

# Abusing Replication: Stealing AD FS Secrets Over the Network

---

[fireeye.com/blog/threat-research/2021/04/abusing-replication-stealing-adfs-secrets-over-the-network.html](https://fireeye.com/blog/threat-research/2021/04/abusing-replication-stealing-adfs-secrets-over-the-network.html)



Threat Research

Douglas Bienstock

Apr 27, 2021

10 mins read

Threat Research

Organizations are increasingly adopting cloud-based services such as Microsoft 365 to host applications and data. Sophisticated threat actors are catching on and Mandiant has observed an increased focus on long-term persistent access to Microsoft 365 as one of their primary objectives. The focus on developing novel and hard to detect methods to achieve

this goal was highlighted with the recent detection of [UNC2452](#) and their access to Microsoft 365. One of this group's key TTPs was to steal the Token Signing Certificate from an organization's AD FS server to enable them to bypass MFA and access cloud services as any user, at any time. While defenders previously associated the defense of this certificate, and thus the entire ecosystem, with careful access control and detection efforts around the AD FS server and service account, this is no longer sufficient. In this blog post we will show how a threat actor, with the right privilege, can extract the encrypted Token Signing Certificate from anywhere on the internal network. Once extracted, a threat actor can easily decrypt it and begin accessing cloud services.

## **Active Directory Federation Services**

---

Active Directory Federation Services (AD FS) is a feature for Windows Servers that enables federated identity and access management. It is often used by organizations to provide single sign-on functionality to access enterprise applications such as Microsoft 365. In technical terms, AD FS functions as an **Identity Provider** (IdP) and Microsoft 365 is a **Service Provider** (SP). We'll use Microsoft 365 as an example going forward, but this technique could apply to any service that is set up to trust AD FS. AD FS verifies a user's identity and issues assertions that describe the user. Microsoft 365 trusts AD FS to verify user identities and provide it with assertions. To Microsoft 365, it doesn't matter how AD FS performed the verification, it just needs the assertions.

In the typical deployment (Figure 1), AD FS will verify a user's identity using Active Directory. At a minimum, an AD FS deployment consists of two servers in an enterprise's on-premises network: the primary AD FS server, and an AD FS Web Application Proxy (WAP). The proxy is placed in the DMZ and has no functionality besides proxying sign-on attempts from the Internet to the AD FS server. The primary AD FS server receives proxied requests, verifies a user's identity, and issues assertions that are packaged into SAML security tokens for the user.

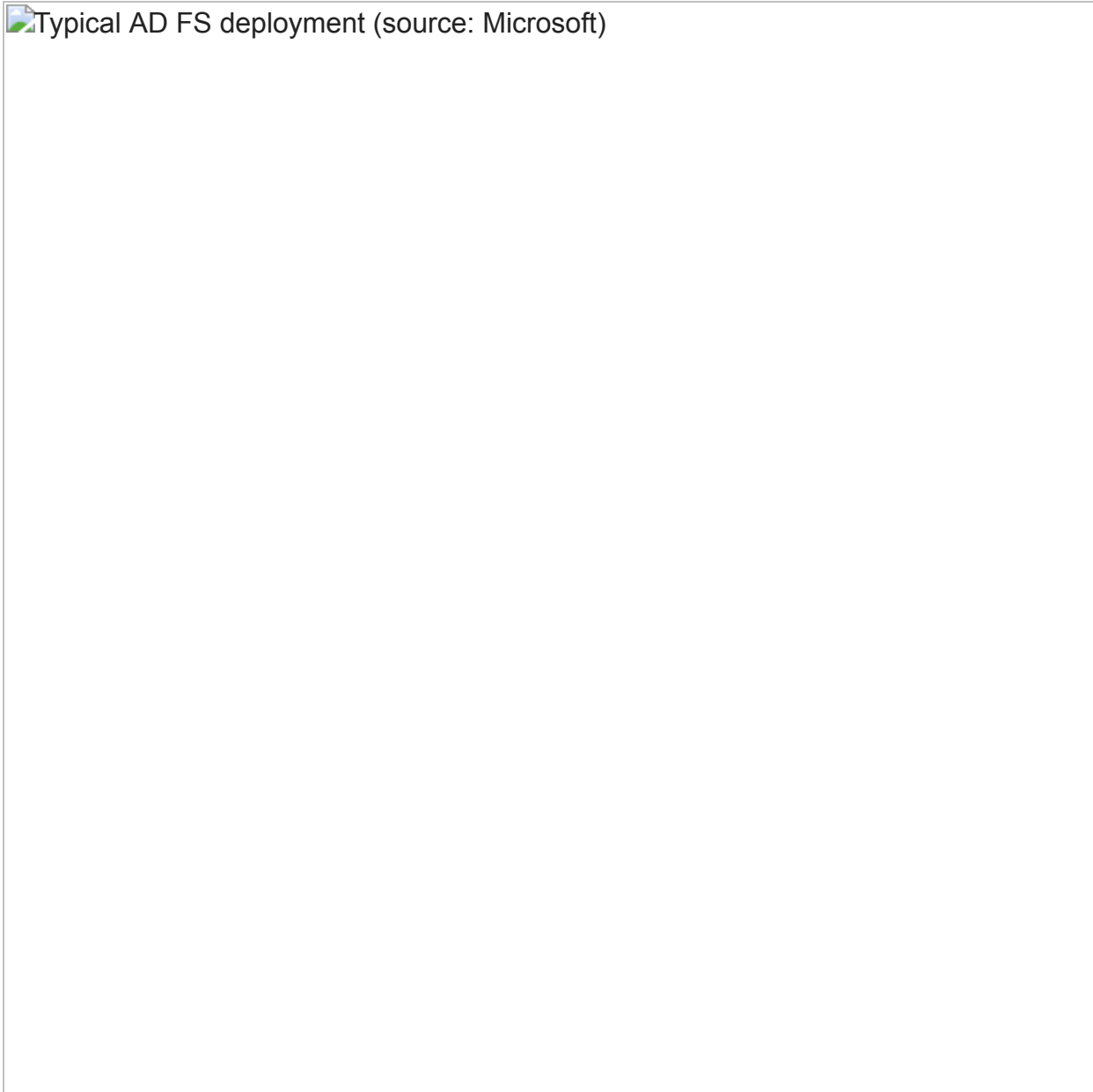
Typical AD FS deployment (source: Microsoft)

Figure 1: Typical AD FS deployment (source: [Microsoft](#))

The SAML token issued by AD FS proves a user's identity to Microsoft 365 and can also be used to make authorization decisions. The SAML token is an XML document with two main components:

1. **Assertions:** Assertions are XML elements that describe the user's identity. An assertion could be a user SID, group membership SIDs, or other elements like the user's department name. A single SAML token can have multiple assertions attached to it.
2. **Digital Signature:** The assertions in the SAML token are digitally signed using a public/private keypair that resides on the AD FS server. This is called the Token Signing Certificate.

**The Token Signing Certificate is the bedrock of security in AD FS.** Microsoft 365 uses the digital signature to validate that the SAML token is authentic, valid, and comes from an AD FS server that it trusts. To enable this verification, an administrator shares the public component of the Token Signing Certificate with Microsoft 365. This is then used to cryptographically verify the digital signature in the SAML token and prove authenticity as well as integrity of the token. In other words, if a threat actor got hold of a Token Signing Certificate, they could generate arbitrary SAML tokens to access any federated application, as any user, and even bypass MFA.

## Golden SAML

---

Golden SAML was coined in 2017 by CyberArk to describe the technique of forging SAML tokens to access SPs given a valid Token Signing Certificate. At TROOPERS 19, I detailed how a threat actor could extract the Token Signing Certificate from an AD FS server, as well as some mitigation strategies for defenders.

In a default AD FS configuration, the Token Signing Certificate is stored within a Windows Internal Database (WID) instance that is running on the AD FS server. WID is more or less MS SQL Express, except the database can only be accessed locally over a special named pipe connection. In AD FS, the database is further locked down to only the AD FS service account. The Token Signing Certificate is stored in an encrypted state in the IdentityServerPolicy.ServiceStateSummary table. Figure 2 contains a single row with a column that stores all the settings that AD FS will need on service start as an XML document.

```
<SigningToken>
  <IsChainIncluded>>false</IsChainIncluded>
  <IsChainIncludedSpecified>>false</IsChainIncludedSpecified>
  <FindValue>99FABAEE46A09CD9B34B9510AB10E2B0C0ACB99B</FindValue>
  <RawCertificate></RawCertificate>
  <EncryptedPfx></EncryptedPfx>
  <StoreNameValue>My</StoreNameValue>
  <StoreLocationValue>CurrentUser</StoreLocationValue>
  <X509FindTypeValue>FindByThumbprint</X509FindTypeValue>
</SigningToken>
```

Figure 2: Example Token Signing Certificate stored in the AD FS database

The Token Signing Certificate as it is stored in the AD FS database is encrypted using symmetric key encryption. Windows uses a technology called Distributed Key Management (DKM) to store the secret value used to derive the symmetric key in an Active Directory container. The AD FS service account can read the attributes of this container, derive the symmetric key, and then decrypt the Token Signing Certificate.

## AD FS Replication

---

AD FS also supports a farm configuration for high availability and load balancing in larger enterprise networks. The individual AD FS servers in a farm can be configured to use unique Token Signing Certificates; however, the default is to have the servers share the same Token Signing Certificate. In order to stay in sync with each other, the farm will have a primary node and secondary nodes. The secondary nodes make use of a replication service to acquire configuration settings and certificates from the primary AD FS server. To facilitate this, AD FS makes use of Windows Communication Foundation (WCF).

WCF is a framework that allows developers to build service-oriented applications. A WCF application has two components: the service that will receive and process messages, and the client that sends messages to a service and receives back responses. The AD FS servers run a WCF service that is called the Policy Store Transfer Service internally.

To send a message to this service, the client will connect to the URL `http://<adfs server name>:80/adfs/services/policystoretransfer`. Note that even though the channel is over HTTP, the actual data being exchanged is encrypted during transit. It is also key to understand that although there is a single primary AD FS server, all nodes in an AD FS farm run this WCF service and can be used for replication.

Upon receipt of a message, the WCF service enforces an authorization check to ensure the calling identity is permitted to receive the requested information. The permission check is done by evaluating an authorization policy that is also stored in the `IdentityServerPolicy.ServiceStateSummary` table of the AD FS database. The policy permits identities whose primary SID matches the AD FS Service account *or* to any identity that is member of the AD FS server's local administrators group. If the identity of the client passes the authorization check, then the WCF service will send back a message containing the requested information.

```
<AuthorizationPolicy>
@RuleName = "Permit Service Account"exists([Type ==
"http://schemas.microsoft.com/ws/2008/06/identity/claims/
primarysid", Value == "S-1-5-21-3508695881-2242692613
-376241919-1107"]) => issue(Type = "http://schemas
.microsoft.com/authorization/claims/permit", Value = "
true");
@RuleName = "Permit Local Administrators"exists([Type ==
"http://schemas.microsoft.com/ws/2008/06/identity/claims/group
sid", Value == "S-1-5-32-544"])=> issue(Type = &quot
;http://schemas.microsoft.com/authorization/claims/permit", Value
= "true");
</AuthorizationPolicy>
```

Figure 3: Default Authorization Policy for AD FS server

## Room for Abuse

---

A threat actor can abuse the Policy Store Transfer Service to acquire the encrypted Token Signing Certificate over the network, similar to the DCSync technique for Active Directory. It is important to note that the data is still encrypted and requires the DKM key stored in Active Directory to decrypt. This technique, however, requires a significant change to how defenders have secured AD FS servers and monitored them for theft of the Token Signing Certificate.

First, previous techniques required code execution on an AD FS server to extract the data or at least an SMB connection to transfer the backing database files. With a strong defense in depth program using secure credential management, EDR, and network segmentation, an enterprise can make it very difficult for a threat actor to access an AD FS server and the Token Signing Certificate. Abusing the AD FS Replication service, however, requires only access to the AD FS server over the standard HTTP port. The default installation of AD FS will even create a Windows Firewall rule to allow HTTP traffic from any system. Additionally, a threat actor does not need the credentials for the AD FS service account and can instead use any account that is a local administrator on an AD FS server. Lastly, there is no Event Log message that is recorded when a replication event occurs on an AD FS server. Altogether, this makes the technique both much easier to execute and much harder to detect.

The authorization policy itself also presents an opportunity for abuse. Because the authorization policy is stored as XML text in the configuration database, a threat actor with enough access could modify it to be more permissive. A threat actor could modify the Authorization Policy to include a group SID such as domain users, S-1-5-21-X-513. Similarly, they could add an ACE to the DKM key container in Active Directory. This would allow the threat actor to easily obtain the Token Signing Certificate and decrypt it using any domain user credentials. This would give them persistent ability to perform a Golden SAML attack with only access to the network as a requirement.

Mandiant has not yet observed this technique used in the wild; however, it is trivial to write a POC for and we are aware of one public tool that will soon support it. Figure 4 shows the output of POC code written in .NET to extract the Token Signing Certificate from a remote AD FS server.

A screenshot of a terminal window titled "POC code output". The window is currently empty, showing no text or code.

Figure 4: POC code output

## Mitigations

---

The best mitigation against this technique is to use the Windows Firewall to restrict access to port 80 TCP to only the AD FS servers in the farm. If an organization has only a single AD FS server, then port 80 TCP can be blocked completely. This block can be put in place because all traffic to and from AD FS servers and proxies for user authentication is over port 443 TCP.

To limit inbound communications, modify the existing firewall rule that AD FS inserts on installation.

```
Set-NetFirewallRule -DisplayName "AD FS HTTP Services (TCP-In)" -RemoteAddress  
<ADFS1 IP address>,<ADFS2 IP Address>
```

If no rule exists, the scriptlet in Figure 5 should be applied to all ADFS servers to create one.

```
New-NetFirewallRule -DisplayName "Allow ADFS Servers TCP 80" -Direction Inbound -  
Action Allow -Protocol TCP -LocalPort 80 -RemoteAddress <ADFS1 IPAddress >,  
<ADFS2 IPAddress>
```

Figure 5: Windows Firewall - Allow ADFS Server - TCP 80

Organizations that are monitoring the internal network can alert on HTTP POST requests to the address that hosts the Policy Store Transfer service. If there is an AD FS farm, then the IP addresses of the AD FS servers will need to be whitelisted against the rule. Figure 6 shows a sample Snort rule to detect this activity.

```
alert tcp any any -> any 80 (msg:"AD FS Replication"; flow:established, to_server;  
content:"POST"; http_method; content:"adfs/services/policystoretransfer"; http_uri;  
threshold:type limit,track by_src,count 1,seconds 3600; priority:3; sid:7000000; rev:1;)
```

Figure 6: Sample snort rule

## Acknowledgements

---

Mandiant would like to acknowledge the great work of Dr. Nestori Syynimaa (@DrAzureAD). Dr. Syynimaa independently thought to research the replication of configuration information between AD FS servers and has published his findings on his [blog](#). Mandiant would also like to thank Microsoft for their collaboration on mitigations and detections for this technique. Lastly, special thanks to Mike Burns of the Mandiant Security Transformation services team for his feedback on mitigations and detections.