USING DIGITAL CERTIFICATES FOR MAX AUTHENTICATION

How to obtain and use a digital certificate that allows you to access MAX services from scripts and batch jobs

If you want to use a program (such as a script or a scheduled task) to do things in MAX.gov, then you need a way for the program to identify itself to MAX. As an interactive user, you would normally use your PIV card or a username and password, but a non-interactive program doesn't have that ability. The solution is to use a digital certificate, which functions much like your PIV card. This document tells you how to do that. By the time you're done, you will know

- What a digital certificate is,
- The steps required to request and obtain a digital certificate for use with MAX,
- How to take care of the digital certificate once you have it, and
- How to use the digital certificate in a program that accesses a MAX site.

If you're interested in a detailed discussion of security and privacy controls that pertain to the use of digital certificates, you can read NIST Special Publication 800-53.[1]

Scope

This document applies to anyone who needs a service ID and a digital certificate to represent it, including MAX staff and agency users.

Policy Reference

Please consult MAX Security Identification and Authentication Policy for a detailed policy statement. Note that as with other MAX documents referenced herein, you will have to have a MAX account to view this document.

Roles and Responsibilities

- Service ID User – that's you. You need a service ID and credentials for it in order to access a MAX system from a non-interactive program, and this document tells you how to do that.
- MAX Point of Contact (POC) – a person on the MAX team who is your advocate in the process, and who can work with MAX staff to complete your request.
- MAX Data Management Team – issues service accounts to the MAX POC.
- MAX Technology Services Team – issues digital certificates for service accounts to the MAX POC based on service account ID and a Certificate Signing Request (CSR) that you will generate.

What is a digital certificate?

In the course of working with systems and web sites, you will run across the term "digital certificate" a lot. Digital certificates are used for a number of different things – the most common use you will see is as an identification and encryption mechanism when you visit a secure (https) web site. Another common place where you'll encounter digital certificates is on your PIV card. Like an HTTPS certificate, the certificate on your PIV card contains an "identity" and a set of encryption keys. Unlike an HTTPS certificate, however, your PIV certificate is marked as being used "to identify a client", rather than "to identify a web site". Other than that, though, they are basically the same thing.

[1] Available as of this writing in draft form at https://csrc.nist.gov/csrc/media/publications/sp/800-53/rev-5/draft/documents/sp800-53r5-draft.pdf.

For our purposes, we can think of a digital certificate as a file that contains, among other things,

- An identity that the certificate represents,
- A unique identifier,
- An expiration date,
- A digital signature that ensures the integrity of the information,
- A reference to the authority that was used to issue the certificate,
- A reference to an authority that can be used to check the validity of the certificate,
- A public key that is used in cryptography[2].

All of the information contained in a certificate is public; that is, it is safe to share it with anyone.

However, associated with the certificate is a private key, which must be carefully guarded. The special thing about PIV cards is that the private key is permanently embedded in the card and can only be used when the card is physically connected and the PIN has been entered. For private keys that are not embedded in a PIV card or a similar hardware device, some file formats (such as .cer, .pem, and .der files) store the certificate and private key in separate files, while other file formats (such as .p12, .pfx, and .keystore files) support storing the certificate and private key in the same file. It is important to know the difference so that you can appropriately protect and avoid sharing any files that may contain private keys.

As we'll see later, you will use a MAX identify certificate by attaching it to a web request in a way that causes MAX to treat your request similar to a request that is authenticated using a PIV card.

> **IMPORTANT**: A digital certificate with its private key are valid login credentials to MAX, and must be treated with care. Be sure to follow the practices below to obtain, protect and use your MAX digital certificate.

Keeping and safeguarding your MAX certificate

Windows and Linux provide different mechanisms for limiting access to your private key. In Windows, keys are stored in a "key store". Each user has a separate key store, and the system has its own store as well. The key store provides a way to keep people from exporting the private key, and it provides an access control mechanism for limiting who can read it. In Linux, the private key is stored in a file, and normal file system permissions are used to regulate which accounts can read it.

Best practices for storing and using the certificate on Windows

The following are recommended practices for working with your MAX certificate:

1. Use a dedicated Windows service account to perform web requests using the certificate, and create the private key and CSR in that account's "key store".
2. Since the private key is configured to be non-exportable (per the instructions below), you can't copy this certificate to another account or machine. If multiple accounts/machines need access to MAX, then request multiple certificates (one for each account on each machine).

[2] See https://docs.oracle.com/cd/E19509-01/820-3503/ggbgc/index.html for a simple explanation, and https://en.wikipedia.org/wiki/Public-key_cryptography for a more extensive discussion of public key cryptography.

Just to review, here's the situation:

1. The target system has your certificate and its private key. This certificate can be used to authenticate to MAX.
2. The target system can't export the private key, so someone who breaks into it (or stumbles upon it) can't take the certificate away and use it for something else.

Best practices for storing and using the certificate on Linux

As mentioned above, the private key (stored in a .key or .p12 file per the instructions below) is the critical piece of information that allows the service ID to log in to MAX. As such, you must protect it from unauthorized use. The following are recommended practices for working with a .key / .p12 file:

1. Make your .key and/or .p12 file readable by the minimum number of entities (users/processes) required for your application to function. Use standard Linux filesystem permissions, ACLs, and/or SELinux contexts to restrict access appropriately.
2. Avoid making any copies of the .key and/or .p12 file. Exclude this file from backups, never commit this file into a source code repository, and never send it via email. If any copies are accidentally created, contact MAX immediately to have your old certificate revoked and a new one issued.
3. Do not use the same key/certificate on multiple machines. If multiple machines need access to MAX, then request multiple certificates (one for each machine).

Getting a MAX Certificate

There are two main steps in getting a MAX certificate:

1. Request a MAX service ID. Note that you can't use your own MAX ID in a certificate like this; it has to be a separate ID (this is a policy requirement).
2. Create a private key and Certificate Signing Request (CSR), and send the CSR to MAX for processing. There are different procedures for creating the private key and CSR depending on whether you're using Windows or Linux. If your program will run on multiple systems, you must request a separate certificate for each system.

If you are not a member of the MAX staff, then your MAX POC will assist you with these steps.

Getting a Service ID

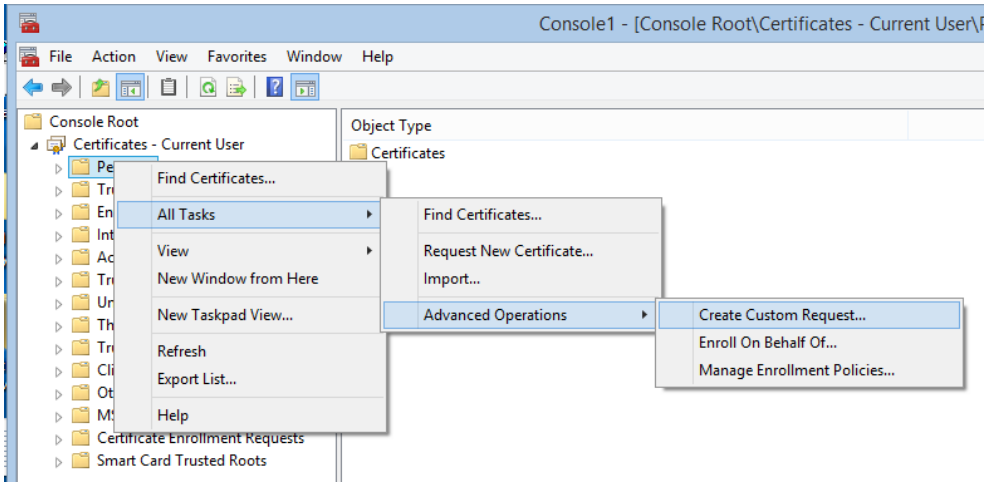
Please ask your MAX POC to request a MAX service ID from the MAX Data Management Team.

> For MAX Staff: Please specify the following in your service ID request to the DMT:
>
> Service ID: (e.g., "S_MYSVC")
> Email: (e.g., "myservice@max.gov")
> First Name: (e.g., "My")
> Last Name: (e.g., "Service")
> Agency: (e.g., "TREASURY")
> Comment: (e.g., "This account will be used for …")

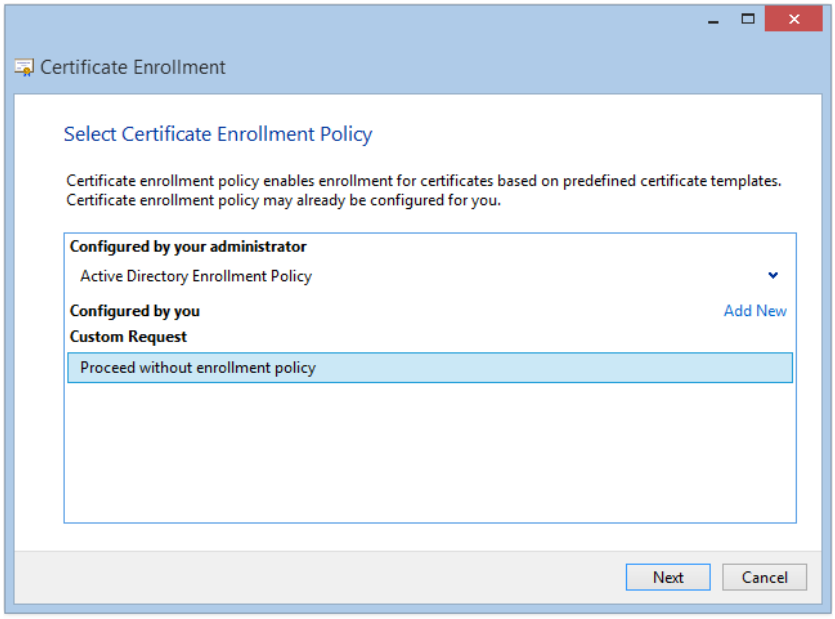
Once you have established a MAX service ID, the next step is to generate a private key and CSR and send the CSR to MAX for processing.

Getting a MAX Certificate on Windows

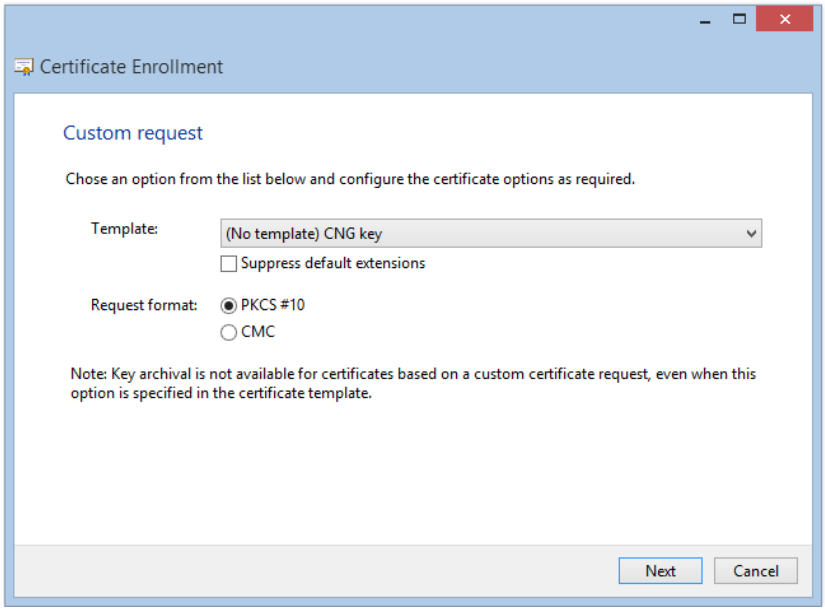
1. Create a private key and CSR on the system where your program will run, in the Windows account that your program will run as. If your program will run on multiple systems or as multiple accounts, you must create a separate private key and CSR for each system/account.
    a. Open the Windows Certificate Manager for the appropriate account:
        i. If your program will run as a normal Windows User, then while logged in as that user, hit the Windows key and type “certmgr.msc” or “Manage user certificates”.
        ii. If your program will run as a Windows Service Account:
            1. Hit the Windows key, type “mmc”, and click “Run as administrator”.
            2. Click “File” -> “Add/Remove Snap-in…”.
            3. Select “Certificates” then click “Add >”.
            4. Select “Service account”, click “Next >” twice, then select the relevant service account.
    b. In certmgr or mmc, select the “Personal” folder, then right-click -> “All Tasks” -> “Advanced” -> “Create Custom Request…”:



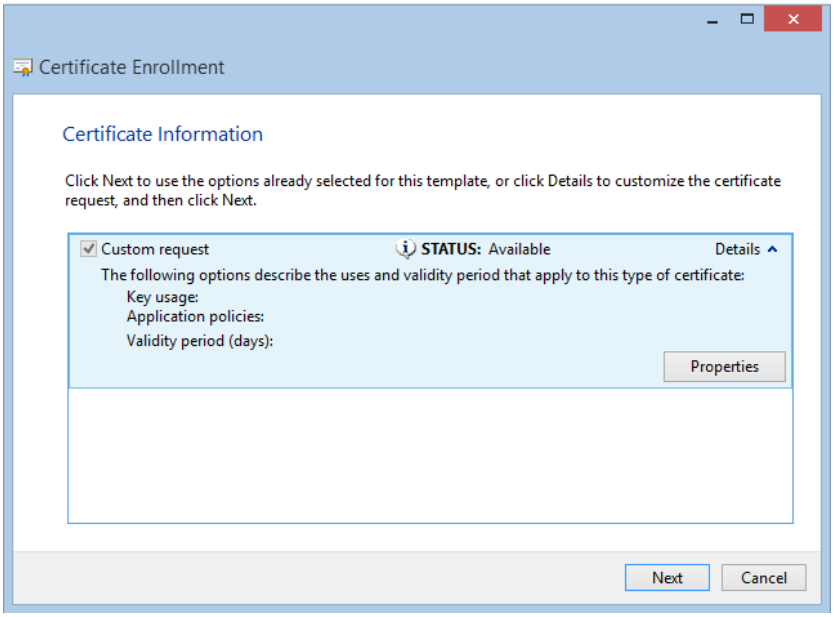
c. Click through the introductory screen(s), then pick “Proceed without enrollment policy”:



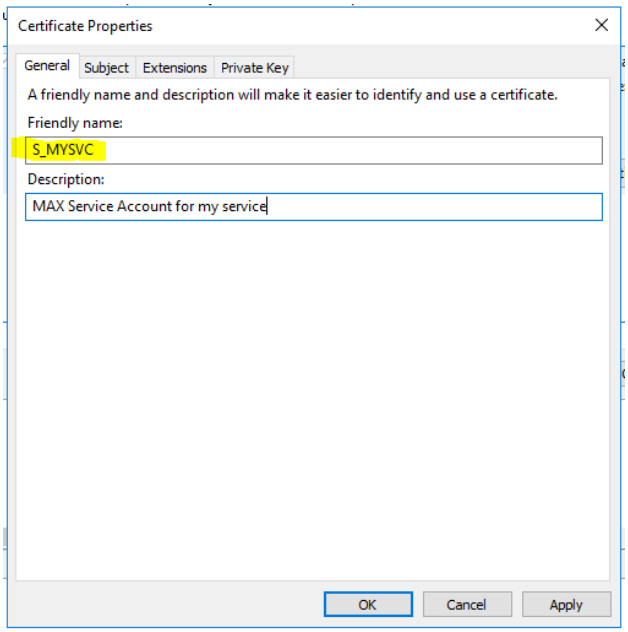
d. On the next screen, pick the options “CNG Key” and “PKCS#10”:



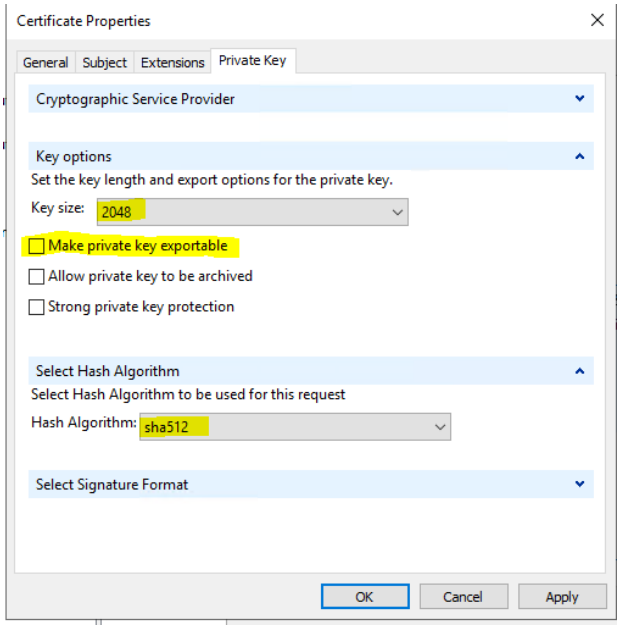
e. On the next screen, expand “Details”, then click “Properties”:



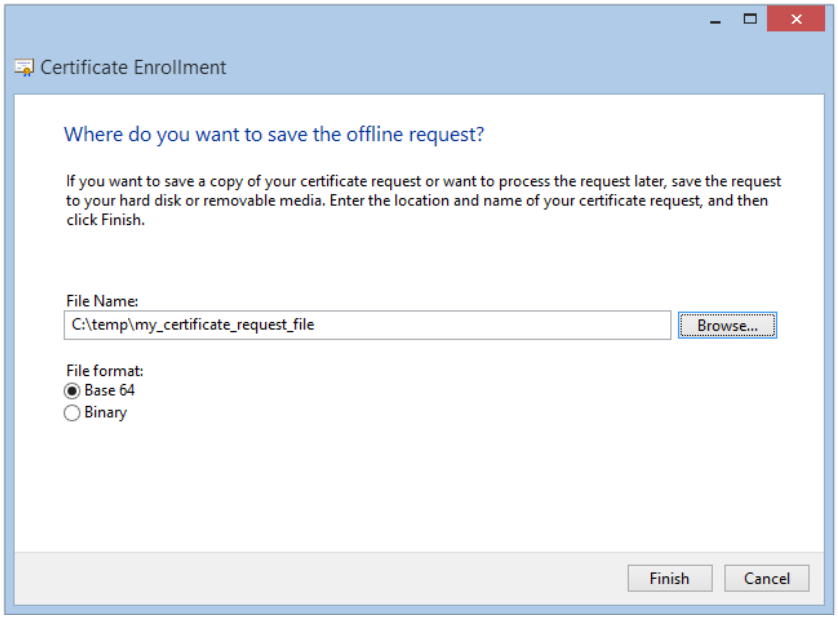
f. In the “General” tab of the Properties window, set the “Friendly Name” to your MAX service ID (e.g. S_MYSVC):



g. In the “Private Key” tab of the Properties window, set the following:
   i. Key size: 2048
   ii. Uncheck “Make the private key exportable”
   iii. Hash algorithm: sha512



h. Select a file location to store the CSR in:



i. Look in the “Certificate Enrollment Requests” folder in certmgr or mmc, and you will see your request. If you do not see a “Certificate Enrollment Requests” folder then right-click on the “Certificates – Current User” folder and click “Refresh”.

2. Open the CSR file using notepad. It will look something like this:

```
-----BEGIN NEW CERTIFICATE REQUEST-----
MIICODCCAaECAQAwADCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEAtr2qObIn
FYpfP2tY+Hot7p/vZHMLSZlRNJldmJQY+AdLulwj228G79TwAvaU78mmKql3a9xY
494icxEG1o3ouYpz04FnVoJkrdl9ZcJz7g+EcQ20tT6WXBT56VsobDDSmupoWWG9
bOwU284MkwCntC5kvyCf7eIJSt6AFpbF+UMCAwEAAaCB9zAaBgorBgEEAYI3DQID
MQwWCjYuMy45NjAwLjIwLgYJKoZIhvcNAQkOMSEwHzAdBgNVHQ4EFgQUCfOc7uFF
k36pBrzspH4q3Gi6hr8wQQYJKwYBBAGCNxUUMTQwMgIBBQwVRU9QUjExMDQyMC5E
Uy5FT1AuR09WDA1EU1xjYXJzb25fc2RhDAdNTUMuRVhFMGYGCisGAQQBgjcNAgIx
WDBWAgEAHk4ATQBpAGMAcgBvAHMAbwBmAHQAIABTAG8AZgB0AHcAYQByAGUAIABL
AGUAeQAgAFMAdABvAHIAYQBnAGUAIABQAHIAbwB2AGkAZABlAHIDAQAwDQYJKoZI
hvcNAQELBQADgYEAaI0xm9VwTkeNJfl1CJBgbJje+RPgTO1D1DTJoAZCL3moAxRi
mXpmigWNDeeWPjSj8nl8v6zutebnFsB8sAvgOxnkDDyK58HWUr3BraorcPJ61T5q
AIpttmMfx18bHPw8h0lfpbltR6uvYsGeSoW9g0ENKuF7jXNCRIruhj+9jNw=
-----END NEW CERTIFICATE REQUEST-----
```

3. Copy the contents of the CSR file from notepad and paste it into an email to your MAX POC.

4. You will receive your certificate in return. Copy the certificate contents from your email, open notepad, paste the contents, then save as a “.cer” file.
   You may also receive a “Certification Authority” (CA) certificate, which should be pasted into a separate “.cer” file.

5. Use certmgr or mmc to import your certificate:
   a. Right-click on the “Certificates” object under “Personal”, then select “All Tasks” -> “Import”.
   b. Browse to and select the .cer file for your certificate (not the CA certificate).
   c. After importing, verify that the certificate is properly associated with the private key:
      i. Double-click on your certificate (which can be identified by its “Friendly Name”).
      ii. At the bottom of the “General” tab, you should see:
      “You have a private key that corresponds to this certificate.”
   d. If you received a CA certificate, then right-click on the “Certificates” object under “Client Authentication Issuers”, then select “All Tasks” -> “Import”.
   e. Browse to and select the .cer file for the CA certificate.

6. If your program will run as a normal Windows User then you can test that the certificate is working properly by opening a web browser and going to:
   https://serviceauth.max.gov/cas-cert/login?service=https://mds.max.gov/authenticated/userinfo.aspx

At this point, you have a complete MAX identity encapsulated in the certificate store, and you can skip to the “Working with your MAX certificate” section below.

Getting a MAX Certificate on Linux

Use the following command to create a private key and CSR on the system where your program will run and as the user that your program will run as. Replace “S_MYSVC” with your MAX service ID. If your program will run on multiple systems or under multiple users, you must create a separate private key and CSR for each system/user.

```
(umask 277 ; openssl req -newkey rsa:2048 -nodes -keyout ~/.max.S_MYSVC.key -out
~/max.S_MYSVC.csr -subj '/CN=request')
```

Note that the ".key" file contains the equivalent of a password and must be protected accordingly. The above command will create the file such that it is only readable by your user account, but you must ensure that the permissions are never altered to allow other users to read the file. This file must never be committed into any source code repositories, and should be excluded from backups; It is easy to generate a new key and certificate if this one is lost, so there is no benefit to keeping backup copies of it. If this file is ever accidentally copied, committed, or otherwise exposed, generate a new key and CSR and contact MAX immediately to have your old certificate revoked and a new one issued.

Copy the contents of the CSR file and paste it into an email to your MAX POC. After sending the CSR, you may delete the CSR file, as it is no longer needed:

```
cat ~/max.S_MYSVC.csr
rm ~/max.S_MYSVC.csr
```

You will receive your certificate in return, which you can copy and paste into a “.max.S_MYSVC.cer” file. You may also receive a “Certification Authority” (CA) certificate; You may ignore this certificate, as it is not needed in Linux (it is only needed for some use cases in Windows).

You can test that the certificate is working properly by running:

```
curl -L --cert ~/.max.S_MYSVC.cer --key ~/.max.S_MYSVC.key
'https://serviceauth.max.gov/cas-cert/login?service=https://mds.max.gov/
authenticated/userinfo.aspx'
```

Some Linux software may require the private key and cert to be combined into a single "PKCS#12" file (".p12" or ".pfx" file) instead of being stored in separate "PEM" formatted ".key" and ".cer" files. If your use case requires this then use the following command to generate a PKCS#12 file from the separate files (you may delete the separate files after generating the .p12 file):

```
(umask 277 ; openssl pkcs12 -inkey ~/.max.S_MYSVC.key -in max.S_MYSVC.crt -certfile
~/.max.S_MYSVC.ca.crt -export -out ~/.max.S_MYSVC.p12)
rm ~/.max.S_MYSVC.key ~/.max.S_MYSVC.cer
```

Since the ".p12" file contains the private key, it must be protected as described above for the ".key" file.

Working with your MAX Certificate

Once you have a MAX certificate, you can configure/code programs/scripts to use it to authenticate to MAX applications as the associated service ID. This authentication process is non-interactive so your programs/scripts can run as scheduled tasks, cron jobs, or other automatically triggered background processes.

The associated Service ID functions just like any other MAX user/identity: you can grant permission for it to read and create pages in MAX Community, download and upload files, and so forth. You can also add it to MAX groups.

In general, several steps are needed to enable a specific program/script to use a MAX application non-interactively:

- Configure the HTTP client program/library so that it has access to the certificate and can attach it to a web request when necessary. In some cases this can be configured globally for the program/library such that it can automatically attach the certificate to relevant requests, but in other cases it must be explicitly configured on a per-request basis. In most cases, you can find relevant examples by Googling "client certificate" with the name of your client program/library.
- Use the MAX.gov Service Certificate authentication site (serviceauth.max.gov) to validate the certificate and establish an authenticated HTTP "session" with the target MAX application. In most cases, this simply means sending a GET request to:
  https://serviceauth.max.gov/cas-cert/login?service=<target application base URL>
  eg:
  https://serviceauth.max.gov/cas-cert/login?service=https://community.max.gov/
- Send additional "normal" (no certificate or further authentication needed) HTTP requests to the target MAX application using the authenticated HTTP session (using the HTTP Cookies established by the previous step).
- Since HTTP sessions will occasionally expire, catch/detect any authentication errors that may occur during "normal" HTTP calls. When needed, re-authenticate through serviceauth.max.gov and retry the request. Some MAX applications will return explicit authentication errors (such as HTTP 401 or 403 responses), while others will return normal HTTP 200 responses with an error message in the body; You will need to experiment with the specific MAX application you are using to determine what an authentication error looks like. Note that any retry logic should be careful to avoid infinite loops if a bug or outage causes continuous authentication failures.

Some environment-specific details and examples are in the following sections.

While the steps outlined above are the recommended approach in all cases, there is an alternative approach that may be used in cases where unmodifiable COTS software is being used to make only infrequent HTTP requests. In place of the last three bullets above, you may be able to prepend "https://serviceauth.max.gov/cas-cert/login?service=" to all of the target application URLs that you need to access, such that a single request is used for both authentication and accessing the target URL. Note that this significantly increases the per-request latency and load impact, it decreases reliability, and it may not work in all cases due to the implementation details of specific MAX applications.

Using your MAX certificate with Windows

In Windows, the same underlying certificate framework is used for almost everything, including browsers (except for Firefox), PowerShell scripts, and programs (e.g. in c# or vb). However, see the "Using your MAX certificate with Java" section below if you are using Java.

Here is an example of how to make a call to MAX using PowerShell:

First, open the certificate store and select the certificate using the friendly name:

```
$store = New-Object
System.Security.Cryptography.X509Certificates.X509Store("My","CurrentUser")

$store.Open("readonly")

$cert = $store.certificates | % { if ($_.FriendlyName -eq "S_MYSVC") { $_ }}
```

Next, use the certificate to establish an authenticated HTTP "session" with the target MAX application:

```
invoke-webrequest -Certificate $cert -SessionVariable session
'https://serviceauth.max.gov/cas-
cert/login?service=https://mds.max.gov/authenticated/userinfo.aspx'
```

Then you can make additional HTTP requests to the target application using the authenticated HTTP session. This example runs a simple MDS (Max Document Services) Excel report:

```
$download = invoke-webrequest -WebSession $session
'https://mds.max.gov/mds4/get_spreadsheet.aspx?folder=tests&xls_filename=simple_user_t
est.xlsx'

[System.IO.File]::WriteAllBytes('simple_user_test.xlsx', $download.content)
```

Note: If your system uses proxy settings, you may have to specify them on the invoke-webrequest calls.

Using your MAX certificate with Linux

In Linux, most applications must be individually configured to use a certificate.

Here is an example of how to make a call to MAX using curl:

First, you will use the certificate to establish an authenticated HTTP "session" with the target MAX application:

```
curl -L -c session --cert ~/.max.S_MYSVC.cer --key ~/.max.S_MYSVC.key
'https://serviceauth.max.gov/cas-cert/login?service=https://mds.max.gov/
authenticated/userinfo.aspx'
```

Then you can make additional HTTP requests to the target application using the authenticated HTTP session. This example runs a simple MDS (Max Document Services) Excel report:

```
curl -L -b session
'https://mds.max.gov/mds4/get_spreadsheet.aspx?folder=tests%26xls_filename=simple_user
_test.xlsx' >simple_user_test.xlsx
```

Note: If your system requires proxy settings then you will have to specify them to curl.

Using your MAX certificate with Java

For most (but not all) Java applications, certificate authentication is configured and managed at the JVM level, and is completely transparent to the application server (e.g. Tomcat), framework (e.g. Spring), and application. In other words, certificate authentication is typically configured using command line arguments for the Java interpreter, and the code running on top of the JVM typically has no options for configuring certificates. Most Java servers (e.g. Tomcat) have a startup script that can be modified to add the necessary command line arguments to the Java interpreter.

In Windows:

Java must be explicitly configured to use the Windows certificate store using the following arguments:

```
java -Djavax.net.ssl.trustStoreType=Windows-ROOT -
Djavax.net.ssl.trustStore=NONE -Djavax.net.ssl.keyStoreType=Windows-MY -
Djavax.net.ssl.keyStore=NONE ...
```

In Linux:

Java uses its own "keystore" file format to store the private key and certificate, and you must generate a file in this format from a ".p12" or ".pfx" file (which can be generated as described in the “Getting a MAX Certificate on Linux” section above):

```
keytool -importkeystore -srckeystore ~/.max.S_MYSVC.p12 -srcstoretype PKCS12 -
destkeystore ~/.max.S_MYSVC.keystore -destalias S_MYSVC
rm ~/.max.S_MYSVC.p12
```

Since the ".keystore" file contains the private key, it must be protected as described above for the ".key" file.

Java must then be configured to use the keystore file using the following arguments:

```
java -Djavax.net.ssl.keyStore=~/.max.S_MYSVC.keystore -
Djavax.net.ssl.keyStorePassword=<password> ...
```

You may be aware that Java has a standard keystore-format file named “cacerts”. Your MAX certificate should not be added to that file. That “trust store” file is used to verify the identity of other servers, and Java requires a separate “key store” file to hold certificates that will be used to authenticate to other servers.

In both Windows and Linux, an additional argument should be included in the Java command line to correct a Java bug that could cause problems when connecting to some MAX apps:

```
java -Dsun.net.http.retryPost=false ...
```

Once the JVM has been configured, you can use HttpsURLConnection to authenticate via

```
https://serviceauth.max.gov/cas-cert/login?service=...
```

and then make “normal” calls to the target application, similar to the examples in the “Using your MAX certificate with Windows” and “Using your MAX certificate with Linux” sections above. HttpsURLConnection automatically saves cookies by default, so unlike the Windows and Linux examples, no explicit cookie configuration/handling is needed.

If you are using a Java framework or library that does not use HttpsURLConnection internally, note that a few frameworks/libraries ignore the -Djavax.net.ssl.keyStore, -Djavax.net.ssl.keyStoreType, and -Djavax.net.ssl.keyStorePassword command line arguments and require you to configure the keystore a different way. For example, the Apache HttpComponents library and Unirest library both require programmatically configuring the keystore as described here and here.

Summary and further information

Now that you've read through this document, you know what a MAX certificate is, how to get one, and what to do with it. Hopefully you also come away with the sense that certificates are useful for a certain class of problem that you couldn't solve another way, but at the same time, a sense that using them requires care and diligence.

If you need help obtaining or using a MAX certificate, please get in contact with MAX Support at support@max.gov, by calling 202-395-6860 or by visiting https://support.max.gov.