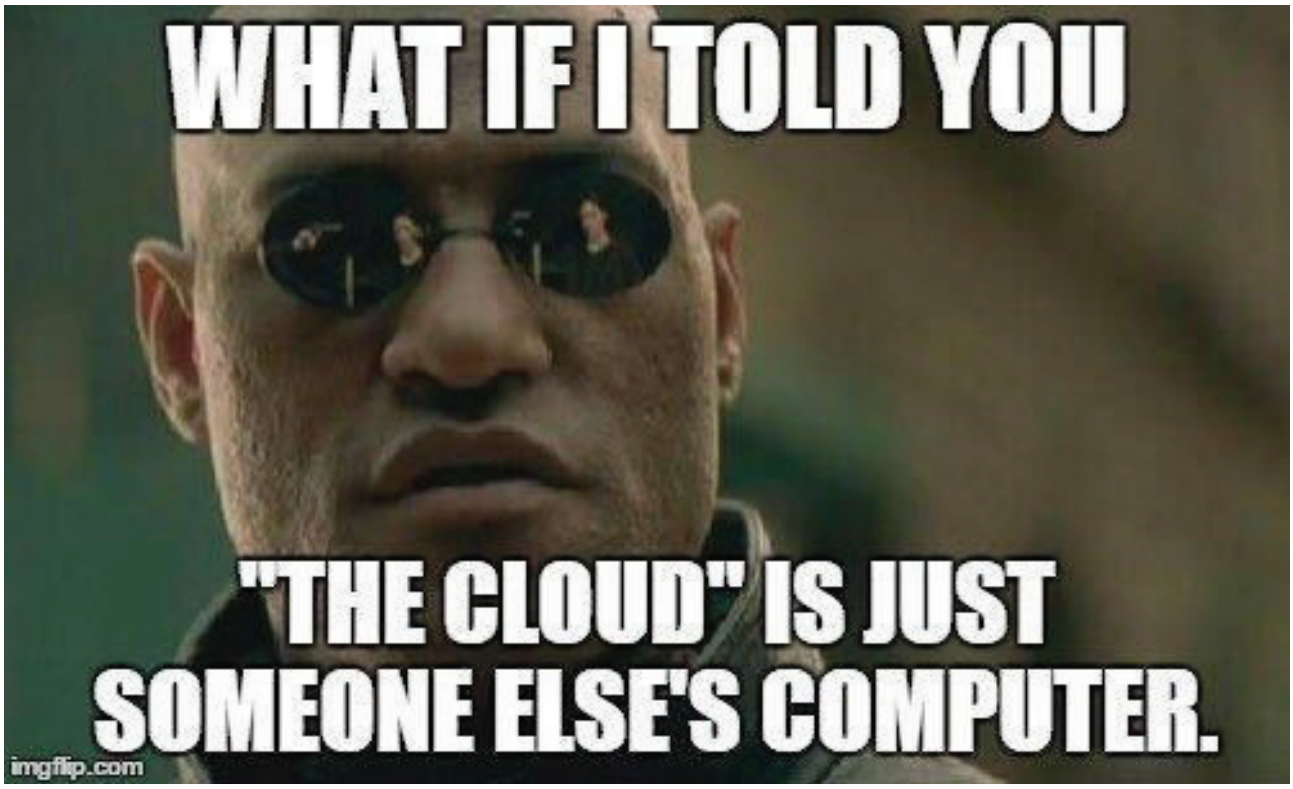
SILVERFORT

Bounce the Ticket and Silver Iodide Attacks on Azure AD Kerberos

Dor Segal, Senior Researcher at Silverfort

Abstract

Microsoft has recently announced general availability of Azure AD Kerberos, a cloud-based implementation of Kerberos. When reviewing it, we found that despite some attempts to make it more secure than the on-prem implementation of Kerberos, Azure AD Kerberos can be attacked with similar techniques. In this white-paper we introduce these two techniques - Bounce the Ticket, and Silver Iodide – the cloud-based versions of Pass-the-Ticket and Silver-Ticket.



Source

Intro

Azure Active Directory (Azure AD) Kerberos is an implementation of the Kerberos authentication protocol that was purpose-built to work in the Azure cloud platform.

Back, in the pre-cloud days, Kerberos (together with its preceding NTLM) was the core for most to all authentications in an enterprise environment. However, the continuous and increasing transition to a hybrid environment mode, in which SaaS applications, cloud workloads and traditional on-prem infrastructure are used conjointly has changed that. Nowadays, the common practice for SaaS applications is to authenticate to a cloud-based identity provider, such as Azure AD. While on-prem workloads still authenticate to an AD with legacy protocols like Kerberos.

As a result, the authentication of cloud infrastructure like Azure Virtual Desktop and Azure Files is dependent on technology built for the on-prem environment even when they themselves do not belong with an on-prem AD domain.

The new Azure AD Kerberos authentication protocol was designed to overcome this anomaly and transit these cloud workloads to the Azure AD cloud native authentication infrastructure, introducing a smooth integration between cloud Azure resources and current on-prem windows protocols and applications such as SMB, RDP, RPC and their likes. In that manner, organizations can consolidate the authentication of all their cloud resources, whether SaaS apps or infrastructure workload in Azure AD.

However, Azure AD Kerberos still has a lot in common with its on-prem peer. In this white paper, we share research we've conducted to check whether existing attack techniques for the traditional Kerberos can apply – with some modifications to the new Azure AD Kerberos.
The results of this research are two new attack techniques. The first one is an adjustment of the infamous Pass-the-Ticket attack to Azure AD Kerberos, which we've named Bounce-the-Ticket. The second, does the same for Silver Ticket and thus we named it Silver Iodide. This white paper describes both attacks in detail, **as well as suggestion how to mitigate them.**

In accordance with responsible disclosure, both techniques were shared with Microsoft's MSRC team prior to this publication. We'd like to share our appreciation with the time and effort it dedicated to go over the research and approve its publication.

Kerberos Reminder

Kerberos is a modern recommended authentication protocol for enterprise that consists of three independent guarding mechanisms (just like the three headed dog Cerberus) that work together to provide a secure channel to transfer authentication messages. The protocol is based on ticket exchanges, each providing a verification for a different part of the authentication process. The Key Distribution Center (KDC) manages the authentication process and oversees handing out tickets to authenticating clients, as well as oversees the validation of the user's password and the handing out service tickets. The KDC is based on a symmetric key cryptography and can optionally involve public key cryptography.
These are the three mechanism Kerberos implements:

#1 Obtaining a Ticket Granting Ticket

This is the first phase of the authentication process, in which the client must provide a valid password to receive a ticket that validates its integrity. The TGT (Ticket granting ticket) is encrypted by the Authentication Service, and is the only one that can verify the integrity of a given TGT ticket. In addition, alongside the TGT, a session key is attached to provide an encryption platform for the Ticket Granting Service. The session key is encrypted with the user's password. Then the user can decrypt the message and extract the session key. Then, the provided key will be used to encrypt the service ticket request (TGS request) for validation purposes.

#2 Obtaining a Service Ticket

In the second phase of the authentication, the user requests a Ticket Granting Service from the KDC. This type of ticket is intended for a specific service. In Active Directory, a registered service is called Service Principal Name (SPN) and its convention is service class/host. To validate service tickets, a shared secret is required – the server's private key. In Windows machines, this secret is saved in the Registry in the Security hive. In other platforms and 3rd party applications, Microsoft offers to export a pair of an SPN and the server's private key called Keytab. The TGS ticket is encrypted with the server's private key and includes a Privilege Attribute Certificate (PAC) that holds the authorization data about the client. The information cannot be edited or modified because only the server knows its own private key.

#3 Application request

The last phase in the authentication is the Application Request (AP REQ). This request is sent from the client directly to the server over the intended protocol (e.g., HTTP, SMB, RPC, RDP). The service ticket is attached and will be decrypted on the server side with the server's private key that is shared with the KDC. In that manner, the server validates the client's ticket is from the right KDC (integrity). Following a successful decryption, the server will go through the PAC and decide if the client is authorized to access the server and reply with an answer

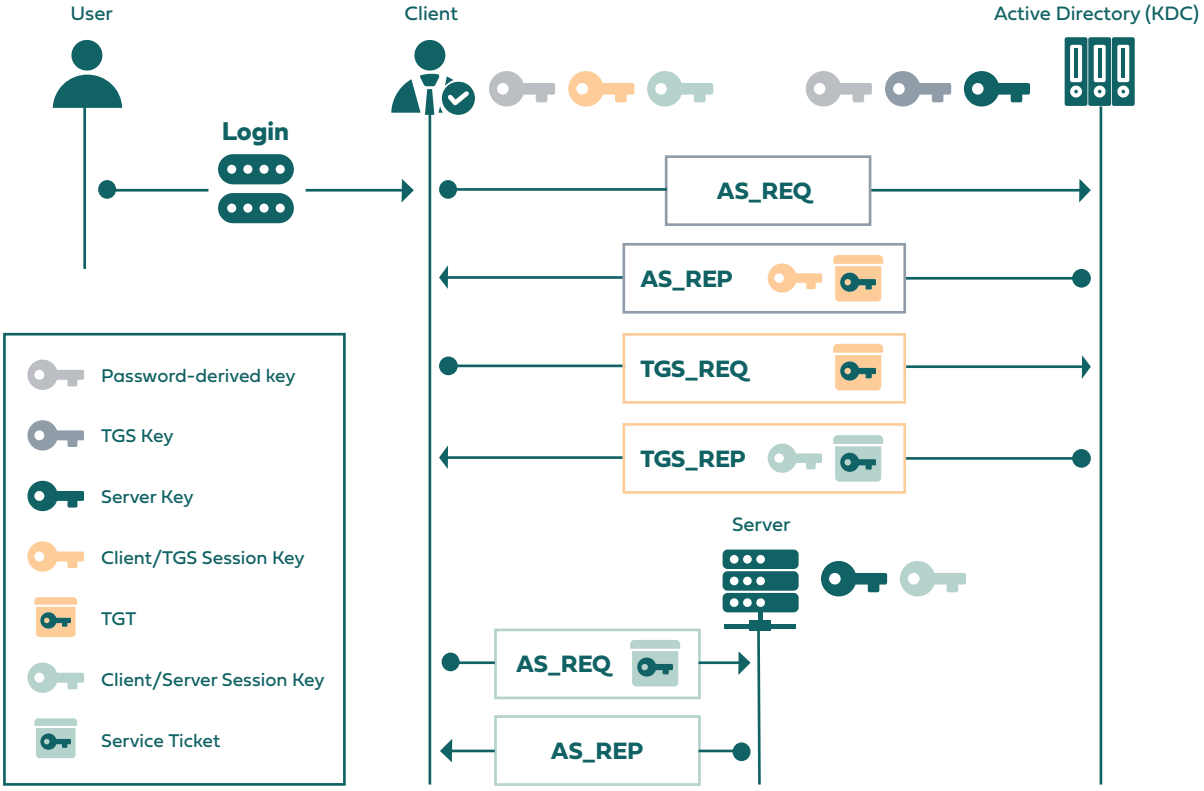


Diagram 1: Kerberos authentication flow

Introducing Azure AD Kerberos

Kerberos works with two services, the Authentication Service (AS), that's accountable for validation of the user's password, and the TGS (Ticket Granting Service), that grants tickets for authentication to different domain resources and provides details to the server about the user. Azure AD Kerberos works the same, with a few major modifications:

Source

Encryption of Client-KDC Communication

In Azure AD Kerberos, the KDC (consisting of the AS and the TGS) is in the cloud, and therefore clients cannot communicate with the KDC over non-encrypted channels. Hence, Azure AD utilizes the KDC proxy Kerberos extension (MS-KKDCP), that transfers the Kerberos messages over a TLS tunnel.

KDC Proxy Cloud Domain Mapping

During registration to Azure AD, Windows maps the domain windows.net to the KERBEROS.MICROSOFTONLINE.COM realm. Based on this mapping, when the user attempts to access a resource that requires Kerberos authentication in the domain windows.net, the KDC proxy protocol is used to obtain a service ticket from the Azure-based KDC with the realm KERBEROS.MICROSOFTONLINE.COM.

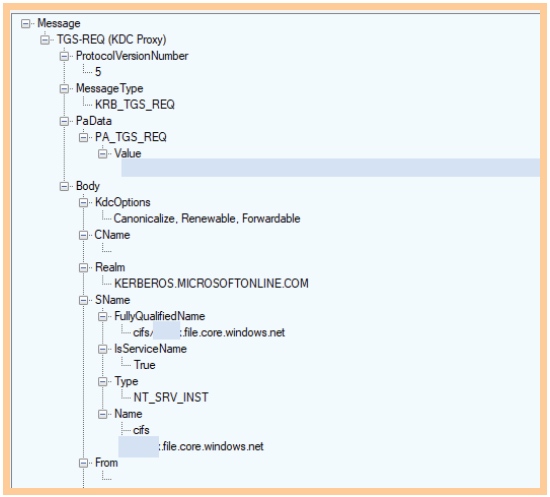
```
+----------------------------------------------------------------------+
| SSO State                                                            |
+----------------------------------------------------------------------+

                AzureAdPrt : YES
     AzureAdPrtUpdateTime : 2022-11-16 12:02:00.000 UTC
     AzureAdPrtExpiryTime : 2022-11-30 12:02:07.000 UTC
      AzureAdPrtAuthority : https://login.microsoftonline.com/
             EnterprisePrt : NO
    EnterprisePrtAuthority :
               OnPremTgt : NO
                CloudTgt : YES
       KerbTopLevelNames : .windows.net,.windows.net:1433,.windows.net:3342,.azure.net,.azure.net:1433,.azure.net:3342
```

Screenshot 1: KerbTopLevelNames field to map between cloud resources to cloud KDC

Decoupling of Authentication and TGS Services

Microsoft has separated the endpoints of the Authentication Service and the Ticket Granting Service from each other, so now the cloud TGT is retrieved alongside the Primary Refresh Token (PRT), a token obtained during the sign on to Azure AD. The TGT is provided by Azure AD, when the request to login.microsoftonline.com/<tenantid>/oauth2/token is sent with a POST attribute of tgt=true. The response contains an encrypted blob that includes the PRT alongside the krbtgt. The TGS will be requested from the endpoint login.microsoftonline.com/<tenantid>/Kerberos with a KDC Proxy defined TGS request alongside the TGT just like a normal Kerberos TGS request.



Screenshot 2: Kerberos TGS request using a KDC Proxy

Lastly, compatible resources like Azure Files accept Kerberos authentication request AP-REQ (Application request) with the acquired service ticket.

It seems like a lot of thought and efforts were put into Azure AD Kerberos security. The first security enhancement is the separation of the AS and the TGS endpoints enable a stronger security mechanism to protect the krbtgt secret key and avoid the Golden Ticket attack. The second one is the use of KDC Proxy and the transition to TLS provide additional security layer to the authentication messages over the web. However, is Azure AD Kerberos better protected against the rest of common attack techniques that its on-prem peer is vulnerable to? Let's explore this deeper.

Attacking Azure AD Kerberos Protocol

We'll introduce two new attack techniques we've developed, that leverage weaknesses within these modifications, which we've named **Bounce the Ticket** and **Silver Iodide**

Bounce-the-Ticket Attack

Pass the ticket attack reminder

One of the main security issues with Kerberos is its in-memory ticket caching in the lsass process. This mechanism wasn't modified in Azure AD Kerberos at all. Once an adversary has access to a Windows machine, it can extract the TGT from memory and use it to request any Kerberos service ticket they wish. This attack is called Pass-the-Ticket (PTT).

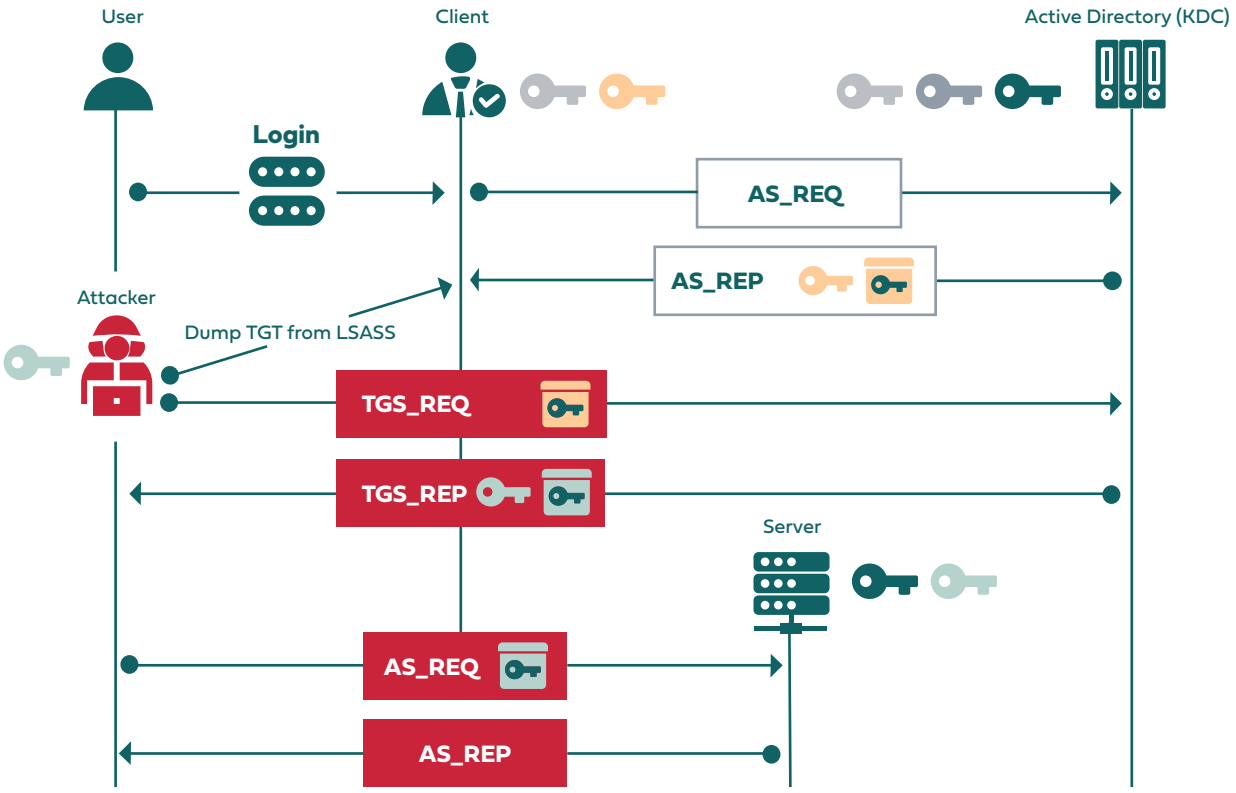


Diagram 2: Pass-the-Ticket attack

Bounce-the-Ticket

We've named this variant of Pass-the-Ticket for Azure AD Kerberos Bounce the Ticket, and they are actually quite similar. Here too, we extract the TGT from memory, but instead of accessing an on-prem TGS we use it to access the cloud-based one. The obtained ticket can be used to access cloud-based resources that rely on Azure AD Kerberos.

In hybrid environments, where both an on-prem AD is used and an azure AD Kerberos domain is used, an attacker can utilize the bounce the ticket attack to pivot from the on-prem AD environment to the Azure AD managed cloud one.

Implementation in Rubeus

While the attack is similar to the on-prem pass-the-ticket, there are still some differences that require changes to existing attack tools to facilitate the attack. We chose to use Rubeus, a tool by GhostPack, which is used for Kerberos interaction and abuse. The following alterations were made to adapt Rubeus to Azure AD Kerberos:

- The realm was changed to KERBEROS.MICROSOFTONLINE.COM to match the cloud realm
- The encrypted PAC username field was changed to include a full UPN (followed by the original domain name).
- A different and constant LogonServer: login.microsoftonline.com.

```
    }
    kvi.LogonServer = new Ndr._RPC_UNICODE_STRING("login.microsoftonline.com");
    if (!String.IsNullOrEmpty(displayName))
    {
```

Screenshot 3: LogonServer modification

Fortunately, Rubeus already supports KDC Proxy extension as it is simply not a new extension and used to give applications the ability to query Kerberos over the web.

To run the attack with Rubeus, we:

1. Use Mimikatz, or another attack tool to extract the cloud TGT from memory and save it as a file.

a. Privilege::debug -> sekurlsa::tickets /export

b.

```
mimikatz 2.2.0 x64 (oe.eo)
'## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
 '#####'        > https://pingcastle.com / https://mysmartlogon.com ***/

mimikatz # privilege::debug
Privilege '20' OK

mimikatz # sekurlsa::tickets /export

Authentication Id : 0 ; 4118744 (00000000:003ed8d8)
Session           : RemoteInteractive from 3
User Name         : sfuser
Domain            : dsesx
Logon Server      : (null)
Logon Time        : 14/11/2022 6:11:59
SID               : S-1-12-1-3359798570-1206766658-3358109879-3612924077

         * Username : sfadmin
         * Domain   : KERBEROS.MICROSOFTONLINE.COM
         * Password : (null)

        Group 0 - Ticket Granting Service
         [00000000]
           Start/End/MaxRenew: 14/11/2022 6:18:29 ; 14/11/2022 7:18:29 ;
           Service Name (02) : cifs ; dsesx.file.core.windows.net ; @ KERBEROS.MICROSOFTONLINE.COM
           Target Name  (02) : cifs ; dsesx.file.core.windows.net ; @ KERBEROS.MICROSOFTONLINE.COM
           Client Name  (01) : sfuser@ad.dsesx.silverfort.io ; @ KERBEROS.MICROSOFTONLINE.COM
           Flags 40000000    : forwardable ;
           Session Key       : 0x00000001 - des_cbc_crc
             fa2bc43dfc3c266e7c2e5a1a5ce9bd60274e96b6cd0b8c2c7cef64b696f967c7
           Ticket            : 0x0000001
```

2. Use the extracted ticket to request a service ticket from the cloud KDC, using, and put it back into lsass, using the following command:
   Rubeus.exe asktgs /ticket:ticket.kirbi /service:cifs/AzFilesShare.file.core.windows.net /proxyurl:https://login.microsoftonline.com/tenantID/kerberos /enctype:AES256 /ptt
3. Access the target service directly.
4. Success!

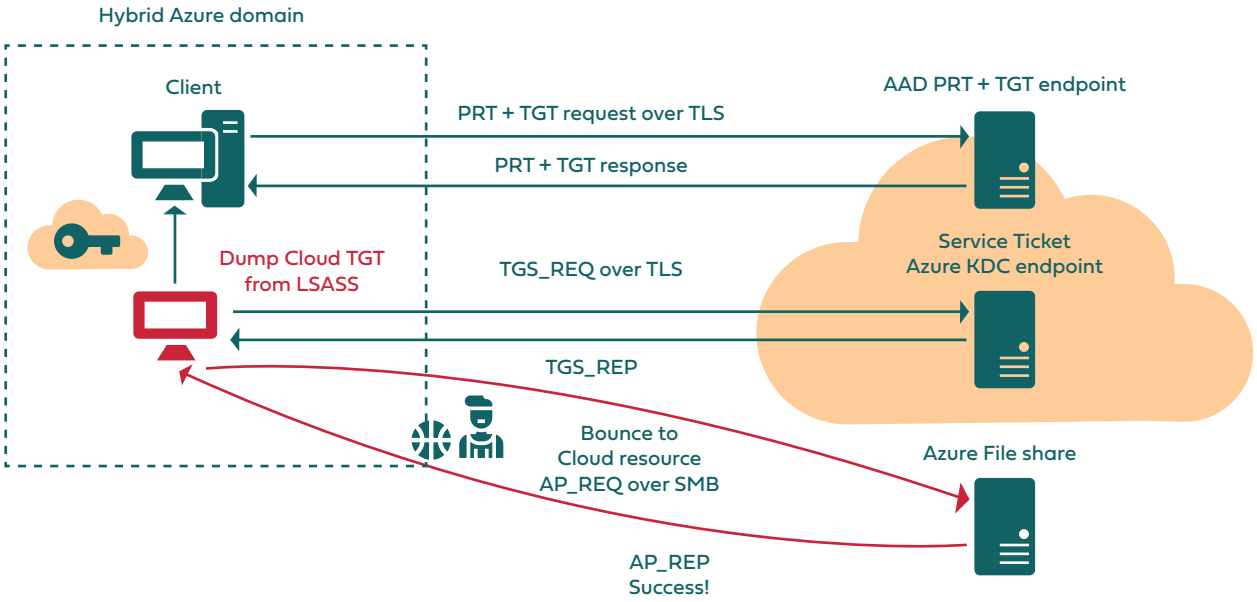


Diagram 3: Bounce-the-Ticket attack

Silver Iodide

We next introduce an attack technique, based on the on-prem silver ticket, in that enables an an attacker to utilize a stolen server key to forge tickets that can be used for accessing that server with any required privileges.

Silver ticket reminder

In the classic on-prem Silver Ticket attack, an attacker first obtains a Server Kerberos key. This can be done using a variety of techniques, for example Kerberoasting. Once the attacker obtains a server key, this key can be used to forge service tickets. This is made possible because the service tickets in Kerberos are encrypted with the server key, so an attacker that has hold of the server key can forge tickets to that server. For example, an attacker can generate a ticket that specifies a username the attacker doesn't have credentials for, or elevate the initial privileges to the attacker's compromised users to higher ones, enabling the attacker to access the target service as an admin.

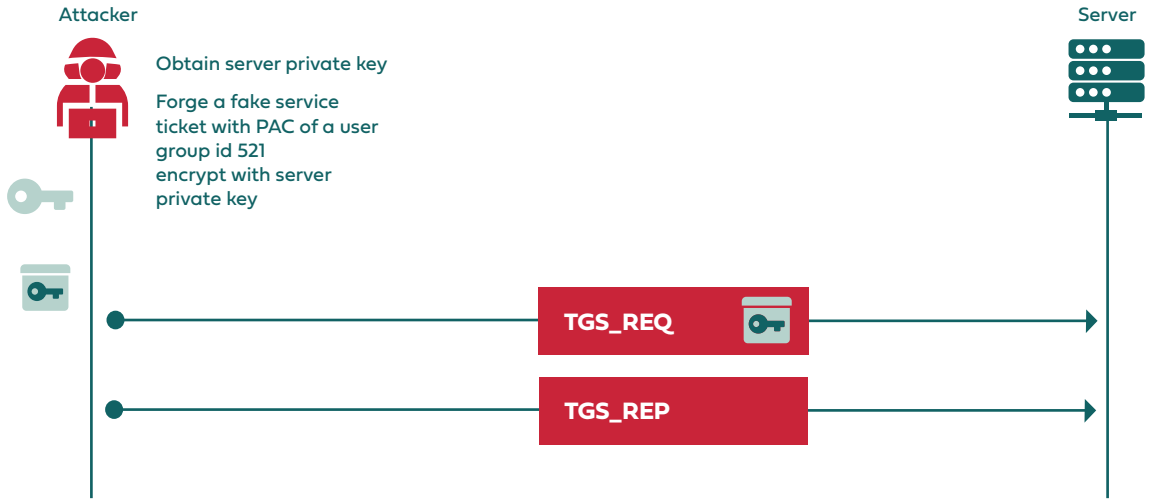


Diagram 4: Silver Ticket attack

Silver Iodide technique

The Silver Iodide attack is similar to the on-prem Silver Ticket attack with a couple of differences. The primary challenge in adapting this technique to the cloud, is that in Azure AD Kerberos the server keys are generated by Azure using cryptographically strong random number generators. This means that we cannot rely on attacks that assume weak encryption keys, such as Kerberoasting attacks. We must obtain the server key in a different way. The specific way to obtain the server key varies in respect to the application that's being attacked. For this demonstration we analyze a security gap in Azure Files.

To adapt the on-prem exploitation of Silver Ticket to the cloud, we modified the Rubeus Silver Ticket forging method and changed all the required fields to match the KERBEROS.MICROSOFTONLINE.COM realm and the logon server and use the acquired key to forge our own Silver Iodide ticket.

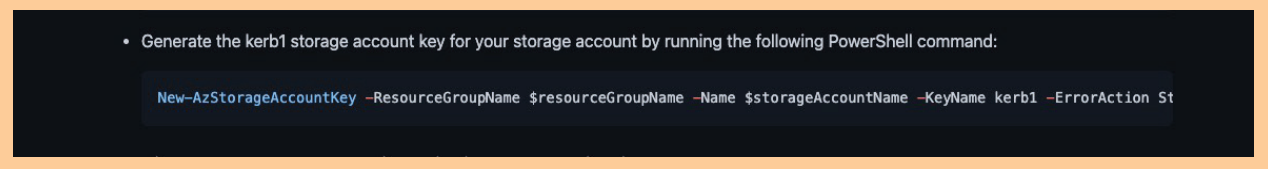
Attacking Azure Files with Silver Iodide

Introduction to Azure Files and How They Use Azure AD Kerberos

Azure Files is a serverless cloud based file share. It is the cloud equivalent of an SMB fileshare. It can be used in different ways. In our context, we are mostly interested in access to Azure Files from Azure Virtual Desktop. That's because this access can be done using the SMB with the Kerberos authentication protocol, via Azure AD Kerberos.

Silver Iodide Privilege Escalation Attack on Azure Files

The Azure file share is configured inside the Azure portal. In the preview version of AAD Kerberos the user was required to configure a storage account key that acts as the storage's private Kerberos key using a powershell command.



Screenshot 4: From Azure AD Kerberos preview docs

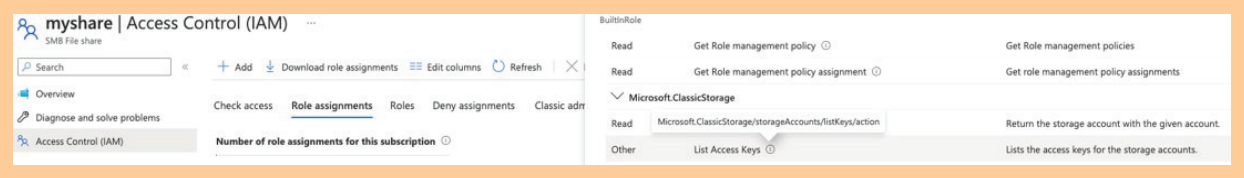
In the GA version, the command is no longer needed, and configuration is done automatically when configuring Azure Files.

This key can be later extracted with the following powershell command:
$kerbKey1 = Get-AzStorageAccountKey -ResourceGroupName $resourceGroupName -Name $storageAccountName -ListKerbKey | Where-Object { $_.KeyName -like "kerb1" }
The above works both in the preview and the GA version.

Apparently, the attacker doesn't necessarily need to have permission to the resource to run the Get-AzStorageAccountKey command. You only need to be a member of one of the following groups:

- Owner
- Contributor
- Avere Contributor (Microsoft.Storage/storageAccounts/)
- DevTest Labs User
- Disk Snapshot Contributor
- Log Analytics Contributor
- Logic App Contributor
- Reader and Data Access
- Storage Account Contributor (Microsoft.Storage/storageAccounts/)
- Storage Account Key Operator Service
- Virtual Machine Contributor

It seems like some of the above groups are not necessarily related to having permissions to access the data in the resource itself, like DevTest Labs User.



So to achieve privilege escalation, all we need to do is:

1. Obtain a weak user in one of the above groups.
2. Then use that user to extract the Kerberos Server key from Azure Files with PowerShell.
3. Having obtained the Server ticket, use Rubeus to forge a service ticket to Azure Files
4. Extract or manipulate the target information in Azure Files

Lastly, we will modify Rubeus Silver Ticket forging method and change all the required fields to match the KERBEROS.MICROSOFTONLINE.COM realm and the logon server and use the acquired key to forge our own Silver Iodide ticket

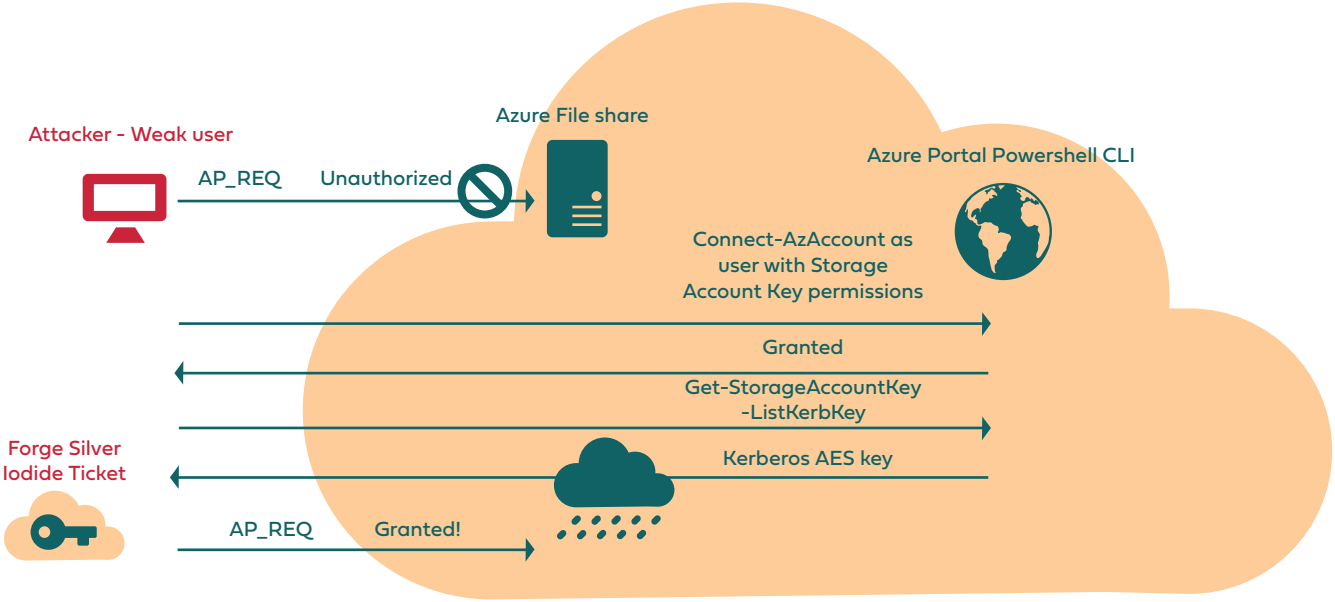


Diagram 5: Silver Iodide attack

Suggested Mitigations

Since there is no fix for these attacks, we recommend the following mitigations:

1. Review and monitor for any changes to Azure Access Control (IAM) and the share's access control permissions to validate that only authorized users have permissions for the Microsoft.ClassicStorage/storageAccounts/listKeys/action - Kerberos key extraction operation.
2. To avoid the bounce the ticket attack, reduce the number of computers allowed to hold cloud TGTs to the minimum required. You can do that by restricting the "Allow retrieving the Azure AD Kerberos Ticket Granting Ticket during logon" group policy to security groups that use Azure AD Kerberos.