PKI Interoperability Document

Zeb Bowden
Systems Architect
VTmig – http://vtmig.w2k.vt.edu
Version 1.2 - 01/12/2005

I. Project Introduction

From a security standpoint having an internal Public Key Infrastructure (PKI) established and in use is an invaluable resource for an organization. A good PKI can bring tremendous benefits to the organization, not only in terms of security but also in terms of costs and efficiency. An excellent way to leverage these benefits is to incorporate this PKI into an Active Directory (AD). This will provide an excellent mechanism for getting certificates out to the users of the certificates (users, servers, etc). Additionally, this also has the potential to ease management of the PKI.

The goal of this project was to determine at what level certificates issued by the Virginia Tech PKI could be utilized and integrated into the existing and future Active Directory structure at Virginia Tech. Another goal of this project was to compare the value/benefits that would be derived from bringing up a Microsoft subordinate Certification Authority versus just customizing and adapting to the current OpenCA PKI.

II. Project Scope

The intended audience for this document is anyone interested in PKI interoperability between Microsoft and 3rd party Certification Authorities. However, many references are made to Virginia Tech specific locations and resources. Please keep this in mind if you are using this document as a guide in an organization other than Virginia Tech. Many topics discussed assume that the reader has at least novice understanding of PKI.

III. PKI Interoperability (Active Directory and OpenCA)

Zeb Bowden
09/09/04
v.1.2

Phase 1. Push VT DEV Root CA to the Active Directory

Tasks:

- Import cert info AD
- Set up multi boot workstation (w2k,xp,2k3)
- Verify cert distribution to workstations
- Check validation of cert path

Notes:
From the DEV VT CA website (http://radev.eprov.iad.vt.edu) you can download the Root CA certificate in several formats. To work properly with Active Directory we need to get the certificate in either the CER or CRT formats, either of these work fine.

1. Copy the certificate file (.CER or .CRT) up to one of the root Domain Controllers.
2. Open Active Directory Sites and Services and create a Group Policy Object (VT DEV Root CA GPO) on the Default-First-Site-Name Site.
3. Edit the GPO you just created; Computer Configuration->Windows Settings->Security Settings->Public Key Policies. Right click Trusted Root Certification Authorities, select All Tasks and then Import.
4. Follow the directions given by the Certificate Import Wizard, when given the opportunity browse to the location of the certificate file.

After you're done with the wizard you should see a certificate issued to DEV Virginia Tech Root CA in the Trusted Root Certification Authorities. You will now need to either allow time for replication to occur or force replication using the Active Directory Sites and Services MMC. After this has replicated you can force the Group Policy on a machine in the site by using secedit for Windows 2000 or gpupdate for XP. Then you should be able to open up the certificates MMC and verify that you have a certificate issued to DEV Virginia Tech Root CA under your Trusted Certification Authorities. To test the certificate path validation go to https://radev.eprov.iad.vt.edu, you should not be prompted about the certificate not being signed by a trusted source.

Conclusion:
This was tested under Windows 2000, Windows XP, and Windows 2003 Server (Enterprise) and worked as expected. We can be reasonably certain we can push the Virginia Tech Root CA using the Active Directory.

Phase 2. Use a web/SSL certificate issued by a subordinate DEV CA

Tasks:

- Set up IIS 6.0 web server on Windows 2003 Web Edition
- Get a web server certificate from DEV VT Root CA
- Assign the certificate to the web server from IIS
- Test SSL via IE 6.x
- Determine enabled features

Notes:

One nice feature of IIS is that it can generate a Certificate Request (CSR) for you. This hides some of the tedious management tasks involved with setting up an SSL enabled web server. However, this automatic CSR generation does not appear to be very customizable and therefore will not work with the DEV CA. To get a correctly formatted CSR we will need to use another tool that allows for much configuration of the CSR format such as OpenSSL. OpenSSL is a command line tool that is very powerful and also fairly complex. The tool is configurable via a configuration file that could potentially be packaged by the cert team and offered for download making generating a correctly configured CSR much easier. So assuming we have a correctly configured configuration file the command line to generate the CSR is: "*openssl req -config sslconf.txt -newkey rsa:2048 -keyout key.pem -out request.csr*". It is very important to save this private key (key.pem), we will need it later. Do not transmit this private key across the wire, just leave it as is until we get our signed certificate.

Now that we have our CSR (request.csr) we need to upload it to the DEV website for it to be signed. Go to https://radev.eprov.iad.vt.edu -> Request a certificate -> Server Request -> Fill in the appropriate information (make sure you select Web Server for the role). Probably a good idea to keep track of the PIN you enter, it's used to revoke a certificate. After the certificate is signed you can then download it from the website in DER format which is binary. Unfortunately, we need to get it into PEM format so we can read it in order to attach our private key to our certificate. OpenSSL provides an easy way of performing this conversion, the syntax is: "*openssl x509 -in <der_certificate> -inform DER -outform PEM -out certificate.pem*". Replace <der_certificate> with the path and filename of the certificate you downloaded.

Now we need to add the private key (key.pem) to the end of our certificate.pem file, use notepad to do this (simple cut/paste). Now we want to get our certificate in PKCS12 format, which is what IIS wants. Again we use OpenSSL: "*openssl pkcs12 -export -out exported_key_cert.pfx -in certificate.pem -name "zebweb"*". At this step you will be asked for a password that is used to encrypt the certificate because we don't want the private key to be readable, you will need this password in the next step.

Next we will import our PKCS12 formatted certificate into our computers store using the certificates MMC. Import it into the local computers personal certificate store. After the certificate is stored in the certificate store of the machine the key.pem and exported_key_cert.pfx files should be deleted. Having plain text files around on your system is just another security risk and it also makes it very easy to put the contents of those files on the wire.

Now we want to enable SSL on our web server. Open IIS Manager, navigate to your website, select properties and then select the Directory Security tab. Click the server certificate and then assign existing certificate at which point you'll select the one you just imported in the computer's personal certificate store.

At this point your web server is SSL enabled; you can get to it using https://servername. However, users will still receive the popup box alerting them that the CA that signed the certificate could not be verified. First make sure that the computer you're accessing the web server from trusts the DEV VT Root CA, this trust must be there. When you try it again you'll notice that it still is not trusted, that's because the entire certificate trust chain is not being pushed by the web server. The PKI structure at VT is such that web server certificates are signed by a subordinate CA (Class 1 Server CA) so on the web server itself we need to make sure that the Class 1 Server CA is a trusted CA. Once that certificate is added then the entire chain is passed and we no longer get the popup. To test the chain being passed we can use OpenSSL: "*openssl s_client -connect <server>:443 -prexit –showcerts*". You should see 2 certificates in the chain: the web server's certificate and the Class 1 Server CA certificate.

Now we will look into certificate revocation. The biggest problem with certificate revocability in relation to web servers is the fact that it isn't turned on by default in most browsers. Another big problem is the mechanism by which you send the list out, right now it's from a website and that's it. Certificate revocation does work however, but it requires you to:

1. Go to the VT PKI DEV website and request that it be revoked
2. Download the newest Certificate Revocation List (CRL) and install it on your machine
3. Turn on CRL checking in your browser

Conclusion:

This was tested using Windows 2003 Server (Web Edition) and IIS 6.0 and while it did not work as hoped, I think it is acceptable. It would be nice to have everything work using just wizards but the OpenSSL tool isn't that bad. The OpenSSL tool is pretty easy to use after you get the hang of it. Maybe the certificate team could put some how-to's up on their website (which I think they are in the process of doing now). With that in mind, it's worth mentioning that there are other tools that can perform the tasks OpenSSL does. OpenSSL appears to have the strongest user base and it can be used on just about every platform so in the long run it will probably make our lives easier.

Certificate revocability isn't the greatest, but it does work properly if you turn it on and maintain the revocation list. I think more and more people are just operating under the assumption that we can't rely on revocation of certificates

Phase 3. Use a user certificate issued by a subordinate DEV CA

Tasks:

- Determine mechanism for importing cert into the AD
- Get a user certificate from subordinate of DEV VT Root CA
- Test digital signing
- Test encryption
- Test smart card login

Notes:

You can easily map a certificate to user account with the Active Directory Users and Computers MMC by right-clicking the user -> tasks -> Name Mappings.

However, this isn't really a practical solution because it doesn't scale. Programmatically we can assign a certificate to a user by updating the UserCertificate property of each user account. The best way to do this is with the ldifde tool. When importing the certificate you need to really watch the ldifde syntax:

- replace \n with CRLF
- need :: at the beginning of the certificate data
- spaces before each line

We want to import just the data of the PEM formatted certificate.
See Appendix 4 for an example .ldf file for importing a user certificate.
Keep in mind that UserCertificate is the key property but UserSMIMECertificate is the one that will change when you "Publish to GAL" from Outlook (KB815623). The userCertificate property is used to encrypt messages to be sent to you not the userSMIMECertificate like you'd think.

Another gotcha is that Outlook needs to know that the CA that issued the user certificate is trusted before user certificates will work. As with most certificate work, you won't really get any meaningful error messages to help you debug this.

Now all of this has only dealt with the Public Key of the user. The private key, which was created when the user requested the certificate, will need to be on the machine from which the user wishes to encrypt or sign email. This should be done by converting the certificate into PKCS12 format and then installing that certificate just like in the Phase 2 portion of this document.

Interesting to note that when encrypting email you encrypt with the recipients public key, therefore only their private key can decrypt so how would you view that email in sent

items? Outlook 2003 (at least) handles this by saving a copy of the message encrypted with your private or public key so you can view items you've sent.

Conclusion:

The bottom line is that AD will work for advertising users public keys to encrypt and sign mail using Outlook however we cannot do anything in the way of managing private keys. There is a property called userPKCS12 that has the potential to store the public/private key pair, however I could find no applications that actually used this attribute. That's the real problem with the certificates, sure you can shove the certificate data into an attribute associated with the user. However the problem arises because applications have to be told where to go look for the certificate data. It seems to me, at least at this point, that the applications are what will slow down a PKI deployment.

Notes on using smart card certificates issued by subordinate DEV CA
The whole process of using smart cards to login is so much more complex than encrypting an email so I broke it out into its own section. The reason for the extra complexity is the backend processes that need to occur, such as getting a certificate especially for the Domain Controllers providing the authentication service.

The current version of OpenSSL/OpenCA won't really do what we need it to for Windows smart card integration without hacking it up (have to do this to issue an appropriate DC cert). Since hacking up your PKI infrastructure code is generally a bad idea, we'll test with a newer version of OpenSSL/OpenCA.

Following KB articles are used as reference:
http://support.microsoft.com/default.aspx?scid=kb;en-us;295663
http://support.microsoft.com/default.aspx?scid=kb;en-us;281245
http://support.microsoft.com/default.aspx?scid=kb;en-us;291010

The first thing you will need to do is to configure your Domain Controllers to trust the Root CA as well as the CA issuing the user and DC certificates. You can accomplish this with Group Policy. Next you will need to import the Root CA (in base64 or PEM format) into the NTAuth Store; follow KB article 295663 for instructions on how to do this using certutil.exe. Then you will need to request a DC certificate using the certreq.hta (from http://www.w2k.vt.edu). Next, install that certificate using certreq.hta (copy and paste text in PEM format to text box).

> [Note: After you have done this LDAPS should be enabled on this DC (test w/ ldp.exe connecting to 636)]

Now the user certificate:

1. Request the user certificate from the new VT PKI website.

2. Install the user certificate in PKCS12 format on the smart card using Netscape. There must be a way to do it with IE but I couldn't find it.
3. Get that cert in base64 (pem) format and import that into the userCertificate attribute on the user object using ldifde.

Some "gotcha's" here are that the workstation you wish to login to will need to be able to access the Certificate Revocation List referenced by the user certificate. The workstation must also trust the Root CA as well as the CA issuing the user certificate (this can easily be accomplished by Group Policy).

Conclusion on Smart Cards:
While I was able to make this work, in no way do I recommend that we even attempt this in a production environment. There are too many variables beyond our control as well as too many little things that I consider "hacks" involved in the process. Smart Card login is interesting, however it is also fairly complex and there are lots of places where mistakes can be easily made. The bad thing about these mistakes is that often times there are no good error messages available to help you debug the problem. Hopefully this is where the Microsoft Certificate Services will really add some value. The Certificate Services from Microsoft should handle most of the complex procedures/hacks for us in a controlled manner.

IV. Microsoft PKI Interoperability (Active Directory and Microsoft CA)

Zeb Bowden
01/05/2005
v.1.3

Phase 1. Set up subordinate CA

Tasks:

- Receive correctly formatted subordinate CA cert from the VT DEV OpenCA
- Install W2K3 enterprise edition and Virtual Server 2005
- Enable PKI Services on both online and offline subordinate CAs
- Install IIS on online CA

Notes:

The PKI model tested was to have an offline subordinate CA that only issues one certificate to an online "sub-subordinate" CA. The online CA will issue all the certificates that will actually be used by end-users (machines and/or actual users). Rather than use two boxes for this, Virtual Server 2005 was used to host the offline CA on the online CA machine. Not only does this reduce the cost of the PKI, it also enables portability of the offline CA. Both the online and offline CA's were running Windows 2003 Enterprise Edition. The online CA machine is a member of cntrlsrvs-pilot while the offline CA is configured to be a standalone workstation.

Configuration of the offline CA:
After Windows 2003 Enterprise Edition was installed and secured Certificate Services was installed on the offline CA. I used a Microsoft checklist entitled "Checklist: Creating a certification hierarchy with an offline root certification authority" located at: http://www.microsoft.com/resources/documentation/WindowsServ/2003/standard/proddocs/en-us/sag_CS_Checklist_Offline.asp. Follow these steps to complete the Certificate Services setup wizard.

1. In the first window of the Certificate Services wizard select stand-alone subordinate CA.
2. Next for the Common Name: I put offlineca and then for DN suffix I used "O=Virginia Polytechnic Institute and State University,C=US,DC=vt,DC=edu". The DN suffix needs to meet the requirements set forth by the CA that will issue this certificate (DEV VT OpenCA).
3. I left the default values for the Certificate DB settings and then saved the certificate request as a file.
4. You should get a message about the installation being incomplete (because you have to get the request signed and installed).
5. You should also get a message about IIS not being installed, this is normal for the offline CA.

Install the DEV root VT CA certificate on the offline CA machine. Next, get the certificate request signed (using steps in previous PKI documentation) and install the certificate using the Certificate Authority MMC. I got an error about cannot verify cert chain because revocation server offline. You can ignore this for now by following: http://support.microsoft.com/?kbid=825061. However, a better solution is to manually install the DEV root CA's CRL into the "Intermediate Certification Authorities" store. You need to have this installed to complete the setup of the online CA.

Now we need to configure the CRL Distribution Point (CDP) and Authority Information Access (AIA) extensions so that the certificates issued from the offline CA will point to an online location. (http://support.microsoft.com/?kbid=271386). To modify these extensions:

1. Open the Certification Authority MMC
2. Right click the CA (onlineca) and select properties
3. Choose the Extensions tab
4. Select CDP or AIA in the "Select extension" select box.

For the CDP extension:

1. I first removed all entries except the first one which points to the local file system. If you select the "Add" button it will bring up the "add location" window. In this window you will be able to choose different variables to construct a proper location string. Variables appear in <>'s.

2. Next I added an online location to download the CRL via http:

    a. Select the add button which will bring up the "add location" window, enter: "http://cdp.w2k-pilot.vt.edu/crls/<CAName><CASuffix><DeltaAccepted>.crl".
    b. Check the box titled "Include in the CDP extension of issued certificates".

3. Next I added a location that the CRL will be published to in Active Directory:
    a. Select the add button which will bring up the "add location" window, enter: "ldap:///CN=<CATruncatedName><CRLNameSuffix>,CN=<ServerShortName>,CN=CDP,CN=Public Key Services,CN=Services,DC=w2k-pilot,DC=vt,DC=edu".
    b. Check the box titled "Include in all CRLs. Specifies where to publish in the Active Directory when publishing manually".

In the AIA extension:

1. Select the add button which will bring up the "add location" window, enter: "http://cdp.w2k-pilot.vt.edu/aias/<CAName><CASuffix><DeltaAccepted>.crt"
2. Check the box titled "Include in the AIA extension of issued certificates".

Now set the interval at which the CRL will expire by right clicking on "Revoked Certificates" in the Certification Authority MMC and go to properties. I set the CRL publication interval to 1 year. Uncheck the box next to Delta CRLs because we don't want to publish delta CRLs for the offline CA. To force publication of CRL right click "Revoked Certificates" -> all tasks -> publish. You can also use a Resource Kit tool called certutil.exe to perform this operation, "*certutil.exe –crl*" is the command to issue.

I then created an alias in DNS called cdp.w2k-pilot.vt.edu that points to the IP address of the online CA. Using IIS on the online CA, I created website with crls and aias folders. Next I got the certificate and certificate revocation list from the offline CA and placed the CRL in the crl folder and the certificate in the aias folder. The certificate and revocation list are located in the following directory on the offline CA: "windows\system32\certsrv\certenroll". The revocation list will be named <ca name>.crl and the certificate will be the only file with a .crt extension in the directory.

Now publish DEV VT OpenCA certificate to the Root CA store using "*certutil -dspublish -f filename.crt RootCA*", where filename.crt is the file containing the certificate. Next repeat this step for the offline CA certificate. Finally, publish the CRL using the following command: "*certutil -dspublish -f filename.crl*" where filename.crl is the certificate revocation list of the offline CA.
Note: You need to be logged in as a domain administrator and you may need to wait for replication here (or you can add -dc drclay if you know it's been replicated to that dc)

Configuration of the online CA:
Log onto the machine as an enterprise administrator and install Certificate Services. Select Enterprise Subordinate as the CA type. I changed the key length to 2048 and the common name to the name of the online CA machine (onlineca). The certificate DB and log files I configured to be placed in e:\CertDB and e:\CertLog respectively. Next, save the request to a file and transfer that request to the offline CA via floppy. Since IIS was already installed because of Virtual Server being installed a message pops up asking if it's alright to enable ASP pages. I chose yes because the Certificate Services website requires ASP.

Now get the request signed by the offline CA. I had a great deal of trouble here because the offline CA couldn't get the CRLs. Make sure the CRLs are installed in the machine store on the offline CA.

After you have the signed certificate request, use the Certificate Authority MMC to install it and complete the Certificate Services Installation. The online CA complained about 2 things:

1. Couldn't find the network path. To fix this I just retargeted the MMC and entered the NetBIOS name and it worked.
2. Still couldn't find the revocation list. To fix this, install the DEV root CA CRL into the Intermediate CA store of the online CA.

Now set the interval at which the CRL will expire: right click on "Revoked Certificates" and go to properties. I set the CRL publication interval to 1 week and the Delta CRL publication interval to 1 day. In production these could probably be much longer. All the other settings for the online CA should be correct. You shouldn't need to modify any of the CDP/AIA settings because we are running IIS on the online CA and it will default to that location. After the Certificate Services are running and the Domain Controllers in the forest find out about it they will all get Domain Controller certificates on the next group policy update.

On the online CA I noticed event id: 80 (Warnings) about not being able to publish these Domain Controller certificates in Active Directory. This is because the online CA machine needs to have permissions to Read/Write the userCertificate attribute on all accounts that wish to receive certificates from it and have that certificate published in AD. Since the online CA machine is in a child domain only DCs in that child domain will be able to have their certificate published in AD unless we modify the permissions of every domain controller (give the machine account onlineCA$ read/write on userCertificate). The onlineCA machine is a member of the cntrlsrvs-pilot\CertPublishers global group, it would be best if it were a member of w2k-pilot\CertPublishers. However, since CertPublishers is a global group this is not allowed.

Conclusion:
Setting up the subordinate Certification Authority is the most complex and important part of setting up a Microsoft PKI. Much of the processing/setup takes place behind the scenes which eases the burden on the administrator somewhat. The negative aspect of this is that it's doing things we don't necessarily know about. Several Certificate Templates are enabled by default (ex: Domain Controller Certificates) which is not necessarily a bad thing, it was just unexpected.

A very important thing to keep in mind is that Microsoft Certificate Services really cares about checking revocation lists. In the event that we do decide to use a Microsoft PKI we will need to bump up the priority of CRLs issued from the VT OpenCA because the infrastructure will break down if any CRLs are not available and kept up to date. These are some of the hardest errors to debug as well, but the use of tools like certutil and capimon help out tremendously here.

Hosting a Virtual Server on the online CA worked well in my opinion. I see no reason why we wouldn't want to take advantage of virtualization technology in this situation.

Another thing to note about this is that the location of the online CA does matter. I think we would be better off having it at the root level since the main impact of this CA will be on user accounts. From what I've seen this is not necessary, it will just make management easier and possibly could improve performance.

Finally, permissions and processes were outside the scope of this project so I worked under the idea that Enterprise Administrators would handle all certificate issues. The Microsoft PKI is setup with finer granularity when it comes to permissions in the event

we wanted to take advantage of that functionality. This should make it easier if we wanted to delegate PKI responsibilities to different parties. This document "Defining PKI Management and Delegation" provides more information about delegation and roles: http://www.microsoft.com/resources/documentation/WindowsServ/2003/all/deployguide/en-us/dssch_pki_sshh.asp

Phase 2. Use a web/SSL certificate issued by online subordinate CA

Tasks:

- Set up IIS 6.0 web server on Windows 2003 Web Edition
- Get a web server certificate from the online CA
- Assign the certificate to the web server from IIS
- Test SSL via IE 6.x
- Determine enabled features

Notes:

One nice feature of IIS is that it can generate a Certificate Request (CSR) for you. This hides some of the tedious management tasks involved with setting up an SSL enabled web server. As we saw in the previous portion of the PKI project, when requesting certificates directly from the OpenCA CA we can not take advantage of this feature. However, when using a Microsoft Subordinate CA not only can we allow IIS to generate the CSR, we can also interactively have the request signed and installed. A caveat is that the user who is generating the request (i.e. the user logged into the web server) must have enroll permissions on the "Web Server" certificate template. The entire process: generate request, get the request signed, and install the certificate; literally was completed in a matter of seconds.

After the certificate is installed and SSL is enabled on the web site, users may access the https version of your website. As long as they trust the DEV Root CA they will trust the certificate installed on the IIS server. Our IIS server was a member of a domain; therefore it had all of the necessary certificates installed in order to properly build the certificate chain. If the IIS server were not a member of a domain we would need to make sure the DEV Root CA, offline CA, and online CA certificates are installed as well.

To test the chain being passed we can use OpenSSL: "*openssl s_client -connect <server>:443 -prexit –showcerts*". You should see 3 certificates in the chain: the web server, the online CA, and the offline CA.

The biggest problem with certificate revocability in relation to web servers is the fact that it isn't turned on by default in most browsers. Not only is it not turned on by default, but this behavior doesn't appear to be configurable via Group Policy. Revocation does actually work if it is turned on. It's important to keep in mind that the act of just revoking a certificate does nothing. An updated CRL must be published before the revocation is effective.

Another problem with CRLs is caching. The client must go get the newly published CRL before revocation is effective. Unfortunately CRLs are cached by the clients and there is no control over this caching (http://www.microsoft.com/resources/documentation/WindowsServ/2003/standard/proddocs/en-us/sag_CS_SrvAdCRL.asp). Therefore, you can't be sure the clients will know about a newly revoked certificate until the current CRL has expired. Issuing CRLs with a short validity period is a workaround for this however a way to publish a CRL that is effective immediately is what is desired.

Conclusion:
The combination of IIS and Microsoft Certificate Services is hard to beat. The ease of use is as good as one could hope. Just about anyone could have an SSL enabled IIS web server up and running in a matter of minutes. I'm not sure if that's a good thing or not but it is really that simple. We could create a security group and only allow members of that group to receive web server certificates automatically to help with management.

As far as revocation goes, it works but I wouldn't classify it as reliable. Given the fact that revocation checking is off by default, and that very few people turn it on, I wouldn't recommend relying upon it's functionality in the case of web server certificates.

Phase 3. Use a user certificate issued by a subordinate CA

Tasks:
- Determine mechanism for importing cert into the AD
- Get a user certificate from subordinate CA
- Test digital signing
- Test encryption
- Test smart card login

Notes:
For user certificates, auto enrollment seems to be the best way to issue certificates because all of the ground work/processes are already in place beginning with Windows XP. By this I mean that if an auto-enrollment template is in place and the user has been granted the Autoenroll permission all they have to then do is login to a machine with their domain account. They don't have to be prompted for anything; the act of logging in is enough to give them a useable certificate. It's not instantaneous but it is relatively fast.

To create the Auto Enrollment certificate template:
1. Open Certificate Templates MMC and duplicate the User Template and name it userAutoenrollment (left default values)
2. I then created a group called testuserenrollment and gave only that group Enroll and Autoenroll (and read) permissions on the template -> Make sure you give Authenticated Users at least Read on the Template (http://support.microsoft.com/?kbid=283218)

3. Then add that Template to the CA using the Certification Authority MMC by right clicking Certificate Templates and then New -> Certificate to Issue.
4. Then delegate onlinemsca$ read/write userCertificate property on the IAD OU so that this will be inherited by all objects in the IAD OU.
5. For computers, by default it will try to Autoenroll automatically because that is the default policy setting. Users seem to as well, however you can't see auto enrollment settings until you install the 2003 Server adminpak. For Autoenrollment to work properly, you will also want to enable the other 2 options (Renew expired certificates and Update certificates). You can modify these User Autoenrollment settings in the following location in the group policy object: User Configuration -> Windows Settings -> Security Settings -> Public Key Policies: Autoenrollment Settings. I created a new GPO and applied it to the IAD OU.

After the certificate template and Autoenrollment configuration is complete all that's left is to wait for replication and then login to a machine with a user that has Autoenroll permission on the new template. Sometimes you may have to login, do a "*gpupdate /force*", and then logoff and back on. At the time the certificate is issued to the user, it is also published in Active Directory to be used by AD aware applications (ex: Outlook).

At this point signing and encrypting email works as expected. You should be able to open Outlook and start signing and encrypting email messages. To encrypt messages you need the public key of the recipient. This means that the recipient would also need to have been issued a certificate and that certificate would need to be published in Active Directory.

One thing I did notice was that every time I logged into a new machine I received a new user certificate. The use of roaming profiles should remedy this and it may be something we would want to look into. Potential problems do exist here, however the use of smart card certificates can solve this. If we ever did decide to just use regular user certificates, issued via Active Directory/Autoenrollment we will want to look into this more closely and document a solution involving roaming profiles. I was able to create a situation where a user had several certificates published in Active Directory, where that user could not read encrypted email messages sent to him. Presumably this is because the sender encrypted the message with the wrong public key.

Another way to issue user certificates is an on-demand type of method where the user is required to login to a website and request a certificate. This is all built in to the Certificate Services and works as well. I didn't test this with user certificates directly; however I did use this when testing smart cards. Another configuration option would allow the user to confirm whether or not they want an auto enrollment certificate or to inform them when they receive one. Once we move our production web server to IIS 6, it may be possible to incorporate some certificate services functionality into Hokies Self Service.

It is also possible to configure the CA so that certificate requests must be approved before they are actually issued. This would give some more control over the number of

certificates out there however I think it would come down to us approving every request that came in (unless there's a policy in place).

Conclusion:

One important thing to keep in mind when dealing with Microsoft Certificate Services is once you've added the certificate template to the CA, ACLs are the only mechanism for controlling who can get what type of certificate. Certificate Template ACLs can be modified using the Certificate Templates MMC. I was caught off guard by this and was somewhat concerned at this behavior. As soon as the group policy update on Domain Controllers occurs after the online CA is up and running, certificates will be issued to them. Other templates, such as Basic EFS, are also available to administrators and other privileged accounts by default.

User certificates work quite well assuming you meet the necessary criteria (Template, ACLs, etc). The integration and lack of user input provided by the Autoenrollment functionality is excellent. Not only is this a good mechanism to get certificates out to users but it should also reduce support costs because everything works behind the scenes and user education should be minimal. A negative view of this is that the users won't even know they have a certificate and therefore won't realize they need to protect their private key (or even that they have a private key).

Notes on using smart card certificates issued by a Microsoft Subordinate CA:

The way we should issue smart card certificates is in an "on behalf of" scenario, just like Hokie Passports. The user would go to an office, prove their identity, and then an officer would request (via web interface) a smart card certificate to be created and placed on a smart card. The officer in this situation would be an "enrollment agent" and could request and receive a certificate for anyone. The machine the enrollment agent used would be known as the enrollment station. I chose to have an Enterprise Administrator as the enrollment agent for testing but this role can be delegated or assigned to another user or users.

The first step in setting up the smart card login environment is to first get an enrollment agent certificate for our enrollment agent. Setting up the template is done just like in the user certificate section above. The difference will come in at the request/install phase. To get the certificate login on the enrollment station with the enrollment agents credentials and go to the CertSrv website (https://onlinemsca/CertSrv) using Internet Explorer.

1. From here you will want to select "Request a Certificate", "advanced certificate request", and then "create and submit a request to this CA".
2. This will bring up a certificate request screen; you should select the enrollment agent template and complete the certificate request.
3. After you complete this request process the installation of the certificate will be handled by the browser.

This user is now ready to begin issuing smart card certificates to other users. First however, the enrollment station should be equipped with an appropriate smart card reader. In addition the smart cards need to be configured with a PIN that the enrollment agent knows. The configuration of PINs on smart cards is usually handled by software provided by the smart card vendor.

To issue a smart card certificate to anther user the same CertSrv website will be used.

1. Login to the CertSrv website with the enrollment agents credentials.
2. Select "Request a Certificate", "advanced certificate request", and then "Request a certificate for a smart card on behalf of another user by using the smart card certificate enrollment station".
3. This will bring up the certificate request screen for the smart card certificate.

   Things to note on this screen are to choose the correct template (smart card user) as well as the correct CSP provider. The CSP provider I tested with was Schlumberger; this will need to correspond to the type of smart cards you are using. The smart cards that will work with Windows XP without having to install the drivers are listed here: http://www.microsoft.com/technet/prodtechnol/winxppro/maintain/certenrl.mspx#EDAA.
   After you select the appropriate user and make the request you should be prompted to enter the PIN on the smart card. After you enter the correct PIN the public/private key pair will be written onto the smart card and after replication occurs, you should be able to use it to login. The user who gets this smart card should change the PIN as soon as possible; perhaps even at the time they pick up their card.

Since the "Smartcard User" template was selected we can use this certificate not only for logins but also for encrypting and signing email. It would be possible to add additional functionality (EFS for instance) to the smart cart user template so that users would only have to deal with one certificate and it would be stored on the smart card.

Certificate revocation does work, however it doesn't work as expected and in my opinion it is not a reliable way to disallow authentication (or any other user of the certificate). The reason for this is the caching of the CRLs. If CRLs are cached then you can't be sure the machines in your organization are using the updated CRL until after the expiration date of the old revocation list. If it is desired that we have an immediate way to stop a user from using a certificate then CRLs do not provide a solution. A quick hack for a revoked smart card certificate, assuming the user has logged into the workstation before and their credentials are cached is to unplug the network line and then login. After login you can then plug the line back in and you're authenticated and have access to the resources.

Some things I noticed in the smart card testing:

- We will almost certainly have to install smart card drivers on systems wishing to use smart card logins. Smart card technology is developing and changing

rapidly. Because of this, all of the smart cards contained in the link are old and quickly becoming obsolete. Most, if not all, of them are not even being produced any longer.

- Certificate Revocation Lists are critical for smart cards to work. Each time a smart card login occurs the CRLs for each certificate in the chain are checked at the Domain Controllers. If any of these CRLs are unavailable or expired authentication will fail. To guarantee that the CRLs are available for the OpenCA CA I would recommend installing that CRL manually in the certificate store of each Domain Controller.
- A relatively new technology that Windows XP and Windows 2003 support is called Delta CRLs. This allows for potentially faster CRL checking because only the changes made since the last CRL was published are checked. This is not supported on Windows 2000 so our domain controllers have to check the complete CRL each time.
- Renaming users that use smart cards is a bad idea, in fact, after a rename (change of the user's UPN) the user will not be able to authenticate any longer (with their smart card). Since we don't do renames this isn't a big deal, but it is worth noting. Moving users did not have any noticeable negative effects. The user's DN is used in the subject of the certificate and this is not updated in the certificate when a user is moved. Therefore the DN in the user's certificate will not be the actual DN of the user. For instance if I work in the CS department, my DN is CN=user,OU=cs,OU=vt,DC=w2k,DC=vt,DC=edu. However if I change departments and thus change OUs my DN now looks like: CN=user,OU=newdepartment,OU=vt,DC=w2k,DC=vt,DC=edu. If someone were to examine my certificate it would still look to them as if my user account were located in the CS OU.
- In addition to the GPO settings I've discussed above there are also two Security Options relating to Interactive Logons that can be configured to further secure the workstation: "Require Smart Card" and "Smart Card Removal Behavior". There is also a setting on the user's Active Directory object that can require them to login with a smart card.
- Smart Card login information is cached just like regular username/password information is cached. This behavior can be configured by editing the following security option: "Interactive logon: Number of previous logons to cache (in case domain controller is not available)".

Conclusion on Smart Cards:

Using smart cards in a Microsoft PKI environment seems much more reliable than in the OpenCA scenario we tested previously. It still does not eliminate the CRL problems because we will still have to check the revocation list provided by the Root CA because it's at the top of our certificate chain. The management of certificates was much easier and the overhead involved with getting a certificate was significantly reduced by using the Microsoft PKI.

The most challenging part of Smart Card Certificates (and just regular user certificates for that matter) is not technical it's going to be in the policies/procedures. It is possible to configure the Microsoft Certificate Services to allow anyone with a smart card reader to issue themselves a smart card certificate. This is a simple solution, however it is not the most secure and it is not a recommended way to issue smart card certificates. According to Microsoft's Best Practices for Implementing a PKI: "The most secure way to initially enroll user certificates is to do a face-to-face authentication at the registration authority and store user certificates on hardware tokens. This provides the highest level of assurance, but also the highest cost of deployment." See http://www.microsoft.com/technet/prodtechnol/windowsserver2003/technologies/security/ws3pkibp.mspx for the entire Best Practices document.

It is important to realize that even though smart card technology is good there is still the tradeoff between security and usability. Training and education would be a key part to rolling this out, especially for the administrators of the PKI.

Appendix 1: Microsoft CA vs. OpenCA

	Microsoft CA subordinate	*OpenCA CA*
Initial Cost	Medium	Low
Estimated Support Cost	Low	Medium[1]
Support[2]	Available	Not Available
Integration into Existing AD Infrastructure	Yes	Requires Customization[3]
Level of automation that would need to be built	Very Little	High[4]
Staffing requirements[5]	?	?
Required assurance level	?[6]	High Assurance
Ease of use (user)[7]	Easy	Novice
Ease of use (admin)	Novice	Expert

[1] The main source for the increased support cost in the OpenCA scenario is the lack of delegation as well as the lack of automation.
[2] Support here refers to conventional, paid support. For example: Microsoft PSS would provide support for a Microsoft PKI environment. The OpenCA PKI relies on the Open Source community.
[3] The level of customization required to integrate an OpenCA PKI into AD will depend on the desired amount of functionality. For a Microsoft CA the customization should be minimal regardless of desired functionality.
[4] All tasks that we wish to be automated will need to be built for an OpenCA PKI.
[5] Staffing requirements are indeterminate at this point
[6] The required assurance level of a Microsoft CA is yet to be determined. There are no technical restrictions on this.
[7] The ease of use for the Microsoft CA assumes we are using the built-in features (autoenrollment, web enrollment, etc) to make it easy for the user. An OpenCA PKI would require the users to know more about certificates but it may be possible to recreate the features that a Microsoft PKI would have, thus making it easier on the user.

Appendix 2: Functionality comparison, specifically related to our Windows Active Directory environment

	Microsoft CA	*OpenCA CA*
Web Server Certificates	Excellent	Good
User Certificates	Good	OK, requires customized scripts for creation and management[8]
Smart Card Certificates	OK	Bad, requires customized scripts for creation and management[9]
Email Security[10]	Good	Good
Active Directory Publication	Good	OK, requires customized scripts for creation and management
Certificate Revocation	OK	OK, requires manual imports of revocation lists

[8] The steps needed to get user certificates into the AD seemed very "clunky", I did not gain a high level of confidence for how well this would work on a day to day basis.

[9] Smart Card Certificates issued directly from an OpenCA CA and imported into AD just did not work reliably. This was largely due to the fact that revocation lists are a requirement for smart card authentication and currently the priority we place on CRLs is not high enough to support smart cards in AD. In addition to the authentication not working reliably, the customization steps seemed dangerous and nearly unmanageable.

[10] In our testing encrypting and signing of email worked the same in Outlook regardless of which PKI was used. This is due to the fact that Outlook/Exchange does not manage the certificate(s); rather they use the certificate information provided by Active Directory.

Appendix 3: Hardware and Software requirements

Hardware and software requirements are derived from the testing environment. There are other ways to configure the hardware and software of these machines that would provide a working PKI.

Software:
Operating System: 2 copies of Windows 2003 Server, Enterprise Edition
Additional Software: Microsoft Virtual Server 2005, Standard Edition

Hardware:
The minimum hardware requirements will be dictated by the requirements to run Virtual Server 2005.
http://www.microsoft.com/windowsserversystem/virtualserver/evaluation/sysreqs.mspx

Requirement	Virtual Server 2005 Standard Edition
Minimum CPU Speed	550 MHz or faster; 1.0 GHz or faster recommended
Processor	• Computer with up to 4 physical processors • Celeron, Pentium III, Pentium 4, Xeon, Opteron, Athlon or Duron processor required
Memory	256 MB minimum; additional memory needed for each guest operating system
Hard Disk	2 gigabytes (GB) of available hard-disk space; additional disk space needed for each guest operating system
Display	Super VGA (800 × 600) or higher resolution recommended
Host Operating System	• Windows Server 2003, Standard Edition • Windows Server 2003, Enterprise Edition

I customized a server from the Virginia Tech Dell Premier website and the cost was around $3400. This system was a PowerEdge 2650 with dual processors (2.8 GHz, Xeon), 1.0GB RAM, and a 36 GB drive (RAID 1 + hot spare).

Appendix 4: Example of file to be used with ldifde.exe

Example .LDF file:

dn: CN=plainuser,OU=ward,OU=vt,DC=w2k-pilot,DC=vt,DC=edu
changetype: modify
replace: userCertificate
userCertificate::

MIIGRAYJKoZIhvcNAQcCoIIGNTCCBjECAQExADALBgkqhkiG9w0BBwGgggYX
MIIGEzCCA/ugAwIBAgIBVTANBgkqhkiG9w0BAQUFADCBrjETMBEGCgmSJomT
8ixkARkTA2VkdTESMBAGCgmSJomT8ixkARkTAnZ0MQswCQYDVQQGEwJVUz
E8MDoGA1UEChMzVmlyZ2luaWEgUG9seXRlY2huaWMgSW5zdGl0dXRlIGFuZCB
TdGF0ZSBVbml2ZXJzaXR5MSwwKgYDVQQDEyNERVYgVmlyZ2luaWEgVGVjaC
BDbGFzcyAxIFNlcnZlciBDQTEKMAgGA1UEBRMBMzAeFw0wNDA1MDUxNTIxN
DRaFw0wNTA1MDUxNTIxNDRaMF8xEzARBgoJkiaJk/IsZAEZFgNlZHUxEjAQBgo
JkiaJk/IsZAEZFgJ2dDESMBAGA1UECxMJRW1wbG95ZWVzMRMwEQYDVQQDE
wpQbGFpbiBVc2VyMQswCQYDVQQFEwI4NTCBnzANBgkqhkiG9w0BAQEFAAO
BjQAwgYkCgYEAxbEtbvixWoi/MV9W6J+wZZUg2iqhHlfrdPyw76me8Orbmp0dzTiQ
OT7StxfPfFtx4rj5t6Scf9kTulF50/gSZZBYSBiJfa6snKV2qz3d8EhmSKEflu9gzj4LSNZy
gEcexzOrzcA2muhmfpKNeCX/P0d6Vyqs82Xq+ybqT4Vs5QkCAwEAAaOCAgwwggII
MAkGA1UdEwQCMAAwEQYJYIZIAYb4QgEBBAQDAgSwMAsGA1UdDwQEAwI
F4DApBgNVHSUEIjAgBggrBgEFBQcDAgYIKwYBBQUHAwQGCisGAQQBgjcUAg
IwHQYDVR0OBBYEFKyC82IdKMVYeljuF4ApbfgkzP+OMIGCBgNVHSMEezB5gB
QBCkm23hnVdCbqBCNATQb9wwxaA6FepFwwWjELMAkGA1UEBhMCVVMxEjA
QBgNVBAgTCVZpcmdpbmlhIDETMBEGA1UEBxMKQmxhY2tzYnVyZzEiMCAGA
1UEChMZREVWIFZpcmdpbmlhIFRlY2ggUm9vdCBDQYIBAzAbBgNVHREEFDASg
RBwbGFpbnVzZXJAdnQuZWR1MAkGA1UdEgQCMAAwVAYDVR0fBE0wSzBJoEe
gRYZDaHR0cHM6Ly9yYS5lcHJvdi5pYWQudnQuZWR1L2NhLzA5MS92dGMxc2Nh
L2h0ZG9jcy9wdWIvY3JsL2NhY3JsLmNybDCBjQYDVR0gBIGFMIGCMA4GDCsGA
QQBtGgFAgIBATAOBgwrBgEEAbRoBQICAgEwDgYMKwYBBAG0aAUCAgMBM
A4GDCsGAQQBtGgFAgIEATBABgwrBgEEAbRoBQICBQEwMDAuBggrBgEFBQc
CARYiaHR0cDovL3d3dy5wa2kudnQuZWR1L3Z0YzFzY2EvY3BzLzANBgkqhkiG9w
0BAQUFAAOCAgEAFDB95wTf3Z4EoqJ7xsJ0oDuiROdZzpyLcI1Asyh8+vOBrSPXTe
0KHXV1wObhRIb63MwVlSc2zyLxJAKWXbk3PqXXItcBBAuJwuPktGm9EI3ZyKcT1
WU8pyHY8rJoSrdBsPyfy9KIN8F+SC6MPOFnxdIyInK/h1NDDvqDG8neiVOrOsmb3C
/3kSHbbLa61+801Cvb5LbM/pKWWyK5oi9qUE1P2nYygKxxJUwkrtbd0i0x4MHcPNh
6tFgclL1jOEFh8RMWrkGLjSQ3YBxIjQ/R/Y5hTpUpo+OyvkDITWG5Jj8mCxQpRhJA
zdrMyKEy7SjLJhPtJT9NxPYSgEbsRWCRrQLbRlCpiUIJmBc58CGt7eDL7zuSm8eE/D
gUly5s5w8ySXSDNa52nrNTNEoK3KF8JLihtMf6YDU3oQLLvtU3cvQVxoo3F0OjXbt
CdPyfv/Qf9hwz/uWhXdYpUp9GareLIW9/j6BjriuJfPUIP0cCCHjloyuxu45m7m2zIy7Pn
EF1lZs8KSQynAwyJgE39ULpozO3oJlWtdihwBrsj5p3/D77xiU9hnf+WE5GHNzyDX9
ZfoGhUBbI6UmtGNk2nW3TzpjBhSIGAWemz+3rIP7wBS2l39leQ5Nm3pgx548DKJLS
XVlNxFOfh6gUO0Og++v0QP0wQkhji29kqxy7N5sstX2hADEA

-