good level of protection of the confidentiality and integrity of e-mail messages.

However, a solution based on OpenPGP does not scale as well as a PKI based solution. If the number of people with whom it is desired to exchange encrypted and/or signed mail gets big, then the key management becomes soon an important issue whereas in a PKI based solution, using certificates and certification authorities, it does not.

Therefore, for a large Windows network, set #0 of the tested solutions would be the most suitable.

The next section describes in detail how to implement this kind of solution.

PS: As a side note, the author was very surprised to learn that Microsoft is using PGP, and not Microsoft Certificate Services, to digitally sign their security bulletins sent by mail [MSF02]. One might expect that they used their own solution instead of someone else's to guarantee the authenticity of their security bulletins. Forcing Windows administrators to use non-Microsoft products when there is are Microsoft products that fulfill the same task does not seem to fit Microsoft. There must be historical or political reasons behind, because technically one solution is not better than the other for this particular matter.

# 3 Implementation Guide

This section describes how to add encryption and digital signatures to a Windows based e-mail system network using Microsoft Certificate Services.

The following assumptions are made about the network:

- There is a single Active Directory forest
- Servers are running Windows Server 2003 Standard Edition
- Its workstations are running Windows XP SP1
- Mail servers are running Microsoft Exchange Server 2003
- Client e-mail application used by users to send and receive mail is Microsoft Outlook 2003, installed in their workstations.
- DNS, AD, Exchange and Outlook are all properly configured and users can exchange non-encrypted and non-digitally signed e-mails without problems.

The procedure that will be described would probably apply to other versions and setups, but it has only been thoroughly tested by the author with the above configuration.

It is assumed that the reader is familiar with the concept of a Public Key Infrastructure and all its elements, like certificates, certification authorities (CA), certificate revocation lists (CRL), etc. This section will focus on how to implement a PKI using Microsoft Certificate Services with the objective of encrypting and signing e-mail messages.

## *3.1 Step 0: Designing a CA Hierarchy*

Designing a CA hierarchy involves deciding whether the root CA will be built in-house or a well-known external CA will be used instead, how many levels of subordinate CAs will be set up and whether those CAs will be off-line or on-line.

Having an in-house root CA poses the problem of distributing the root CA's certificate to all those who need to trust it in order to decrypt or verify the signature of messages encrypted or signed using certificates that ultimately depend from this root CA. For example, if the objective is to exchange encrypted e-mail only inside a company then an in-house root CA may be best (and cheapest) choice. If the company wants to exchange encrypted or signed mail with a few other organizations, it might still be possible to use an in-house root CA and distribute its certificate among those organizations. However, if the objective is to exchange encrypted or signed mail with the rest of the world, then an external well-known root CA may be a better option.

The number of levels of subordinate CAs that is appropriate depends on the size and structure

(e.g. physical distribution or management boundaries) of the organization. For a small or middle size organization, a single level of subordinate CAs may be appropriate.

Finally, the choice between off-line and on-line CAs must balance security and convenience. Off-line CAs need someone to put the certificate requests in, getting the certificates out and then distributing them among the users, so they are an administrative burden, but if they are kept in a physically secure place it is very difficult that they get compromised. On-line CAs can be configured to accept requests and distribute the certificates through the web, in an almost unattended process, which is very convenient, but because they are connected to the network they are susceptible to network attacks and the probability of a compromise is much higher.

In order to keep this example simple yet illustrative, the following CA hierarchy will be assumed during the rest of this implementation guide.

A single CA will be set up off-line in-house. This CA could be a root CA or subordinate to a external well-known root CA. Since setting up a root CA and then distributing its certificate requires more effort, this option will be described in detail. Nevertheless, differences in the procedure for the other option will be noted where appropriate.

Then, a single CA, subordinate to the off-line CA, will be set up on-line and configured to allow user web enrollment.

Microsoft distinguishes four types of certification authorities (CA) in its CA software, Certificate Services

- Enterprise root CA
- Enterprise subordinate CA
- Stand-alone root CA
- Stand-alone subordinate CA

Enterprise CAs require access to Active Directory (AD), which means they need to be connected to the network (on-line), while stand-alone CAs can be installed off-line.

Therefore, the hierarchy described above will be implemented by installing a “stand-alone root CA” off-line and an “Enterprise subordinate CA” on-line. The differences of installing the first or a “stand-alone subordinate CA” will be noted.

## *3.2 Step 1: Installing an off-line standalone root CA*

Install Windows Server 2003 in a standalone server not connected to any network (it will be referred to as “standalone” from now on). Do not join this computer to any domain and never

connect it to any network. This is important to protect the root CA that will be installed in it. This root CA will be used to generate a subordinate certificate to an on-line enterprise CA.

Logon as Administrator.

Make sure the name of the computer and workgroup it belongs are correct, because after installing Certificate Services they cannot be changed due to the binding of the machine name to CA information. Changing the machine name or domain membership would invalidate the certificates issued from the CA. You can check or change the name and workgroup of the computer at Start-> Right click on My Computer -> Properties -> Computer Name -> Change.

Launch the "Add or Remove Programs" applet in the Control Panel (Start-> Control Panel -> Add or Remove Programs). Click "Add/Remove Windows Components". Select "Certificate Services". If you click "Details" you will see that both "Certificate Services CA" and "Certificate Services Web Enrollment Support" are selected. For some obscure reason you can't install the former without the latter even though you will not use web enrollment in this standalone server. Click next. Select "Stand-alone root CA"[5] and "Use custom settings to generate the key pair and CA certificate if you want to specify things like the key length or the hashing algorithm. Click next. If you have selected "Use custom..." then make your choices. For example, you may select "Microsoft Strong Cryptographic Provider" as CSP (Cryptographic Service Provider), "SHA-1" as the hash algorithm and 2048 as the key length. Select "Allow this CSP to interact with the desktop" so that the CSP can interact with the desktop of the user currently logged on. Click next.

Enter a common name for the CA (e.g: "MyRootStandaloneCA"). Select a validity period (e.g: 5 years). Click next. Enter locations for the certificate database, database log, and configuration information (you may leave the defaults, "C:\WINDOWS\system32\CertLog", "C:\WINDOWS\system32\CertLog" and "C:\CAConfig"). Click next. Insert the Microsoft Windows Server 2003 Standard Edition CD when prompted to do so. If you haven't installed IIS previously, you will get a popup message informing you that web enrollment support will be unavailable until IIS is installed[6]. Acknowledge the message clicking OK. Finally, click Finish and exit from "Add or Remove Programs" by closing the window.

Now that the root CA has been installed, some additional steps are needed to be make it ready for issuing certificates[7].

First, ensure that the default action upon receipt of a certificate request is to set it to pending. To

5 If you want to install a standalone subordinate CA this is the moment to make that choice. The process of installing and configuring such CA would be almost identical to that of installing the enterprise subordinate CA described later.

6 You do not need web enrollment on an offline standalone root CA.

7 These steps are clearly explained in the help documents of the certificate services snap-in, under "Certificate Services - > Checklist: Creating a certification hierarchy with an offline root certification authority".

do this, logon to the standalone server (which contains the offline root CA) and launch “mmc” (Start-> Run-> mmc). Add the “Certification Authority” snap-in: File-> Add/Remove Snap-in-> Add-> Select Certification Authority.-> click Add-> Select “Local computer” and click finish, close, and OK.

In the certification authority snap in, navigate to the name of the CA you created (e.g: “MyRootStandaloneCA). Right click on its name and select Properties. Then click “Policy Module” and “Properties...”. Select “Set the certificate request status to pending. The administrator must explicitly issue the certificate”. Click OK twice to close this window.

Then, you must edit the CDP and AIA extensions so that the URLs that will be included in all CRLs and certificates match the URLs where the CA's certificate and CRLs will be published periodically. Both the CA's certificate and the CRLs created by the CA must be made permanently available to any user that needs to verify the validity of any certificate under the CA's hierarchy. This can be done via a web server (http://), Active Directory (ldap://) or a network share (file://). If you know the URLs where they will be published, enter them now. To do so, navigate to the same window where the “Policy Module” tab was and select the tab named “Extensions”. Note that because the system is isolated some of the variables in the URLs will not be of much use. For example, if you know you will publish the CRL and the CA certificate in a web server and the URL to access the directory where the files will be located is “http://server.example.com/rootcacerts/”, then add a new entry in both CDP and AIA extensions, similar to the default but changing the server and directory part. For example:

Default:

http://**<ServerDNSName>/CertEnroll**/<CaName><CRLNameSuffix><DeltaCRLAllowed>.crl

New entry:

http://**server.example.com/rootcacerts**/<CaName><CRLNameSuffix><DeltaCRLAllowed>.crl

You can't edit the entries; you have to enter your own. It is not recommended to delete the default ones, though Better, keep them for reference and unmark the checkboxes for them and mark the checkboxes for your entries.

If you are going to publish the CRL in AD, you need to put the appropriate URL in place and manually publish a new version of the CRL before you publish it in AD using “certutil”, which is explained in the next section. In particular, you need to create an “ldap:///CN=<” entry similar to the default but replacing the variable <ConfigurationContainer> by the configuration container of your root domain in the forest. For example:

Default:

ldap:///CN=<CATruncatedName><CRLNameSuffix>,CN=<ServerShortName>,CN=CDP,CN=Public Key Services,CN=Services,**<ConfigurationContainer>**<CDPObjectClass>

New entry:

ldap:///CN=<CATruncatedName><CRLNameSuffix>,CN=<ServerShortName>,CN=CDP,CN=Public Key Services,CN=Services,**CN=Configuration,DC=example,DC=com**<CDPObjectClass>

If you don't know yet these URLs, you should find out before issuing any certificate. A wrong URL will make it impossible for clients to validate certificates and any certificates with the wrong URLs will have to be re-issued.

If you are going to publish them using AD and you don't know yet the exact LDAP URLs, you can import the root CA certificate into AD first (see next section) and take note of the LDAP URLs where it gets stored, which are displayed if it is imported successfully (see next section). Then, go back to the off-line CA,and edit the LDAP URLs to resemble that one. The URL in the AIA extension must match exactly the one that was displayed when importing the certificate. The URL in the CDP extension should be easy to guess from it.

Then, manually force the CA to create (publish) a new CRL so that it reflects your changes in the CDP extension. To do this, in the same snap-in as before, Certification Authority, navigate[8] to "Revoked Certificates", under the CA name leaf, right click on it and select "All Tasks-> Publish". Select "New CRL" and click OK. The graphical interface does not tell you anything about whether or where the CRL was saved, but if everything went fine you will find a file called "<CA_NAME>.crl" (e.g. "MyRootStandaloneCA.crl" under the directory "Systemroot\system32\CertSrv\CertEnroll\".

In the same directory you should see a copy of the certificate of the CA (e.g. "<SERVER_NAME>_MyRootStandaloneCA.crt", where <SERVER_NAME> is the hostname of the server where the CA is installed). You will need to copy both files to a removable storage device so that you can publish them (see next section).

Finally, perform a sound backup[9] of the CA, including a specific backup of the CA elements and a full system backup. To back up the CA elements, in the same snap-in as before navigate to the CA name leaf, right click on it and select "All Tasks->Back up CA". Then, follow the instructions of the wizard. Store the backups in a secure location.

8 Alternatively, the CRL can be published using the command "certutil -crl".

9 You may want to perform more than one, just in case.

### 3.3 Step 2: Publishing the certificate and CRL of the root CA

The certificate and the CRL of the root CA must be made accessible to all users. That includes anyone receiving mails signed by certificates issued by this PKI who needs to verify the validity of the signature.

As it was introduced in the previous section, they can be published in a number of ways: via web, using a network share, or importing them into Active Directory. These ways are not exclusive: the same certificate and CRL may be published in multiple web pages, in multiple network shares and in several Active Directory (AD) forests, all at the same time. All the URLs pointing to those places must be declared in the root CA in the CDP and AIA extensions (Tab "Extensions" in the Properties window of the CA, see previous section).

The two files you took into the removable storage device (e.g. floppy) must be copied to all those places and they must be made available using the appropriate server. For example, to publish them in a web URL (e.g. http://server.example.com/rootcacerts/FILE_NAME), the files must be copied to a directory in the web server's filesystem and then the web server must be configured to serve that directory as expressed in the URL, for example creating a virtual directory and making it readable by everyone.

To publish them in a network share URL (file://\\server.example.com\rootcacerts\FILE_NAME), the files will need to be copied to a directory in the server's filesystem and that directory be shared with read permissions for everyone.

To publish them using AD, the command "certutil" must be executed with domain admin privileges in a server of the domain as follows:

First, execute:

certutil -f -dspublish <ROOT_CA_CERT_FILE>

where <ROOT_CA_CERT_FILE> is the name of the file containing the root CA certificate. For example:

certutil -f dspublish rootca_MyRootStandaloneCA.crt

where "rootca" is the hostname of the system where the root CA is installed.

This command accomplishes two purposes: first, it imports the certificate into the Trusted Root Certification Authorities store of AD, making it automatically a trusted root certification authority for all systems in the entire forest[10]; second, it imports it to a particular location in AD where it will be available for any system requiring a copy.

10 It will take some time before it propagate to all systems, but eventually it will.

If you don't publish the root CA certificate to AD, you will need to find another way to push the certificate into the Trusted Root Certification Authorities of the systems that will use certificates. Group policy is very suitable for that task. Using group policy you can push the certificate to all systems on particular domains or organizational units.

External users will need to import the certificate into their respective Trusted Root Certification Authorities store in some other way. This problem of pushing the certificate to the Trusted Root Certification Authorities of users would not exist if the off-line CA would have been a subordinate to an external well-known root CA. In that case, the certificate of the root CA is already in the store of most systems because it is shipped with the operating system. Yet the problem of publishing the certificate and the CRL for anybody to access would still be the same.

The second execution of “certutil” should be:

“certutil -f -dspublish <ROOT_CA_CRL_FILE>”

where <ROOT_CA_CRL_FILE> is the name of the file containing the root CA CRL. For example:

certutil -f dspublish MyRootStandaloneCA.crl

This imports the CRL to a particular location in AD where it will be available for any system requiring a copy.

Note that for this command to complete successfully the CDP extension must have been edited properly prior to producing the CRL file. See previous section for information on how to edit CDP and AIA extensions.

Both commands return the LDAP URLs where the objects are stored. Table 1 shows an example of a successful execution of these commands.

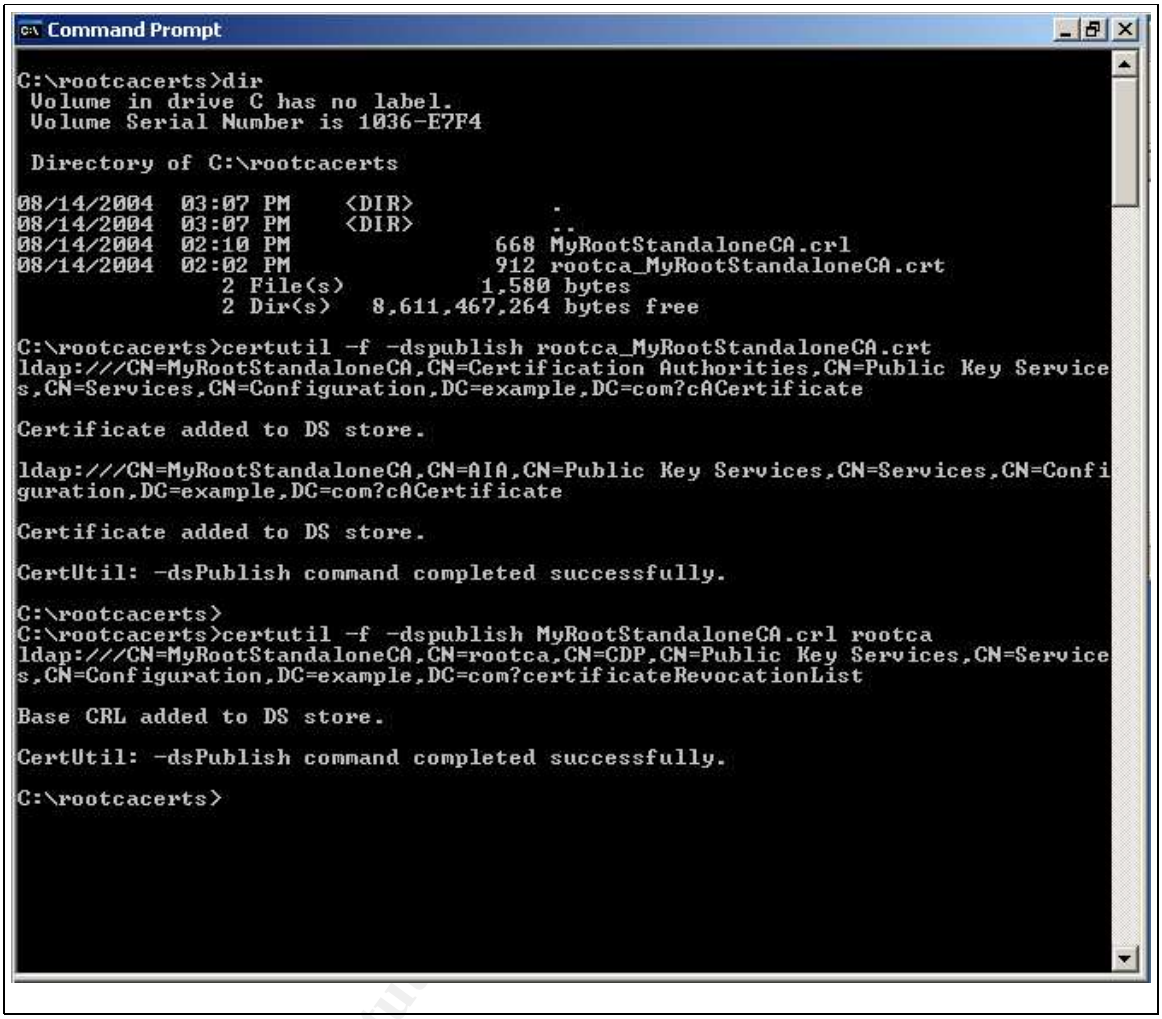


Table 1 Successful import into AD of a root CA certificate and CRL using "certutil"

## *3.4 Step 3: Installing an on-line Enterprise subordinate CA*

Log as administrator onto the server where the on-line Enterprise subordinate CA is to be installed. The server must be a member of the domain so that user certificates are automatically published to AD when they get created.

If you are going to use web enrollment, make sure IIS is already installed in the system.

Launch the "Add or Remove Programs" applet in the Control Panel (Start-> Control Panel -> Add or Remove Programs). Click "Add/Remove Windows Components". Select "Certificate Services". Click "Details" and verify that both "Certificate Services CA" and "Certificate Services

Web Enrollment Support" are selected. Click OK. Click next. Select "Enterprise subordinate CA" and "Use custom settings to generate the key pair and CA certificate if you want to specify things like the key length or the hashing algorithm. Click next. If you have selected "Use custom..." then make your choices. For example, you may select "Microsoft Strong Cryptographic Provider" as CSP (Cryptographic Service Provider), "SHA-1" as the hash algorithm and 2048 as the key length. Select "Allow this CSP to interact with the desktop" so that the CSP can interact with the desktop of the user currently logged on. Click next.

Enter a common name for the CA (e.g: "MyEnterpriseCA"). Leave the suggested suffix (e.g. DC=example,DC=com). You cannot choose the validity period because it will be determined by the parent CA. Click next. Enter locations for the certificate database, database log, and configuration information (you may leave the defaults, "C:\WINDOWS\system32\CertLog", "C:\WINDOWS\system32\CertLog", "Store configuration..." unchecked). Click next. Select "Save the request to a file" and specify a file name. You can point it directly to a removable device (e.g. a floppy) that you will use to carry the request to the root CA (e.g: "A:\certrequest.req"). Click next. It will need to temporarily stop IIS; allow it by clicking yes on the popup message. Insert the Microsoft Windows Server 2003 Standard Edition CD when prompted to do so. After a while you will get the popup message shown on Table 2 explaining the next steps required to complete the installation of the enterprise CA.


Table 2 Next steps to finish the installation of the enterprise intermediate

Acknowledge the message by clicking OK and finally, click Finish.

Take the removable device where the certificate request was stored and insert it into the server that contains the off-line root CA.

Logon to that server and launch "mmc" (Start-> Run-> mmc). Add the "Certification Authority" snap-in: File-> Add/Remove Snap-in-> Add-> Select Certification Authority.-> click Add-> Select "Local computer" and click finish, close, and OK.

In the snap in, navigate to the name of the CA you created (e.g: "MyRootStandaloneCA). Right

click on its name and select “Submit new request...”. Locate the certification request file in the removable device and click open. This will load the request into the “Pending Requests” folder in the snap-in. Go to that folder, right click on the request and select All Tasks-> Issue. In less than a second the request will disappear from “Pending Requests” and move to the “Issued Certificates” folder. Go to that folder, right click on the request and select All Tasks-> Export Binary Data-> Save binary data to a file (Binary Certificate)-> Enter a name for the file to be saved in the removable device and save it.

Take the removable device, containing now the request from the enterprise CA and the recently issued by the root CA for the enterprise CA and insert it into the server where the enterprise CA was installed.

Logon to that server as administrator. Launch the Certification Authority snap-in in “mmc” following the same procedure as in the standalone server. Right click the CA's name (e.g. “MyEnterpriseCA”) and select All Tasks-> Install CA Certificate. Locate the file containing the certificate issued by the root CA and click Open. The certificate will get installed.

Finally, to start the CA, right click the CA's name again and select “All Tasks-> Start Service”.

## *3.5 Step 5: Enrolling users*

The process of requesting certificates and getting them is called enrollment.

There is computer enrollment, where computers get certificates for their own use, like communicating using IPSec, and user enrollment, where users get certificates for their own use, like encrypting mail.

This enrollment can be done manually, via web, or automatically (auto-enrollment). In Windows 2000 only computer auto-enrollment existed. In Windows Server 2003 Enterprise Edition it is possible to have user auto-enrollment as well, but not in Windows Server 2003 Standard Edition, which is the version considered in this paper. In this case, web user enrollment is the best option.

Users must obtain their own personal certificate to sign and receive encrypted mail. To obtain the certificate using web enrollment they must follow the following procedure:

They[11] must launch Internet Explorer in their workstation and navigate to the following URL:

http://<ENTERPRISE_CA_NAME>/certsrv/

11 Because of a mismatch in the versions of a DLL included in Windows XP and Windows Server 2003, the first user to enroll from a particular workstation must be a user with administrator privileges over that workstation, so that some activeX programs can be downloaded and installed. After an administrator has followed the process once, all other users can enroll without needing special privileges. This is recognized as a bug by Microsoft in their KB article 818026 [MSF01]

where <ENTERPRISE_CA_NAME> is the DNS name of the server running the Enterprise CA (e.g. “server.example.com”).

After authenticating themselves to the web server they will get a welcome page with a link called “Request a certificate”. When they click that link they will get to a page with a link called “User Certificate”. Clicking that link will get them to a page with a button called “Submit”. When they click that button they will get a message warning them that the website is going to request a certificate in their behalf and that they should continue if they trust the web server. If they click “Yes” the certificate will be generated and in a few seconds they will be presented a page with a link called “Install this certificate”. They must click that link to have certificate added to their Personal Certificates Store in their workstation and accept the warning message that will popup. Finally, they will get a web page informing them that their new certificate was installed successfully.

They can see their certificate by opening Internet Explorer and clicking Tools-> Internet options -> Content -> Certificates-> Personal. Double clicking the certificate they can see the information stored in it.

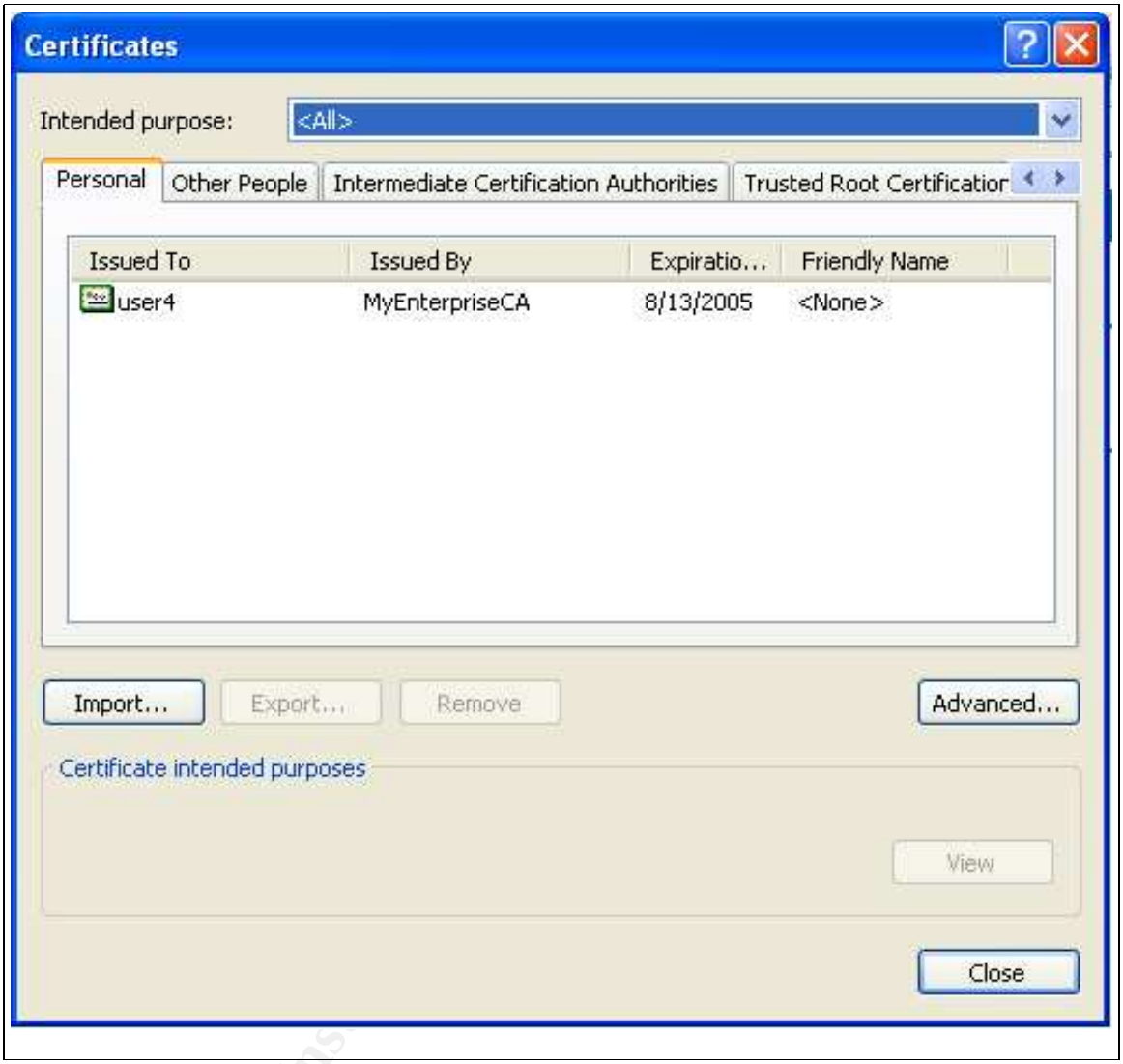


Table 3 User certificate

Next, they should back up their certificate by exporting it to a file in a removable storage device (e.g. floppy) and store that device in a secure place like a locked drawer. To do so, they can open the certificates window as described in the previous paragraph, select their certificate, click "Export..." and follow the instructions of the wizard.

Once their user certificate is in their store, they are ready to send and receive encrypted e-mail and to sign e-mail messages. That process is described in the following section.

## *3.6 Step 6: Sending encrypted and/or signed e-mail*

This is the easiest part after everything else has been set up.

To send a signed and/or encrypted message to any other user[12] in the Enterprise, all that users have to do the following right before sending the message: click the "Options" button (in the window where the message is being composed), click "Security Settings" in the dialog box that will appear and then select the desired option or options:

- "Encrypt message contents and attachments"
- "Add digital signature to this message"

By default both options are cleared and mail is sent in cleartext and unsigned if the user does not explicitly specify otherwise. However, the user can change these default settings by selecting "Tools->Options->Security" from the main Outlook window and marking there the desired default behavior.

To decrypt and/or verify the signature of a received message, users only have to open it as they would do with any cleartext or non-signed message. Outlook will decrypt it and will verify the validity of the digital signature. The only difference with a cleartext/non-signed message will be two little icons on the right of the message window and a new header "Signed By: ". This can be seen in Table 4.



12 The recipients must have obtained their own certificates so that they can receive encrypted mail. It is not necessary, though, for them to receive and be ables to verify digitally signed messages.

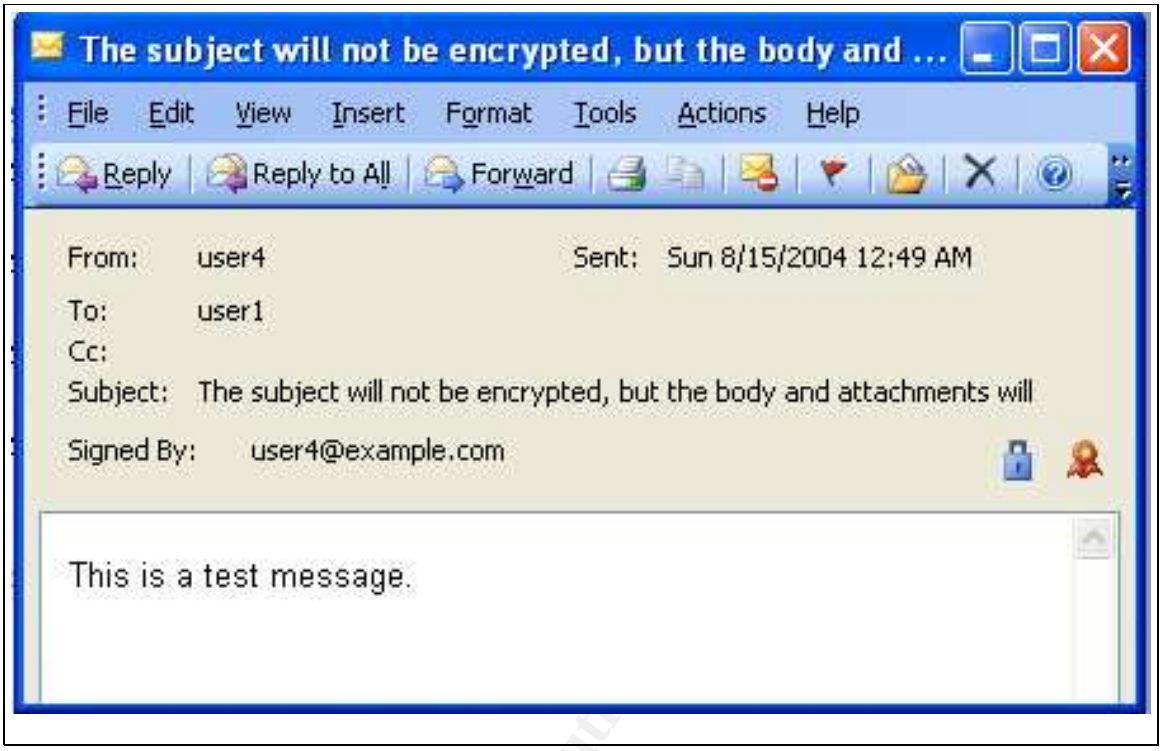


Table 4 Encrypted and signed message received.

This finishes the setup of the proposed solution to the problem of protecting the integrity and confidentiality of important e-mail messages. To conclude, one important warning for users: as the sample message in Table 4 says, the subject of messages do not get encrypted, only the body and attachments do!

# 4 References


[GDA01] G DATA Software AG. *Outlook plugin download page*. http://www3.gdata.de/gpg/download.html

[GDA02] G DATA Software AG. *Home page of G-DATA*. http://www.gdata.de

[GNU01] GNU Project. *Homepage of GnuPG.* http://www.gnupg.org/(en)/

[GNU02] GNU Project. *GNU General Public License.* http://www.gnu.org/copyleft/gpl.html

[MOZ01] Mozilla. *Home page of Mozilla Thunderbird.* http://www.mozilla.org/products/thunderbird

[MOZ02] Mozilla. *Home page of Enigmail.* http://enigmail.mozdev.org

[MOZ03] Mozilla. Installation instructions for Enigmail. http://enigmail.mozdev.org/help.html

[MSF01] Microsoft. *KB818026 User Is Prompted to Download ActiveX Control When Requesting Certificate*. http://support.microsoft.com/?id=818026

[MSF02] Microsoft Corporation. *Microsoft Security Notification Service.* http://www.microsoft.com/technet/security/bulletin/notify.mspx

[MSF03] Microsoft Corporation. Checklist: *Creating a certification hierarchy with an offline certification authority*. http://www.microsoft.com/resources/documentation/WindowsServ/2003/standard/proddocs/en-us/Default.asp?url=/resources/documentation/windowsserv/2003/standard/proddocs/en-us/sag_cs_checklist_offline.asp

[RFC01] Network Working Group. *RFC 2440 – OpenPGP Message Format.* http://www.faqs.org/rfcs/rfc2440.html

[RFC02] Network Working Group. *RFC 3156 – MIME Security with OpenPGP.* http://www.faqs.org/rfcs/rfc3156.html

[VMW01] VMware, an EMC Company. *Home page of VMware*. http://www.vmware.com

Last Updated: April 22nd, 2018

# Upcoming Training


| | | | |
|---|---|---|---|
| **SANS Security West 2018** | **San Diego, CA** | **May 11, 2018 - May 18, 2018** | **Live Event** |
| **SANS Crystal City 2018** | **Arlington, VA** | **Jun 18, 2018 - Jun 23, 2018** | **Live Event** |
| **SANSFIRE 2018** | **Washington, DC** | **Jul 14, 2018 - Jul 21, 2018** | **Live Event** |
| **SANSFIRE 2018 - SEC505: Securing Windows and PowerShell Automation** | **Washington, DC** | **Jul 16, 2018 - Jul 21, 2018** | **vLive** |
| **SANS Boston Summer 2018** | **Boston, MA** | **Aug 06, 2018 - Aug 11, 2018** | **Live Event** |
| **SANS Amsterdam September 2018** | **Amsterdam, Netherlands** | **Sep 03, 2018 - Sep 08, 2018** | **Live Event** |
| **SANS Network Security 2018** | **Las Vegas, NV** | **Sep 23, 2018 - Sep 30, 2018** | **Live Event** |
| **SANS San Diego Fall 2018** | **San Diego, CA** | **Nov 12, 2018 - Nov 17, 2018** | **Live Event** |
| **SANS OnDemand** | **Online** | **Anytime** | **Self Paced** |
| **SANS SelfStudy** | **Books & MP3s Only** | **Anytime** | **Self Paced** |