699.5
Azure AD & Emulation Plans

SEC699.5 | Advanced Purple Team Tactics

SANS

Azure AD & Emulation Plans

Welcome to Day 5 of SANS Security SEC699: Advanced Purple Team Tactics – Adversary Emulation for Breach Prevention & Detection.

For any remarks, please reach out to the authors:

Erik Van Buggenhout

evanbuggenhout@nviso.eu

www.nviso.eu

Update: G01_01

Course Roadmap

- Introduction & Key Tools
- Initial Access
- Lateral Movement
- Persistence
- **Azure AD & Emulation Plans**
- Adversary Emulation Capstone

SEC699.5

Azure AD
Azure AD Structure and Management
Azure AD Hybrid Authentication
Azure AD Authentication Methods
Azure AD Conditional Access
Introduction to Azure Identities
Azure AD Security Logging

Executing emulation plans
APT-28 Emulation Plan
Exercise: APT-28 Emulation Plan
APT-34 Emulation Plan
Exercise: APT-34 Emulation Plan
Turla Emulation Plan
Exercise: Turla Emulation plan

This page intentionally left blank.

Course Roadmap

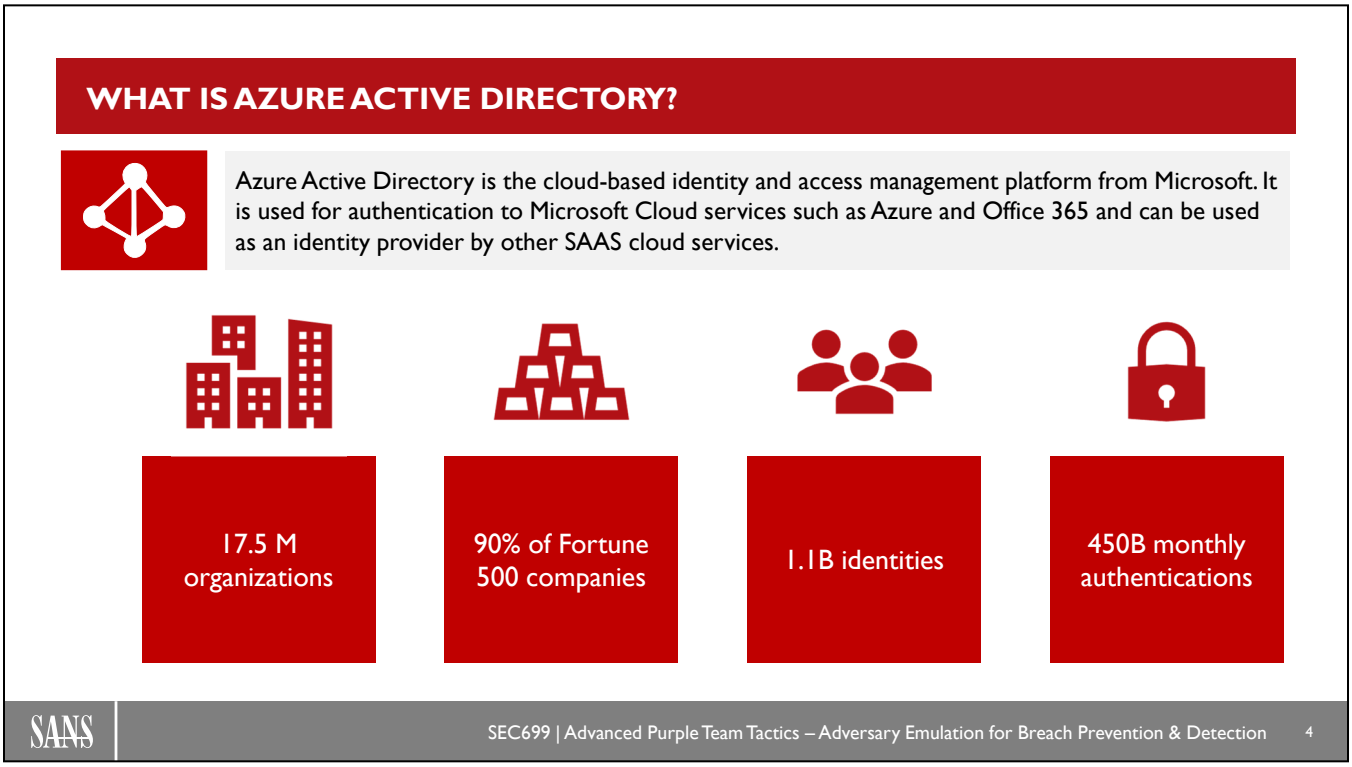
- Introduction & Key Tools
- Initial Access
- Lateral Movement
- Persistence
- **Azure AD & Emulation Plans**
- Adversary Emulation Capstone

SEC699.5

Azure AD
Azure AD Structure and Management
Azure AD Hybrid Authentication
Azure AD Authentication Methods
Azure AD Conditional Access
Introduction to Azure Identities
Azure AD Security Logging

Executing emulation plans
APT-28 Emulation Plan
Exercise: APT-28 Emulation Plan
APT-34 Emulation Plan
Exercise: APT-34 Emulation Plan
Turla Emulation Plan
Exercise: Turla Emulation plan

This page intentionally left blank.



What Is Azure Active Directory?

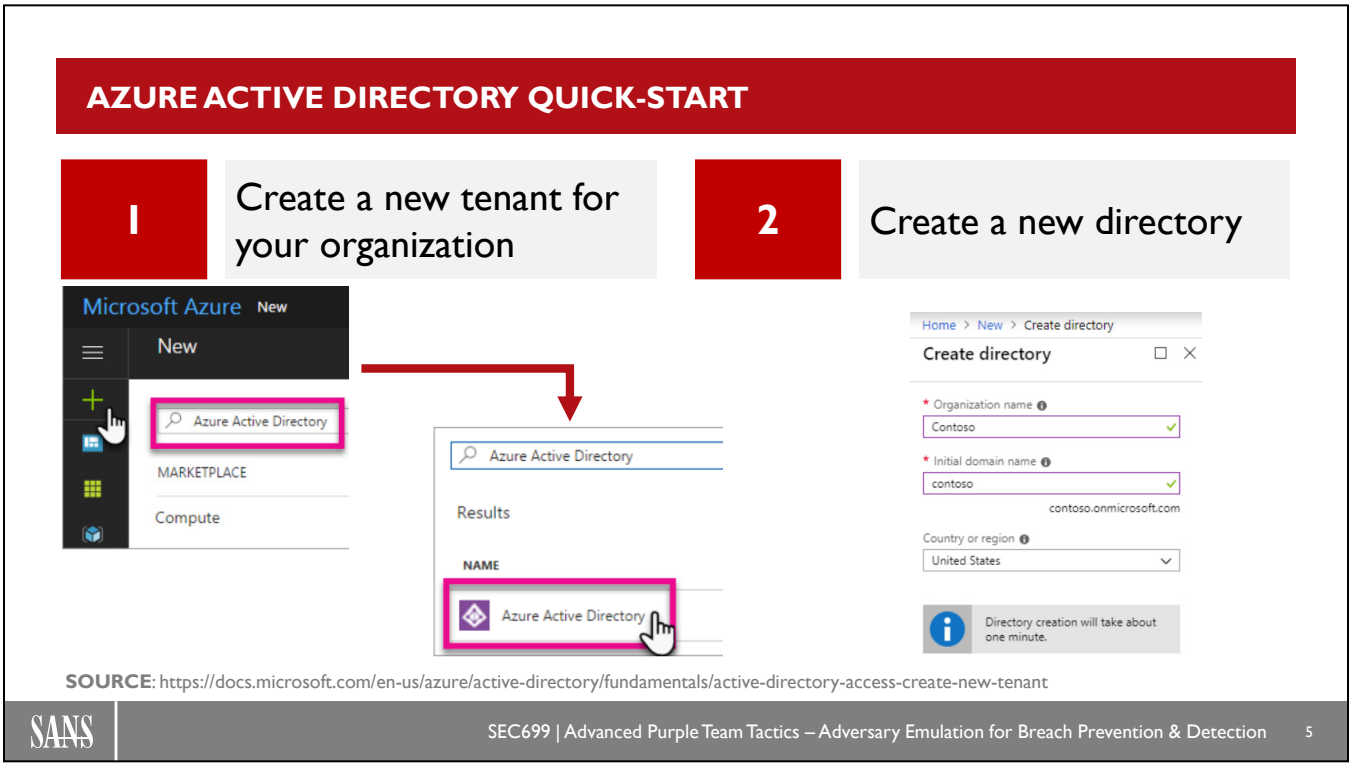
Azure Active Directory or Azure AD is a cloud service provided by Microsoft to support identity and access management integrated into several cloud services such as Azure and Office 365 applications. Azure AD is comparable with your on-prem Active Directory, which is used as an identity server for your on-prem components.

Azure AD is based on IaaS and provides a high scalable, always-on available identity service for your cloud applications. Several capabilities exist within Azure AD by providing identity as a service based on several open standard protocols. These will be described in the next sections.

Azure AD is widely used by many organizations, Microsoft published the following numbers in 2018 (from the Ignite conference):

- 17.5 Million organizations are using Azure AD as identity service for their cloud environment
- 90% of Fortune 500 Companies have implemented Azure AD
- In total, Microsoft has 1.1 Billion identities within their Azure AD service
- 450 Billion authentication requests are handled on a monthly basis

We can conclude that AZURE AD is becoming one of the de-facto standards for identity as a service within cloud environments. This is also caused by the easy integration into other platforms such as Office 365.



Azure Active Directory Quick-Start

Before setting up Azure AD, you need to have an Azure subscription, which can be created for free via https://azure.microsoft.com/en-us/free/. You can create a basic tenant for your organization via the Azure portal using a Global Administrator account. After you sign in to the Azure portal, you can create a new tenant for your organization. Your new tenant represents your organization and helps you to manage a specific instance of Microsoft cloud services for your internal and external users.

1. Creating a new tenant
   - Select Create a resource, select Identity, and then select Azure Active Directory.
2. Create a new directory
   - Type *the name of your organization* into the Organization name box.
   - Type <*initial domain name*> into the Initial domain name box.
   - Select the required region / country in the Country or region box.
   - Select Create.

Your new tenant is created with the domain <initial domain name>.onmicrosoft.com. You now have successfully created your first Azure AD tenant.

Reference:
https://docs.microsoft.com/en-us/azure/active-directory/fundamentals/active-directory-access-create-new-tenant

A QUICK WORD ON AZURE AD LICENSING

License	On-prem Sync	Password Protection	MFA	Self-Service Password Reset	Identity Protection	Priviledged Identity Management
Free	X (No writeback)	X (Cloud only)	X (App only)	X (Cloud only)		
Premium P1	X	X	X	X		
Premium P2	X	X	X	X	X	X

SOURCE: https://azure.microsoft.com/en-us/pricing/details/active-directory/

A Quick Word on Azure AD Licensing

Several licensing models exist within Azure AD. By default, the Free tier is enabled, which gives you basic identity and access management features. In case you want to use more advanced features, an upgrade to a premium license is required. The following licensing options exist within Azure AD:

Azure Active Directory Free

Provides user and group management, on-premises directory synchronization, basic reports, self-service password change for cloud users, and single sign-on across Azure, Office 365, and many popular SaaS apps.

Azure Active Directory Premium P1

In addition to the Free features, P1 also lets your hybrid users access both on-premises and cloud resources. It also supports advanced administration, such as dynamic groups, self-service group management, Microsoft Identity Manager (an on-premise identity and access management suite) and cloud write-back capabilities, which allow self-service password reset for your on-premises users.

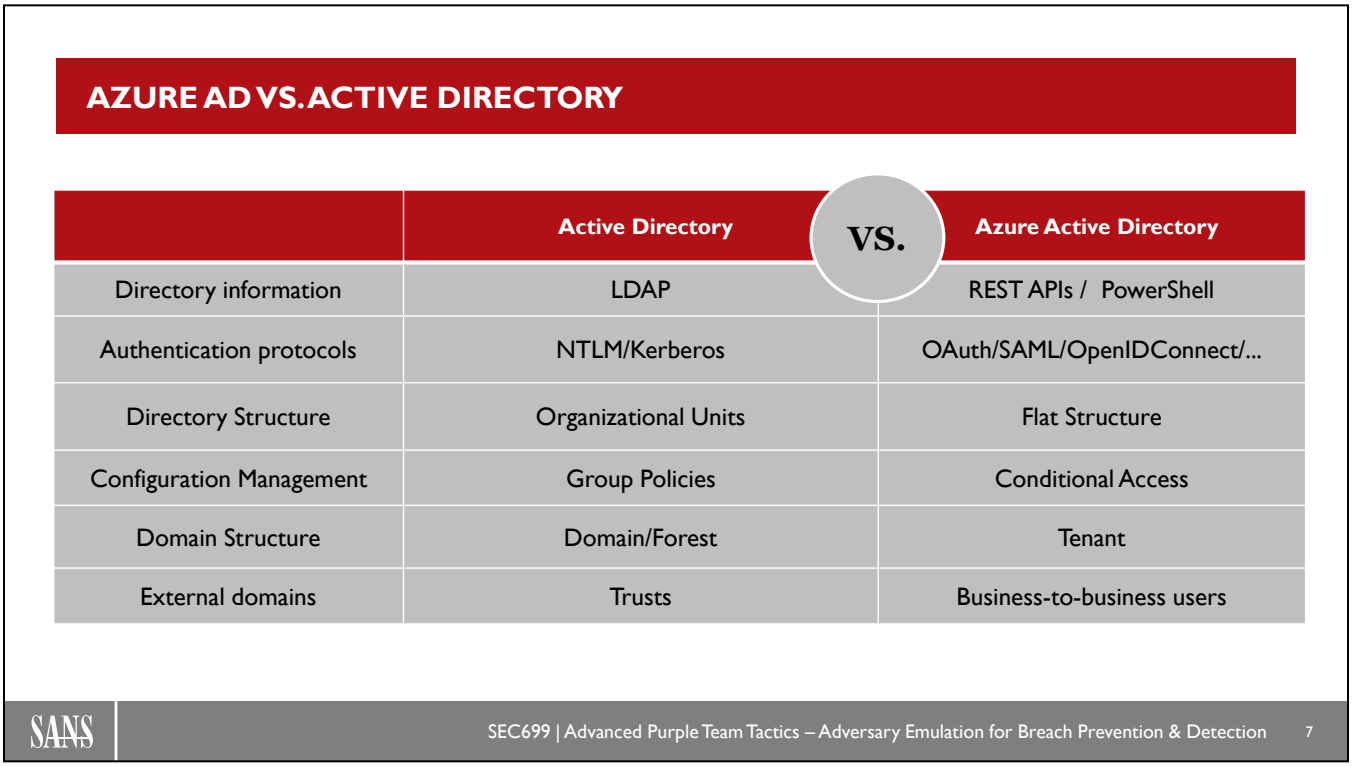
Azure Active Directory Premium P2

In addition to the Free and P1 features, P2 also offers Azure Active Directory Identity Protection to help provide risk-based Conditional Access to your apps and critical company data and Privileged Identity Management to help discover, restrict, and monitor administrators and their access to resources and to provide just-in-time access when needed.

"Pay as you go" feature licenses

You can also get additional feature licenses, such as Azure Active Directory Business-to-Customer (B2C). B2C can help you provide identity and access management solutions for your customer-facing apps.

Reference: https://azure.microsoft.com/en-us/pricing/details/active-directory/



Azure AD vs. Active Directory

It's important to note that, although they share a name, Azure Active Directory and a traditional on-premise Active Directory have very little in common…. Azure Active Directory is not "just a domain controller in the cloud"; they use different protocols, structures, trust and domain/ directory structures. So, let's take a look at the differences between a regular Active Directory and Azure AD:

Request directory information

- Windows Active Directory is essentially a database / phonebook that helps you organize your company users, groups, policies, and more. To request information from your Active Directory, the Lightweight directory access protocol (LDAP) protocol is used.
- Microsoft designed Azure AD to support internet (web) based services: Representational State Transfer (REST) API interfaces are used to retrieve specific information from Azure AD. Another option to request specific information is to execute directory queries via PowerShell.

Authentication Protocols

- Delegated authentication is supported by your on-prem domain controller through authentication protocols such as NTLM and Kerberos. The protocols were originally designed to work on a local network and are not suitable for internet-based authentication.
- Within Azure AD, several web-based protocols are supported to enable delegated authentication: Some of the most common authentication protocols supported are OAuth, SAML and OpenID Connect. This enables organizations to integrate Azure AD authentication with external applications or other SaaS solutions.

Directory Structure

- In Active Directory, you can use Organization Units to organize objects into logical administrative groups. An Organizational Unit or OU can contain objects such as accounts, computers, printers, applications, and many others.
- In Azure Active Directory the structure is flat; there is no logical structure that can be implemented such as organization units. Azure AD has only 3 type of objects: users, groups, and applications. It was designed as an identity provider for modern authentication mechanisms.

Configuration Management

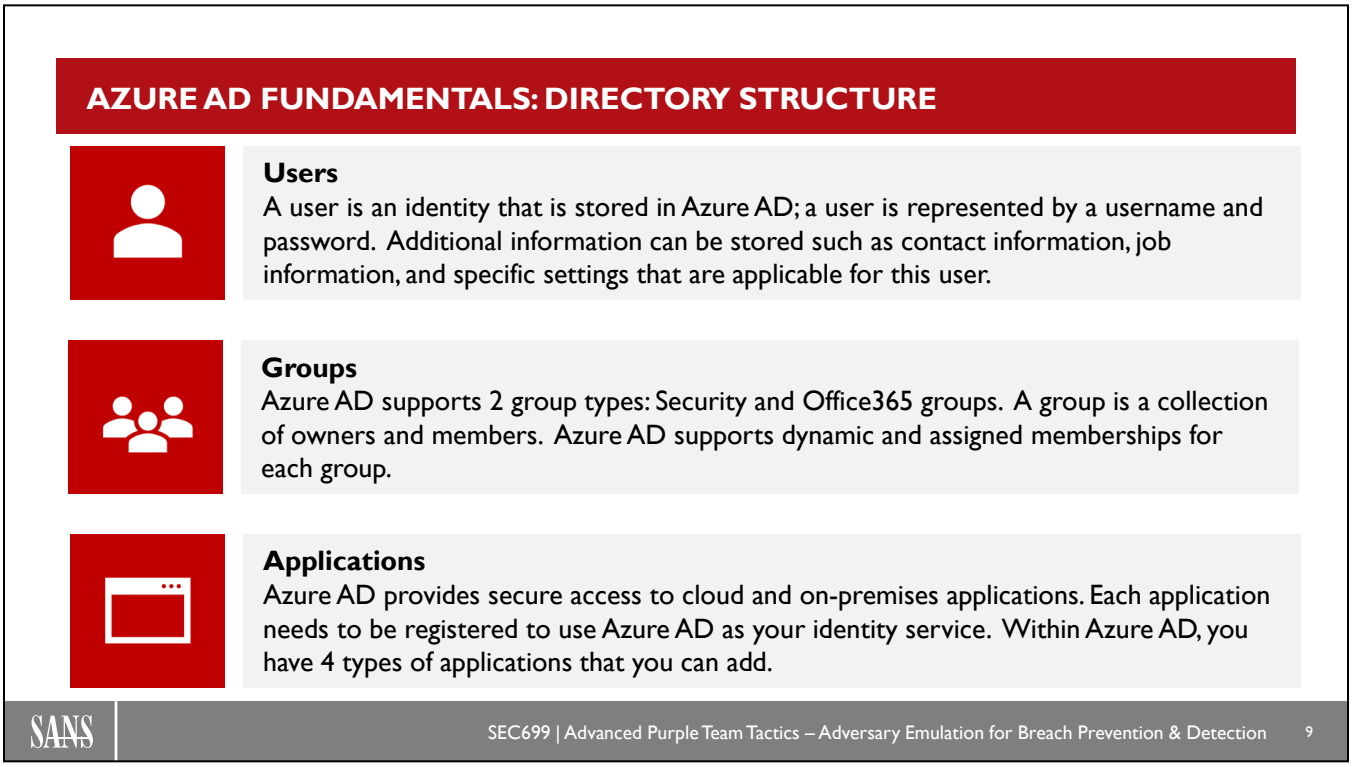
- The group policy feature is used to control the working environment via Windows Active Directory. These provide central management and configuration options for systems, applications, and users settings within the domain.
- Configuration management within Azure AD is very limited; the only type of centralized configuration tool is called conditional access. This policy only allows you to enforce specific conditions in case you want to grant access to applications. It should be noted that Microsoft Intune can be used to adapt configurations enterprise-wide.

Domain Structure

- At the top level, you can define forest and domain tree that include multiple domains. In most cases, these concepts are used for greater isolation and autonomy when necessary. Most common scenarios to have one forest and multiple domains are based on multiple sub-organizations within one forest, multiple departments within one large forest or based on geo-location.
- Azure AD uses the concept of a tenant: A tenant is a fully independent resource, and there is no parent-child relation possible between tenants.

External domains

- Active directory domain-to-domain communications are done via trusts. An AD trust is a secured authentication and communication channel between different entities. Trusts enable you to give access to specific users, groups, and computers across different domains.
- A typical case within Azure when using external domains is that an external user wants to have access to specific resources within your tenant. The concept of trusts doesn’t exist within Azure AD. In case business-to-business collaboration is required, you can invite guest users to access specific resources within your tenant.



Azure AD Fundamentals: Directory Structure

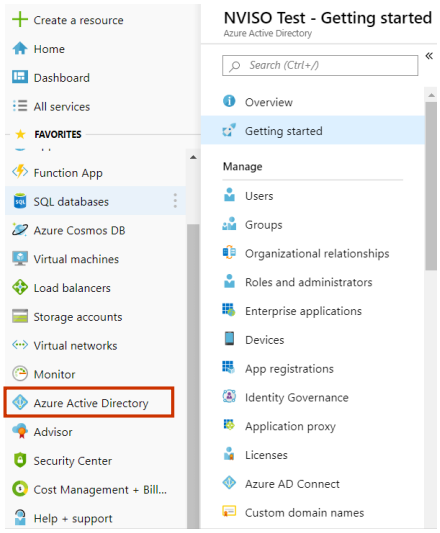
Azure Active Directory is an identity service with a flat structure. Within this structure, three common objects are configured related to identities within Azure AD. An identity is an object that can get authenticated via passwords, secret keys or certificates. Within each tenant, an Azure AD directory is a trusted and isolated structure. Within this directory, you can create the following identities:

- User accounts: This is an identity with data associated with it, an account can't exist without an identity. A user account is created through Azure AD or another Microsoft cloud service and these are stored within your Azure AD and accessible to your organization's cloud service subscription. User accounts are associated with work or school account within the Microsoft Cloud.
- Groups: Groups are created to manage permissions on a higher scale. Resources such as user accounts can be part of a group as a group owner or member. A set of access permissions can be assigned to a group instead of having to provide the rights one-by-one. The group owner is in control and can give management rights to other users as needed. There are 4 common ways to assign access rights:
  - Direct assignment: The resource owner assigns access directly to the user
  - Group assignments: The resource owner assigns access for all members of that group
  - Rule-based assignment: The resource owner uses a rule to define which users are assigned to that resource. The rule is based on attributes assigned to individual users.
  - External authority assignment: Access comes from an external resource, e.g., on-premises Active Directory.

- Applications: Azure AD provides secure access to cloud and on-prem applications. Within Azure AD, there are four main types of applications that can use Azure AD as an identity service provider:
    - Azure AD gallery application: This is a pre-generated list of more than 1,000 applications for single sign-on with Azure AD (e.g., Office 365).
    - On-premises application with application proxy: On-prem applications can use the application proxy to support single sign-on.
    - Custom developed applications: Custom applications can be registered within Azure AD to support single sign-on.
    - Non-Gallery applications: Other applications can be added to Azure AD, if they support username/password, SAML or OpenID Connect protocols.

More information about the Azure AD structure and terminology can be found here:
https://docs.microsoft.com/en-us/azure/active-directory/fundamentals/active-directory-whatis

AZURE AD FUNDAMENTALS: MANAGEMENT INTERFACES



Web Based Interface (https://portal.azure.com)

Azure Active Directory can be configured and managed via portal.azure.com and locate the Azure Active Directory Submenu. Most organizations use this interface for daily operations such as:

1. Management of user accounts
2. Management of user groups
3. Identity and access management monitoring

PowerShell Module (MSOnline)

Azure Active Directory PowerShell Module enables you to execute administration tasks via PowerShell. You can install this module by running the following command in PowerShell:

Install-Module AzureAD

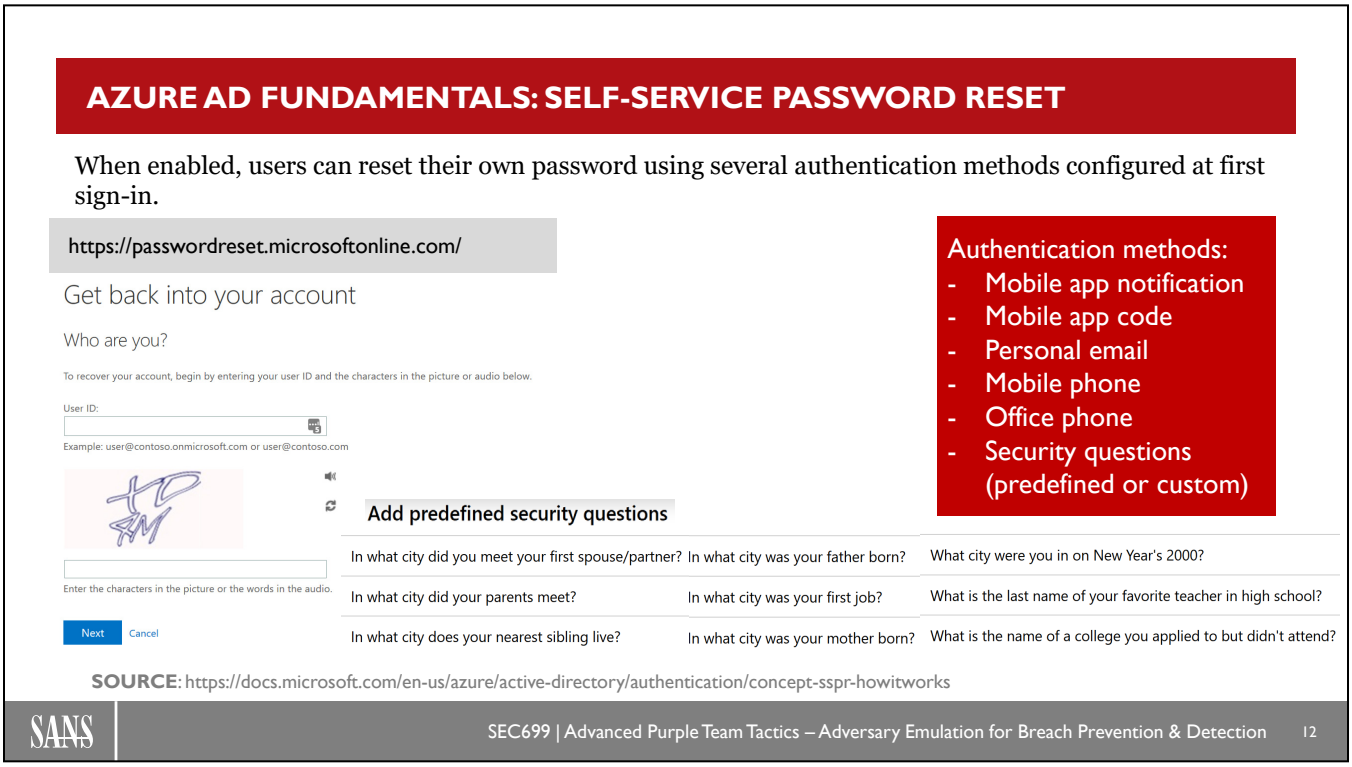
Azure AD Fundamentals: Management Interfaces

Managing Azure AD can be done in several ways. The most common method for configuring Azure AD is by using the web-based portal. In case you are a privileged account (Global Administrator), you can configure all the Azure AD features and use this interface for daily operations such as:

1. Managing user accounts
2. Managing user groups
3. Monitor identity and access management

A more advanced way to manage your Azure AD is via PowerShell. Microsoft has a complete reference guide listing all the Cmdlets that you can access via the PowerShell module. This is very useful in case you want to automate certain tasks, or you need to execute something in bulk on several objects. The Azure PowerShell model can be installed by running the following command *Install-Module AzureAD*. The cmdlets in the Azure AD module enable you to retrieve data from the directory, create new objects in the directory, update existing objects, remove objects, as well as configure the directory and its features.

A full reference to all Cmdlets can be found on the Microsoft website via the following link: https://docs.microsoft.com/nl-nl/powershell/azure/active-directory/overview?view=azureadps-2.0 or use the Get-Help <Cmdlet name>



Azure AD Fundamentals: Self-Service Password Reset

The self-service password reset feature within Azure AD activates the password reset portal. User will be able to reset their own password using several authentication methods to prove their identity. When a user goes to the password reset portal, a workflow is kicked off to determine:

- How should the page be localized?
- Is the user account valid?
- What organization does the user belong to?
- Where is the user's password managed?
- Is the user licensed to use the feature?

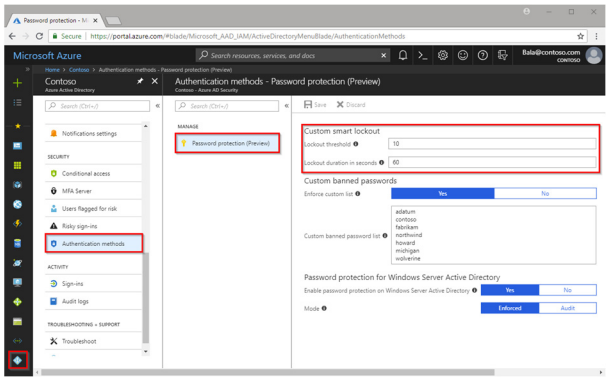
When SSPR is enabled, you must select at least one of the following options for the authentication methods. It is an industry best practice and also recommended by Microsoft that you choose two or more authentication methods so that your users have more flexibility in case they are unable to access their account when they need it. The following authentication methods are available:

- Mobile app notification
- Mobile app code
- Email
- Mobile phone
- Office phone
- Security questions

Users can only reset their password if they have data present in the authentication methods that the administrator has enabled. More information is available here: https://docs.microsoft.com/en-us/azure/active-directory/authentication/concept-sspr-howitworks

AZURE AD FUNDAMENTALS: SMART LOCKOUT

Smart lockout is a protection feature within Azure AD to lock out adversaries that are trying to brute force passwords. This feature can recognize sign-ins coming from valid users and treat them differently than those coming from unknown sources. This feature is enabled by default.



Thresholds can be configured to lock a specific user. Recognition is based on trusted locations and anomalies detected in the authentication requests.

Enable a custom list of bad passwords that can't be used. Azure AD will validate the hashes and report on users that are using weak passwords.

You can also synchronize these password protection mechanisms with your on-premises domain controllers (agent required).

SOURCE: https://docs.microsoft.com/en-us/azure/active-directory/authentication/howto-password-smart-lockout

Azure AD Fundamentals: Smart Lockout

Smart lockout is a protection feature within Azure AD to lock out adversaries that are trying to brute force passwords. This feature can recognize sign-ins coming from valid users and treat them differently than those coming from unknown sources. This feature is enabled by default and offers the following interesting functions:

- By default, smart lockout locks the account from sign-in attempts for one minute after 10 failed attempts. The account locks again after each subsequent failed sign-in attempt, for one minute at first and longer in subsequent attempts.
- Smart lockout tracks the last three bad password hashes to avoid incrementing the lockout counter for the same password. If someone enters the same bad password multiple times, this behavior will not cause the account to lockout. This can be integrated with hybrid deployments, using password hash sync or pass-through authentication to protect on-premises Active Directory accounts from being locked out by attackers.
- Each Azure Active Directory data center tracks lockout independently. A user will have (threshold_limit * datacenter_count) number of attempts, if the user hits each data center.
- Smart Lockout uses familiar location vs. unfamiliar location to differentiate between a bad actor and the genuine user. Unfamiliar and familiar locations will both have separate lockout counters.
- The feature can be used to ban a custom list of weak passwords:
  - The custom banned password list can contain up to 1,000 terms.
  - The custom banned password list is case-insensitive.

- The custom banned password list considers common character substitution. Example: "o" and "0" or "a" and "@".
- The minimum string length is four characters, and the maximum is 16 characters.

Smart lockout is always on for all Azure AD customers with these default settings that offer an interesting mix of security and usability. Customization of the smart lockout settings, with values specific to your organization, requires paid Azure AD licenses for your users.

Reference:

https://docs.microsoft.com/en-us/azure/active-directory/authentication/howto-password-smart-lockout

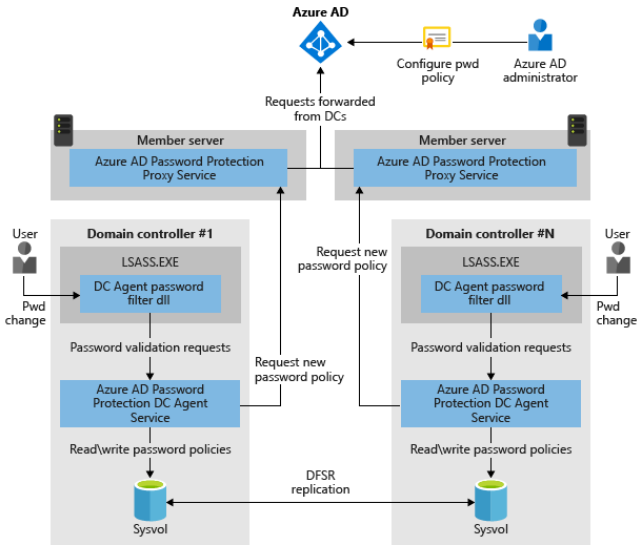
AZURE AD FUNDAMENTALS: PASSWORD PROTECTION

Prevent users from using bad passwords based on a global and custom (max 1,000) banned password list in the cloud and on-premise.

Protect against password spraying attacks and dictionary attacks based on third-party compromised password lists.

New passwords are normalized and fuzzy matching is used to see if it is found in the global or custom password lists.

SOURCE: https://docs.microsoft.com/en-us/azure/active-directory/authentication/howto-password-ban-bad-on-premises-deploy



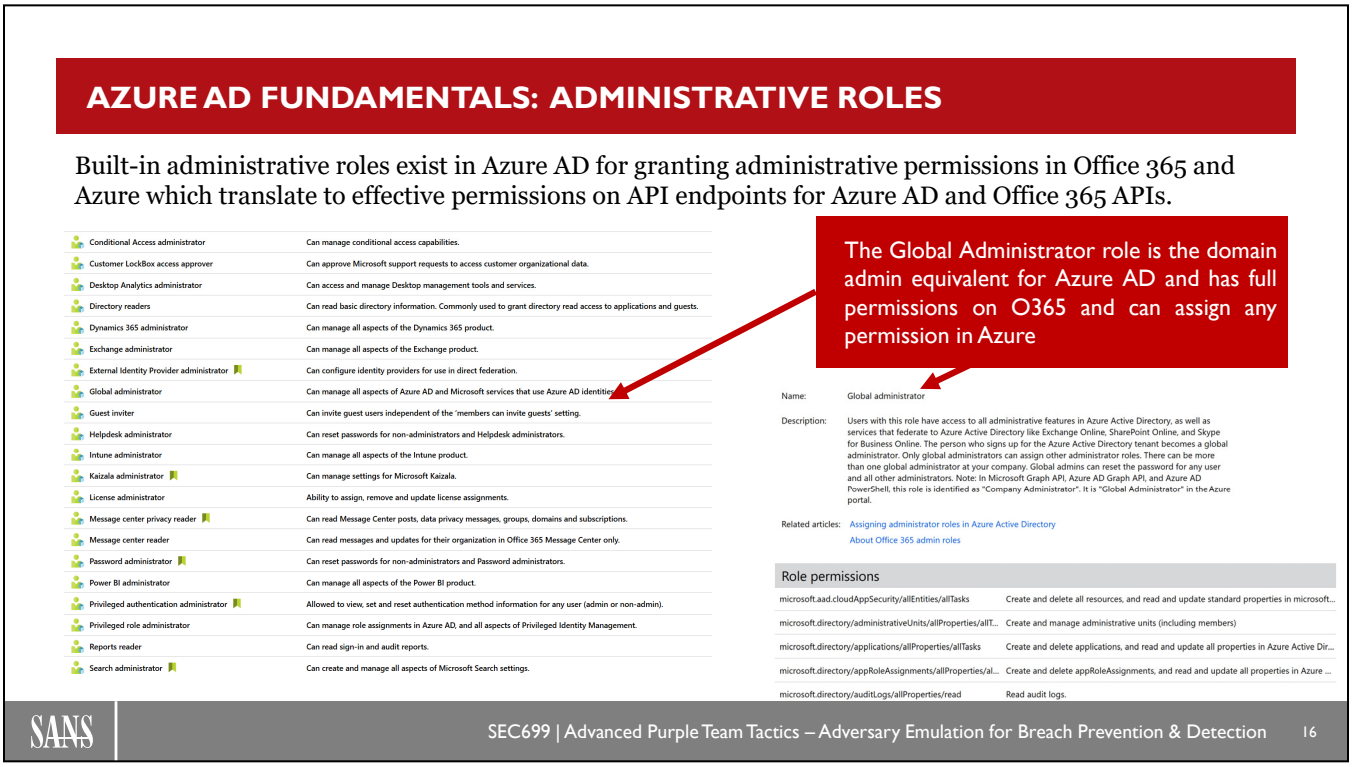
Azure AD Fundamentals: Password Protection

Azure AD Password Protection allows you to eliminate easily guessed passwords and customize lockout settings for your environment. Using it can significantly lower the risk of compromise by a password spray attack. Microsoft maintains and updates a list of weak and vulnerable passwords, but as tenant owner, you are also able to define your own bad passwords. The main use cases for Azure AD Password Protection are:

- Prevent users from using bad passwords based on a global and custom (max 1,000) banned password list in the cloud and on-premise.
- Protect against password spraying attacks and dictionary attacks based on third-party compromised password lists.
- New passwords are normalized, and fuzzy matching is used to see if it is found in the global or custom password lists.

The password policy that you define in Azure can also be synced to your on-premises directory by using Azure AD Password protection DC Agent software. This agent can be found here: https://www.microsoft.com/en-us/download/details.aspx?id=57071

For more information and details, you can find a full overview here: https://docs.microsoft.com/en-us/azure/active-directory/authentication/concept-password-ban-bad



Azure AD Fundamentals: Administrative Roles

Within your on-prem Active Directory, you can define very fine-grained administrative roles; within Azure AD, you have a predefined list of roles. Roles are updated on a regular basis by Microsoft to provide granular permissions. This makes it a difficult task to manage administrator roles; it is recommended to review this list on a regular basis and ensure you are using the right roles for your needs. Every tenant needs a global administrator. This role is comparable to an on-prem domain (or enterprise) administrator. The role gives you full permissions on Azure and Office 365. It is recommended to use this role only in very specific situations and never for normal daily operations. Another best practice is to create one emergency account with the role of global administrator and set a very strong password that is stored offline and can only be used if all other users are locked out from your environment.

Below, we have listed some of the key roles that are described by Microsoft:

- Global Administrator: Users who are assigned to the Global administrator role can read and modify every administrative setting in your Azure AD organization. By default, the person who signs up for an Azure subscription is assigned the Global administrator role for the Azure AD organization. Only Global Administrators and Privileged Role Administrators can delegate administrator roles.

- Application Administrator: Users in this role can create and manage all aspects of enterprise applications, application registrations, and application proxy settings. This role also grants the ability to consent to delegated permissions, and application permissions excluding Microsoft Graph and Azure AD Graph.

- Authentication Administrator: Users with this role can set or reset non-password credentials and can update passwords for all users. Authentication Administrators can require users to re-register against existing non-password credential (for example, MFA or FIDO) and revoke/remember MFA on the device.

- Global Reader: Users in this role can read settings and administrative information across Microsoft 365 services but can't take management actions. Global reader is the read-only counterpart to Global administrator. Assign Global reader instead of Global administrator for planning, audits, or investigations.

- Helpdesk Administrator: Users with this role can change passwords, invalidate refresh tokens, manage service requests, and monitor service health. Invalidating a refresh token forces the user to sign in again. Helpdesk administrators can reset passwords and invalidate refresh tokens of other users who are non-administrators.

- Security Administrator: Users with this role have permissions to manage security-related features in the Microsoft 365 security center, Azure Active Directory Identity Protection, Azure Information Protection, and Office 365 Security & Compliance Center.

- Security operator: Users with this role can manage alerts and have global read-only access on security-related feature, including all information in Microsoft 365 security center, Azure Active Directory, Identity Protection, Privileged Identity Management, and Office 365 Security & Compliance Center.

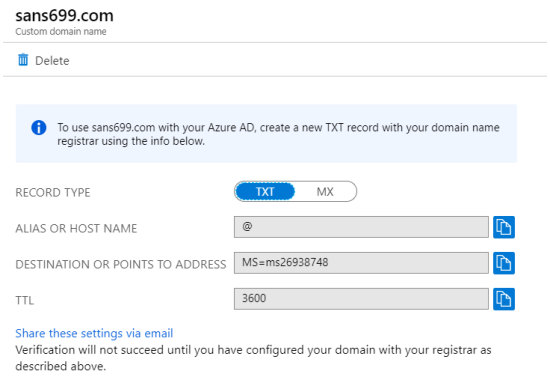
- Security Reader: Users with this role have global read-only access on security-related feature, including all information in Microsoft 365 security center, Azure Active Directory, Identity Protection, Privileged Identity Management, as well as the ability to read Azure Active Directory sign-in reports and audit logs, and in Office 365 Security & Compliance Center.

Many other administrative roles are described on the Microsoft Azure website; you can find all descriptions at https://docs.microsoft.com/en-us/azure/active-directory/roles/permissions-reference

AZURE AD ATTACK STRATEGIES: RECONNAISSANCE

Creating an O365 or Azure AD tenant automatically creates a domain <tenant name>.onmicrosoft.com. Custom domains can also be added to your tenant; owner validation is done via dedicated DNS records.

- Adversaries will fingerprint your DNS records and try to identify these TXT or MX records
- SharePoint Online, Exchange Online and other MS O365 services can be fingerprinted (default login window)
- Looking at the actual MX records or mail headers, in case *.mail.protection.outlook.com is used, you know the organization is using Azure AD



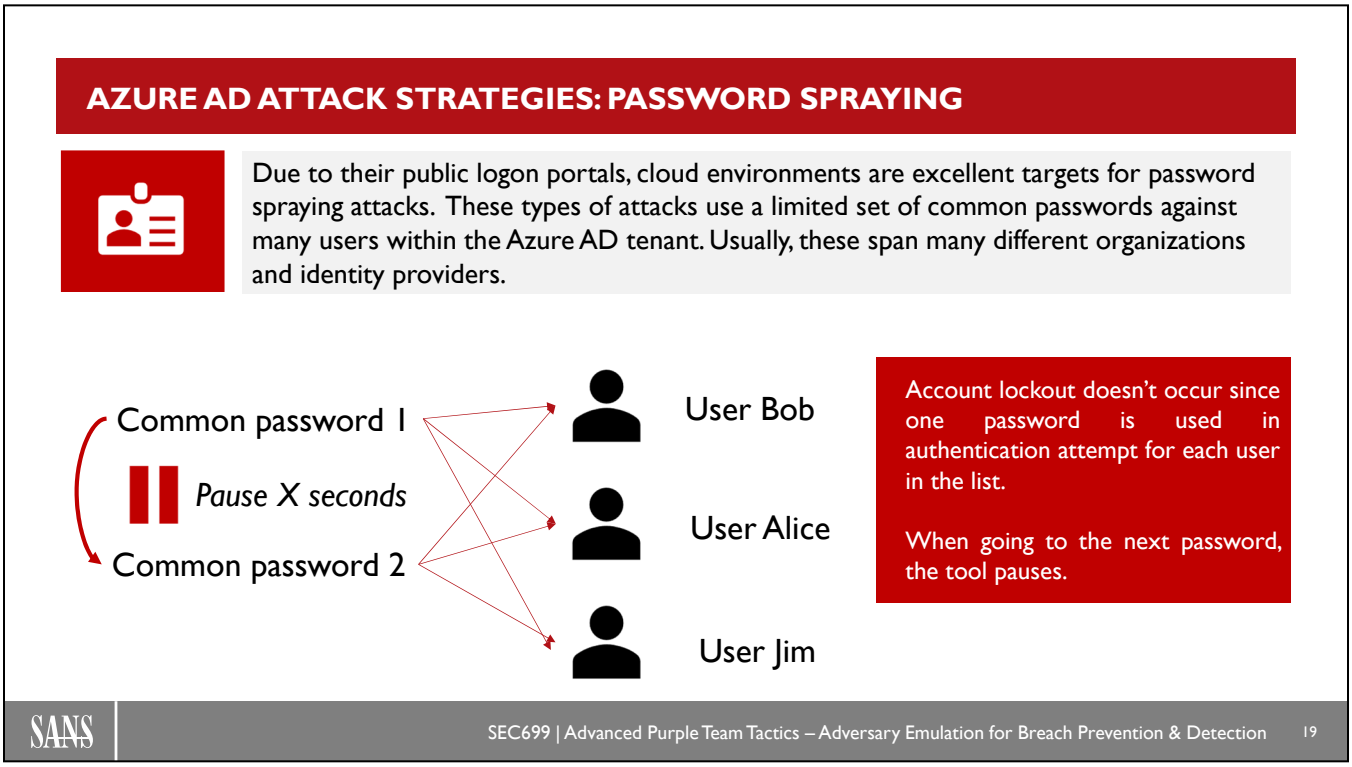
Azure AD Attack Strategies: Reconnaissance

Using Azure AD allows an adversary to perform reconnaissance in a rather simple way: Creating an O365 or Azure AD tenant automatically creates a domain <tenant name>.onmicrosoft.com. Custom domains can also be added to your tenant; owner validation is done via dedicated DNS records.

Adversaries will request these DNS records and try to identify the information contained in the TXT or MX records. Microsoft has 2 options to perform domain validation, where you as an organization need to change your DNS settings with your registrar:

1. A TXT record that has the value of MS=<random number> with a TTL of 3600 seconds
2. A MX record with a very high priority, destination of the MX record ms<random number>.msv1.invalid with a TTL of 3600 seconds

Furthermore, in case the organization is running Azure AD with SaaS (Software-as-a-Service) applications, the adversary can look at the DNS MX records, DNS TXT records or federation records such as adfs.targetdomain.com, sso.targetdomain.com, sts.targetdomain.com, and many more.



Azure AD Attack Strategies: Password Spraying

Due to their public logon portals, cloud environments are excellent targets for password spraying attacks. These types of attacks use a limited set of common passwords against many users within the Azure AD tenant. Usually, these span many different organizations and identity providers.

What is password spraying?

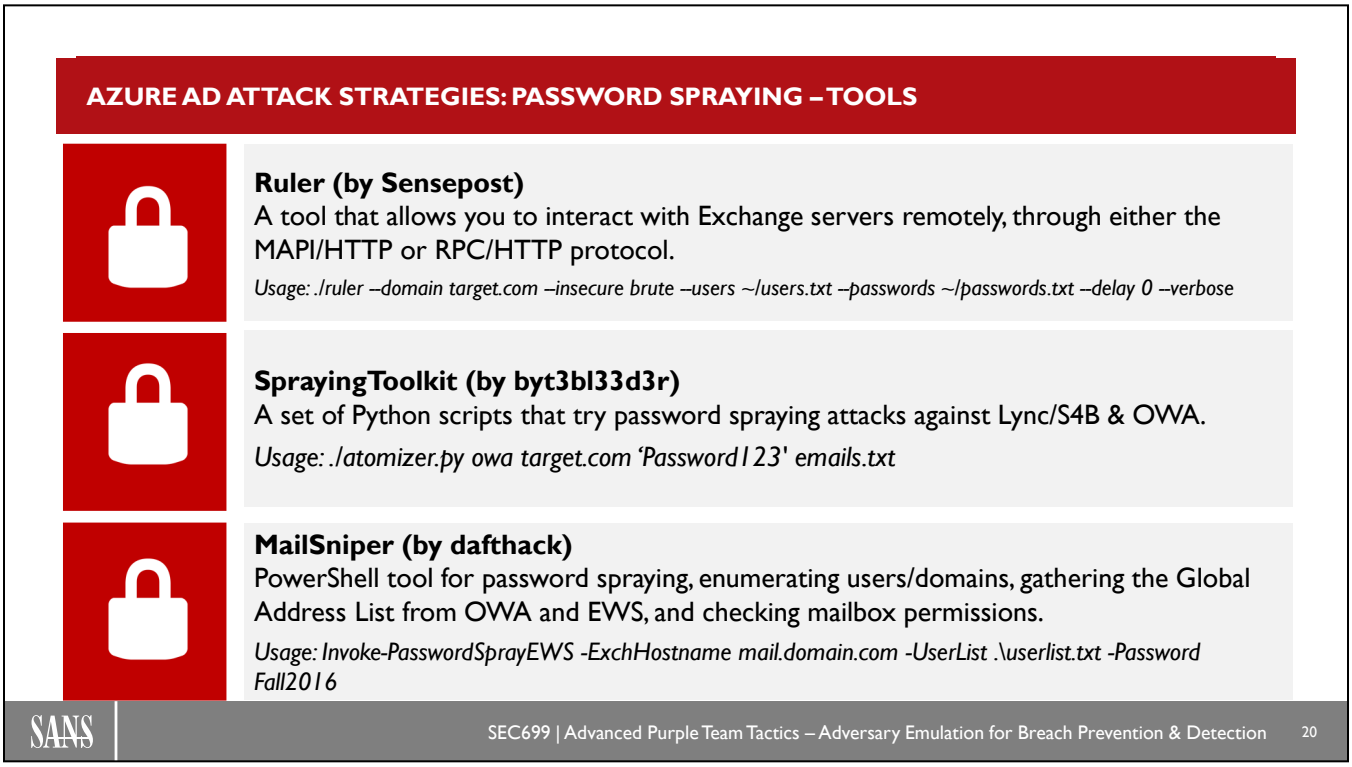
Traditional brute-force attacks attempt to gain unauthorized access to a single account by guessing the password. This can quickly result in the targeted account being locked-out, as commonly used account-lockout policies allow for a limited number of failed attempts (typically three to five) during a set period of time. During a password spray attack, the malicious actor is going to try a limited set of commonly used passwords against many accounts before moving on to attempt a second password, and so on. This technique allows the actor to remain undetected by avoiding rapid or frequent account lockouts.

In our example, the attacker obtained a list of all users and is going target these users with one common password. To remain undetected, the actor will pause the script for several seconds and then move on to a second common password. Again, targeting all users in our scope: Bob, Jim, and Alice.

Password spray campaigns typically target single-sign-on (SSO) and cloud-based applications utilizing federated authentication protocols.

Reference:

https://www.helpnetsecurity.com/2019/03/20/imap-based-password-spraying/



Azure AD Attack Strategies: Password Spraying – Tools

Several tools exist to assist you with password spraying attacks and user enumeration. Once you have obtained a list of commonly used password and users for that organization, you can use some of the following tools to attack specific cloud services. Note that in most cases, password spraying attacks target legacy protocols, as modern authentication mechanisms often have controls against password spraying (e.g., two-factor authentication, conditional access,…)

In the enterprise world, we often still encounter support for legacy authentication protocols to support older Office outlook clients, Mailing tools,… Azure AD is often used together with other O365 services; this means that we could abuse the mail service to actually attack the Azure AD accounts!

The following list is a limited set of tools that can be used for password spraying or user enumeration attacks:

- **Ruler (by Sensepost):** Ruler was developed by Sensepost. It allows you to interact with Exchange servers remotely, through either the MAPI/HTTP or RPC/HTTP protocol. The main aim is abusing client-side Outlook features and gaining a shell remotely. Ruler has multiple functionalities but some interesting functions from our perspective are: User Enumeration, create new malicious mail rules, dump the Global Address List (GAL). The Ruler source code can be found here: https://github.com/sensepost/ruler

- **Spraying Toolkit (by byt3bl33d3r):** A set of Python scripts/utilities that try to make password spraying attacks against Lync/S4B & OWA a lot quicker, less painful, and more efficient. Atomizer is one of the core scripts within the password spraying toolkit. It can execute password spraying via lync, OWA, or IMAP and different file types can be used as password and user list. It allows you to Scrape Google and Bing for LinkedIn profiles, automatically generate emails from the profile names using the specified pattern, and perform password sprays in real-time. The Spraying Toolkit source can be found here: https://github.com/byt3bl33d3r/SprayingToolkit

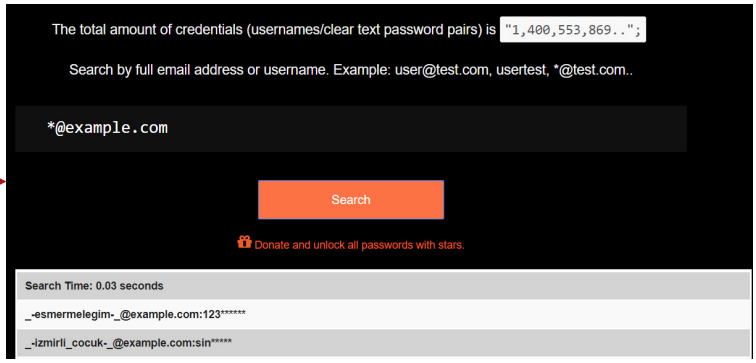
- **MailSniper:** MailSniper is a penetration testing tool for searching through email in a Microsoft Exchange environment. This tool based on PowerShell includes additional modules for password spraying, enumerating users/domains, gathering the Global Address List from OWA and EWS, and checking mailbox permissions for every Exchange user at an organization. The MailSniper source can be found here: https://github.com/dafthack/MailSniper

In case modern authentication is enabled, adversaries will have to rely on other tools, which will be discussed in a later section.

AZURE AD ATTACK STRATEGIES: PASSWORD REUSE ATTACKS

Password reuse or replay attacks are highly effective against public cloud environments if MFA (Multi-Factor Authentication) is not enabled. Employees still often use the same password on personal and corporate accounts. Sites such as "Have I been pwned" and "ghostproject" can assist you to gather a list of passwords that are possibly also used for professional accounts.

Search for a professional account and validate if a password is compromised.



Compromised Credentials

Azure AD Attack Strategies: Password Reuse Attacks

Password reuse or replay attacks are highly effective against public could environments if MFA (Multi-Factor Authentication) is not enabled. Employees still often use the same password on personal and corporate accounts. Sites such as "Have I been pwned" and "ghostproject" can assist you to gather a list of passwords that are possibly also used for professional accounts.

Once compromised accounts are identified for a target organization, they can be reused against O365 and their corresponding Azure AD account. It's useful to note that Microsoft has a built-in security control that aims to prevent such attacks: Microsoft monitors leaked credentials, and in case the hashes of a compromised password are stored in Azure AD, Microsoft will generate a report highlighting these compromised users.

Course Roadmap

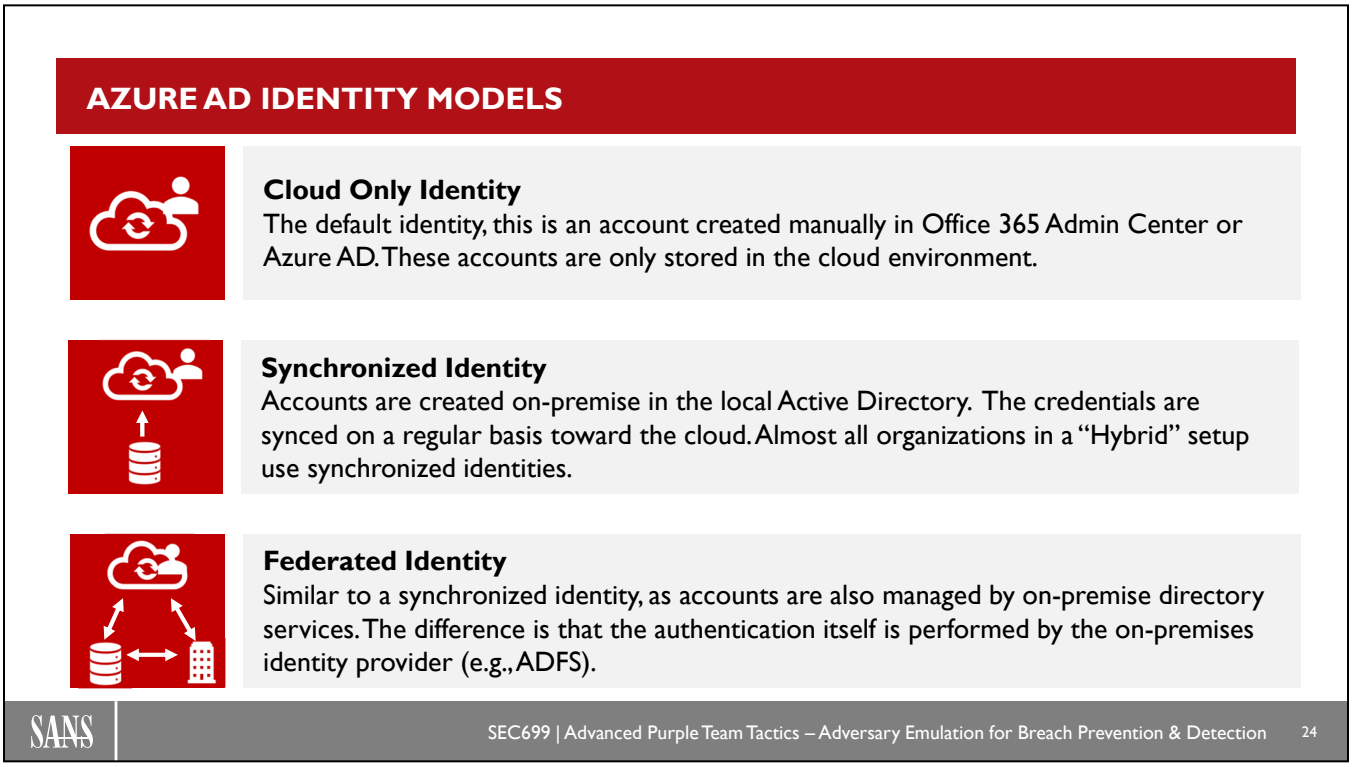
- Introduction & Key Tools
- Initial Access
- Lateral Movement
- Persistence
- **Azure AD & Emulation Plans**
- Adversary Emulation Capstone

SEC699.5

Azure AD
Azure AD Structure and Management
Azure AD Hybrid Authentication
Azure AD Authentication Methods
Azure AD Conditional Access
Introduction to Azure Identities
Azure AD Security Logging

Executing emulation plans
APT-28 Emulation Plan
Exercise: APT-28 Emulation Plan
APT-34 Emulation Plan
Exercise: APT-34 Emulation Plan
Turla Emulation Plan
Exercise: Turla Emulation plan

This page intentionally left blank.



Azure AD Identity Models

Within the identity service of Azure AD, three different types of identity models exist:

Cloud Only Identity

This means that the identity only exists in Azure AD; there is no link between Azure AD and other directories to import identities. This is the default setting. When an Office 365 account is created, this account is stored in Azure AD as a cloud-only identity.

Synchronized Identity

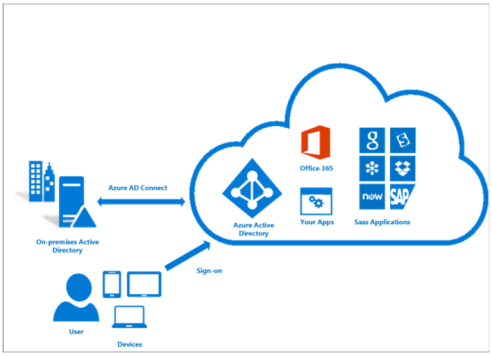
For larger organizations in hybrid mode, this is the most common setup, as identities are managed on-premise and synchronized on a regular basis toward the cloud environment. Typically, the synchronization process is an agent running on a domain controller that populates Azure AD.

Federated Identity

A federated identity is similar to a synchronized identity, except that in case of a federated identity, the on-premises identity provider is responsible for the actual authentication. In case users are requesting access to the cloud, they will be redirected to the on-premises idenity provider, and, after succesful authentication, they will be redirected to the cloud service.

INTRODUCING AZURE AD CONNECT

Azure AD Connect allows you to integrate your on-premises directories with Azure AD. Users can use one common identity to access both on-prem resources and cloud resources.



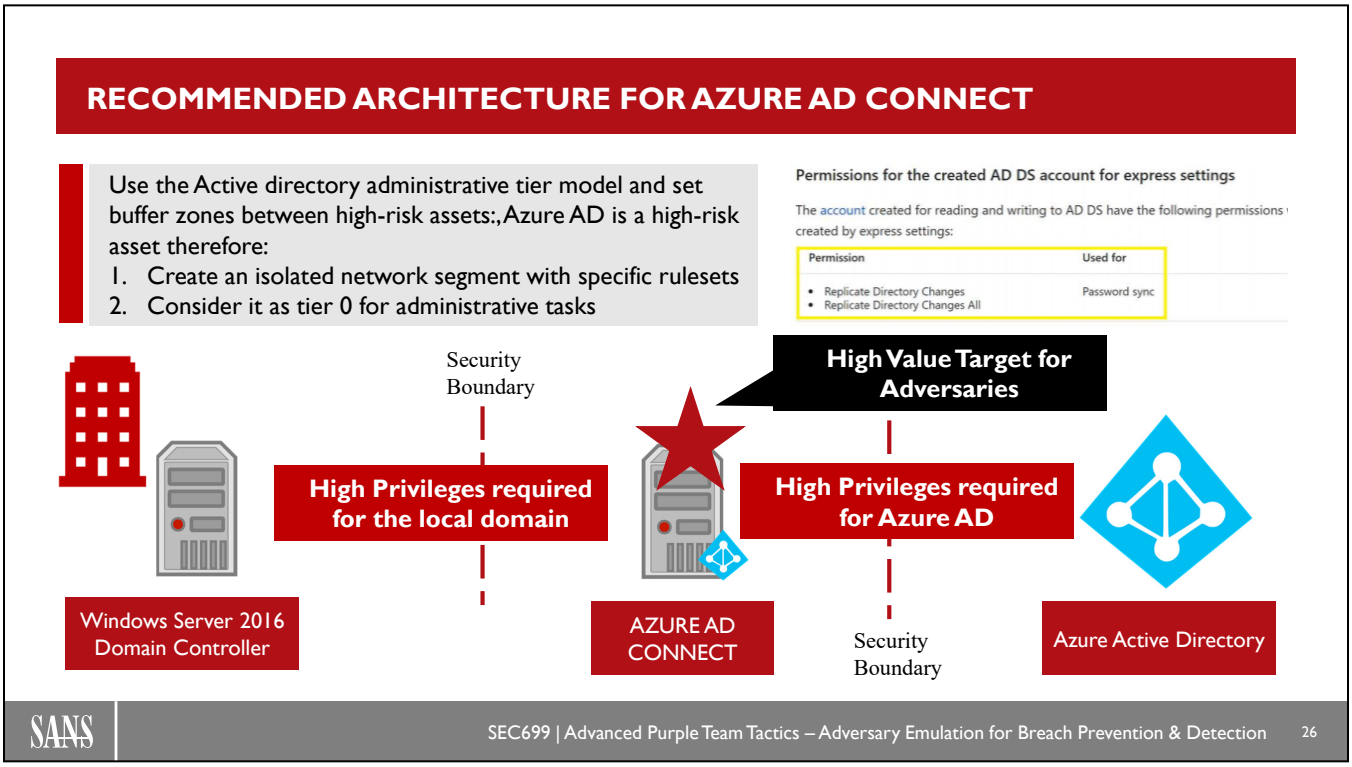
- Users can use a single identity to access on-premises applications and cloud services such as Office 365.
- Azure AD Connect provides easy deployment experience for synchronization and sign-in.
- Integrates with the newest capabilities for hybrid authentication scenarios. Azure AD Connect is the preferred method to establish synchronization between your directories.

SOURCE: https://docs.microsoft.com/en-us/azure/active-directory/hybrid/whatis-azure-ad-connect

Introducing Azure AD Connect

Azure AD Connect is a tool designed by Microsoft to synchronize local identities with cloud identities. This software enabled you to connect your on-premises Active Directory with Azure AD. It has several functionalities, but the main goal is to synchronize users and groups from your local domain toward Azure AD. The benefit is that end-users can use a single identity to access on-prem and cloud services. The default synchronization interval is set to 30 minutes, but this setting can be changed to meet your requirements.

Note that Azure AD Connect is a common target attacker, as it needs to run both with high privileges on your local Active Directory and with high privileges in your Azure AD environment. Typically, the Azure AD Connect user is configured upon installation of the Azure AD Connection and has the following format: “MSOL_XXXXXX”.



Recommended Architecture for Azure AD Connect

Microsoft has developed a set of best practices on how to set up "buffer zones" between different enterprise tiers.

As Azure AD Connect requires high privileges in both environments, we must consider this as one of our most critical servers. Dutch security researcher Dirk-jan Mollema presented a highly interesting talk, "I'm in your cloud … reading everyone's mail", which is a perfect example of what could go wrong when an attacker abuses this service and the accounts that are associated with Azure AD. We will deep-dive into this attack strategy a little later in the courseware.

Furthermore, Adam Chester (@_xpn_) published a very interesting blog post on attacking Azure AD Connect at https://blog.xpnsec.com/azuread-connect-for-redteam/.

References:

The full Microsoft documentation on the "Tiered Admin Model" can be bound at https://docs.microsoft.com/en-us/security/compass/privileged-access-access-model

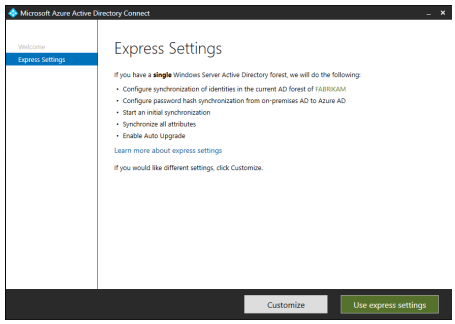
Setting up Azure AD Connect: https://docs.microsoft.com/en-us/azure/active-directory/hybrid/how-to-connect-sync-whatis

CONFIGURING AZURE AD CONNECT (1)

Azure AD Connect Express Settings are used when you have a single-forest topology and password hash synchronization for authentication. Express Settings is the default option and is used for most deployment scenarios.

Let's assume we have a single forest and want to use Azure AD Connect with Express settings:

1	Sign in as a local admin on the Server you wish to install Azure Connect AD on
2	Navigate and double-click AzureADConnect.msi
3	Select the box agreeing to the licensing terms and click "Continue"
4	Express Settings screen, click "Use Express Settings"



Configuring Azure AD Connect (1)

Azure AD Connect Express Settings are used when you have a single-forest topology and password hash synchronization for authentication. Express Settings is the default option and is used for most deployment scenarios.

Let's assume we have a single forest and want to use Azure AD Connect with Express Settings. This means we will need to run through the following steps:

1. To install the Azure AD Connect agent, you need local admin privileges on the sever you wish to install Azure Connect AD on. The MSI package can be downloaded via the official Microsoft website: https://www.microsoft.com/en-us/download/details.aspx?id=47594

2. Transfer the MSI package toward the server and run the AzureADConnect.msi package.

3. Review and accept the licensing terms and select the Express Settings to continue the installation.

In case you want to use specific settings, for example if you have custom identifiers, referred to as anchor points or you want to use other authentication methods, you need to use the custom settings and define your own paramteres.

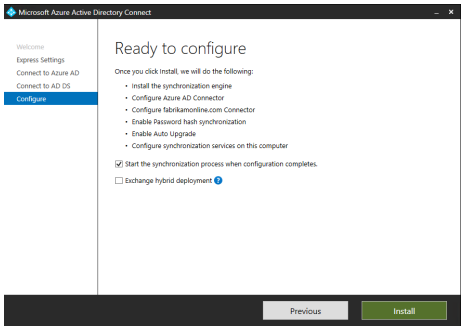
The full installation guide can be found here:
https://docs.microsoft.com/en-us/azure/active-directory/hybrid/how-to-connect-install-express

CONFIGURING AZURE AD CONNECT (2)

In the next steps, we need to step up the connection to Azure AD. This will require you to have both the **global administrator account and enterprise admin account** for your on-premises forest.

Let's assume we have a single forest and want to use Azure AD Connect with express settings:

5	Connect to Azure AD screen, enter the credentials of a global administrator for your Azure AD
6	Connect to AD DS screen, enter the username and password for an enterprise admin account
7	Azure AD sign-in configuration page only shows if you did not complete verify your domains.
8	On the Ready to configure screen, click Install.



Configuring Azure AD Connect (2)

The next screen in the express installation guide is called "Connect to Azure AD". In the next steps, we need to step up the connection to Azure AD. This will require you to have both the global administrator account and enterprise admin account for your on-premises forest. Once the credentials are given and the server has the required outbound access to the internet, the agent can connect to your cloud environment and register the agent.

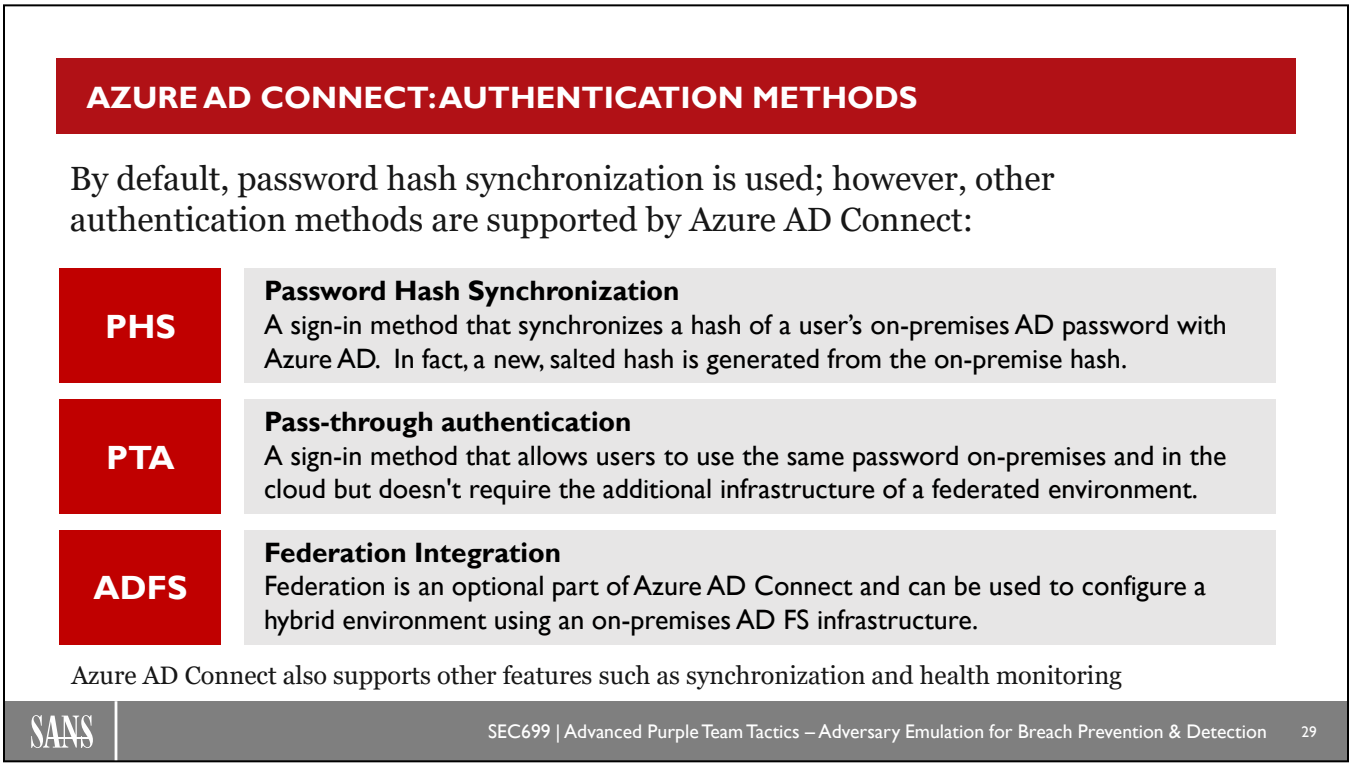
The next screen requests information about your on-premises domain (including the aforementioned enterprise admin account). Additionally, in case your target domains are not yet configured, you can add and verify your domains via the next screen.

That's it! You will now get an overview of all the settings that will be installed, and Azure AD Connect is ready to configure synchronization using the express settings.

The two checkboxes will allow you to:

- Unselect the "Start the synchronization process as soon as configuration completes" checkbox. You should unselect this checkbox if you want to do additional configuration prior to first use (e.g., additional filtering);
- If you have Exchange in your on-premises Active Directory, then you also have an option to enable "Exchange Hybrid deployment". Enable this option if you plan to have Exchange mailboxes both in the cloud and on-premises at the same time.

The full installation guide can be found here:
https://docs.microsoft.com/en-us/azure/active-directory/hybrid/how-to-connect-install-express



Azure AD Connect: Authentication Methods

The following authentication methods are supported and listed in the Azure AD Connect documentation:

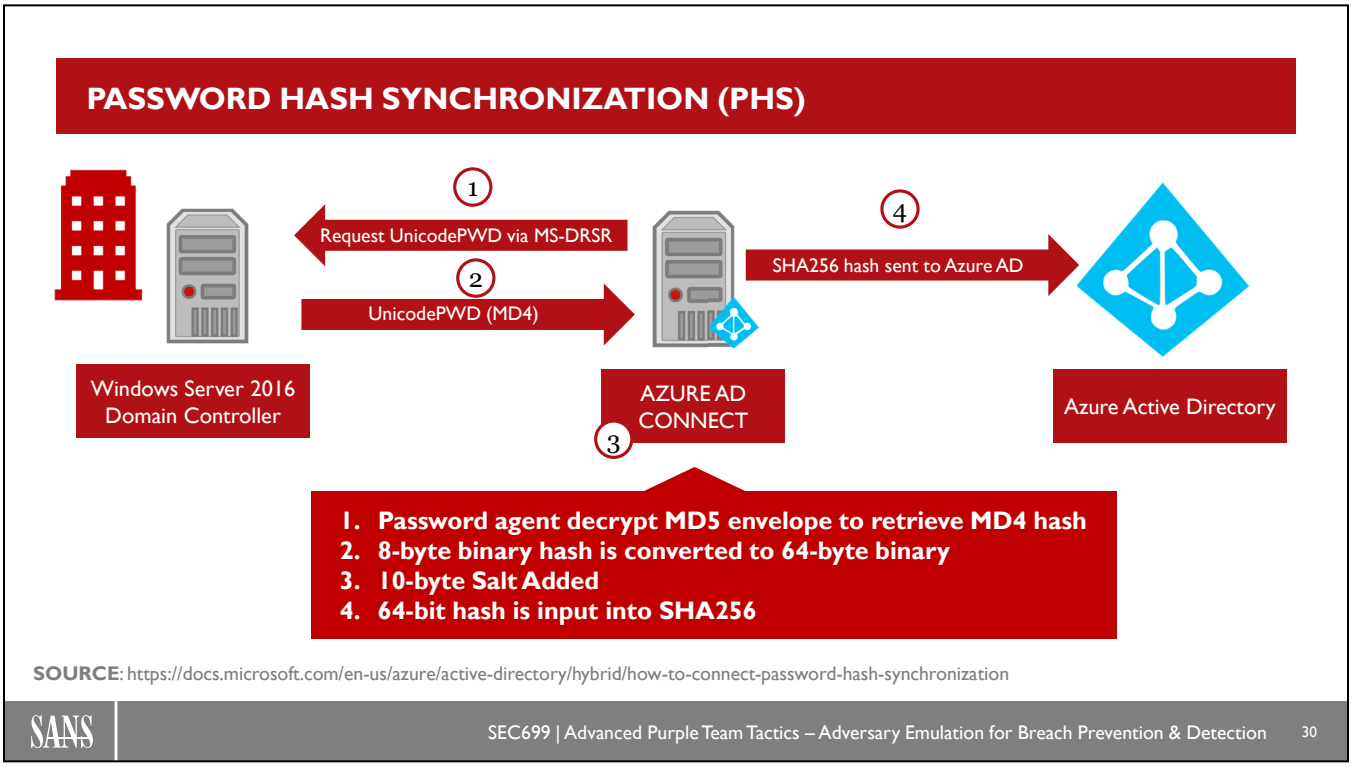
- Password hash synchronization – A sign-in method that synchronizes a hash of a user's on-premises AD password with Azure AD. In fact, a new, salted hash is generated from the on-premise hash.
- Pass-through authentication – A sign-in method that allows users to use the same password on-premises and in the cloud but doesn't require the additional infrastructure of a federated environment.
- Federation integration – Federation is an optional part of Azure AD Connect and can be used to configure a hybrid environment using an on-premises AD FS infrastructure. It also provides AD FS management capabilities such as certificate renewal and additional AD FS server deployments.

We will take a further look at these authentication methods in the next slides.

Additional features that need to be available via Azure AD Connect are:

- Synchronization – Responsible for creating users, groups, and other objects. As well as, making sure identity information for your on-premises users and groups is matching the cloud. This synchronization also includes password hashes.
- Health Monitoring – Azure AD Connect Health can provide robust monitoring.

Reference to these models and more information can be found via the official Microsoft Documentation: https://docs.microsoft.com/en-us/azure/active-directory/hybrid/whatis-azure-ad-connect



Password Hash Synchronization (PHS)

Password Hash synchronization doesn't simply synchronize the passwords from your local Active Directory toward your Azure AD tenant. Microsoft implemented several steps to protect your on-premise passwords. In case Azure AD would be breached and hashes were retrieved, it would still be impossible for an adversary to use these and target your on-premises Active directory accounts.

The following steps are described by Microsoft (source: https://docs.microsoft.com/en-us/azure/active-directory/hybrid/how-to-connect-password-hash-synchronization) and explain how hashes are synchronized in a hybrid model via Azure AD Connect:

1. Every two minutes, the password hash synchronization agent on the AD Connect server requests stored password hashes (the UnicodePwd attribute) from a DC. This request is via the standard MS-DRSR replication protocol used to synchronize data between DCs. The service account must have Replicate Directory Changes and Replicate Directory Changes All AD permissions (granted by default on installation) to obtain the password hashes.

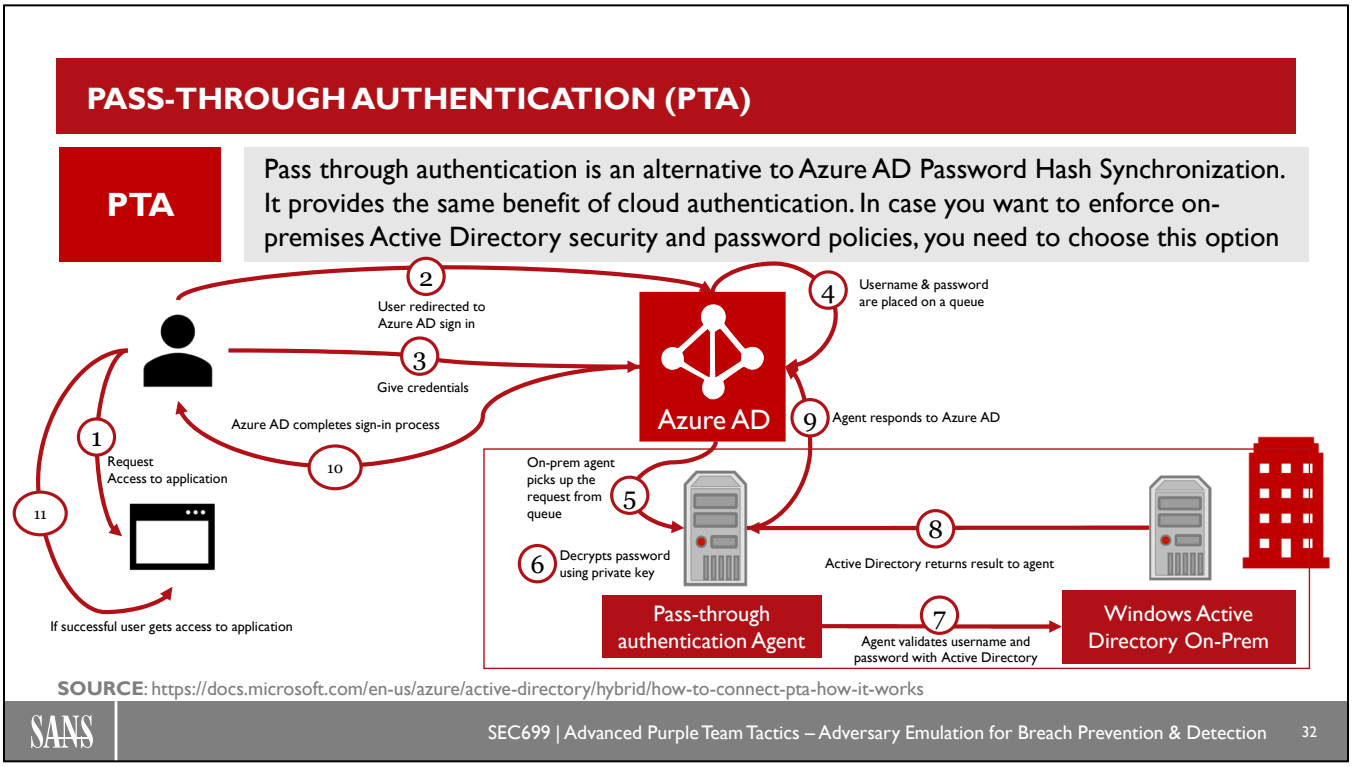
2. Before sending, the DC encrypts the MD4 password hash by using a key that is a MD5 hash of the RPC session key and a salt. It then sends the result to the password hash synchronization agent over RPC. The DC also passes the salt to the synchronization agent by using the DC replication protocol, so the agent will be able to decrypt the envelope.

3. After the password hash synchronization agent has the encrypted envelope, it uses MD5CryptoServiceProvider and the salt to generate a key to decrypt the received data back to its original MD4 format. The password hash synchronization agent never has access to the cleartext password. The password hash synchronization agent expands the 16-byte binary password hash to 64 bytes by first converting the hash to a 32-byte hexadecimal string, then converting this string back into

binary with UTF-16 encoding. The agent adds a per user salt, consisting of a 10-byte length salt, to the 64-byte binary to further protect the original hash. Finally, the sword hash synchronization agent then combines the MD4 hash plus the per user salt, and inputs it into the PBKDF2 function. One thousand iterations of the HMAC-SHA256 keyed hashing algorithm are used.

4. The password hash synchronization agent takes the resulting 32-byte hash, concatenates both the per user salt and the number of SHA256 iterations to it (for use by Azure AD), then transmits the string from Azure AD Connect to Azure AD over SSL.

When a user attempts to sign into Azure AD and enters their password, the password is run through the same MD4+salt+PBKDF2+HMAC-SHA256 process. If the resulting hash matches the hash stored in Azure AD, the user has entered the correct password and is authenticated.



Pass-Through Authentication (PTA)

Pass through authentication is an alternative to Azure AD Password Hash Synchronization. It provides the same benefit of cloud authentication. In case you want to enforce on-premises Active Directory security and password policies, you need to choose this option.

So how does it work? When a user tries to sign into an application secured by Azure AD, and if Pass-through Authentication is enabled on the tenant, the following steps occur (source: https://docs.microsoft.com/en-us/azure/active-directory/hybrid/how-to-connect-pta-how-it-works):

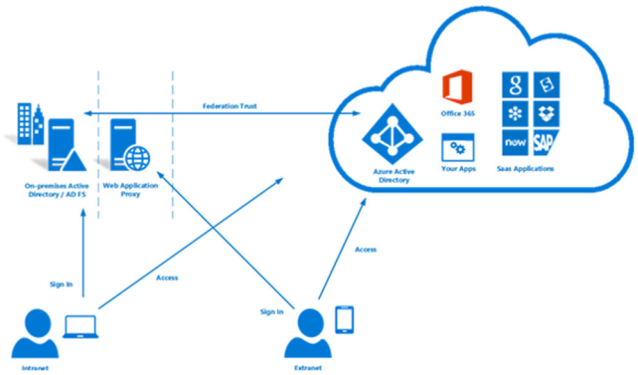
1. The user tries to access an application, for example, Outlook Web App.
2. If the user is not already signed in, the user is redirected to the Azure AD User Sign-in page.
3. The user enters their username into the Azure AD sign-in page, and then selects the Next button, the user enters their password into the Azure AD sign in page, and then selects the Sign-in button.
4. Azure AD, on receiving the request to sign in, places the username and password (encrypted by using the public key of the Authentication Agents) in a queue.
5. An on-premises Authentication Agent retrieves the username and encrypted password from the queue. Note that the Agent doesn't frequently poll for requests from the queue but retrieves requests over a pre-established persistent connection.
6. The agent decrypts the password by using its private key.
7. The agent validates the username and password against Active Directory by using standard Windows APIs, which is a similar mechanism to what Active Directory Federation Services (AD FS) uses. The username can be either the on-premises default username, usually userPrincipalName, or another attribute configured in Azure AD Connect (known as Alternate ID).

8. The on-premises Active Directory domain controller (DC) evaluates the request and returns the appropriate response (success, failure, password expired, or user locked out) to the agent.
9. The Authentication Agent, in turn, returns this response back to Azure AD.
10. Azure AD evaluates the response and responds to the user as appropriate. For example, Azure AD either signs the user in immediately or requests for Azure Multi-Factor Authentication.
11. If the user sign-in is successful, the user can access the application.

ACTIVE DIRECTORY FEDERATION SERVICES

Active Directory Federation Services (ADFS) provides a simplified way for administrators to allow users to leverage on-premise identities (credentials) to access cloud applications.

- As a prerequisite, a federated trust between Azure AD and on-premises environment is to be set up
- Federated authentication ensures that all user authentication occurs on-premise
- Federated authentication allows administrators to implement more tailored / fine-grained controls
- Federated authentication in ADFS is claims-based authentication, which removes authentication management from the application
- ADFS has a token-signing certificate for encrypted communication and authentication



SOURCE: https://docs.microsoft.com/en-us/azure/active-directory/hybrid/whatis-fed

Active Directory Federation Services

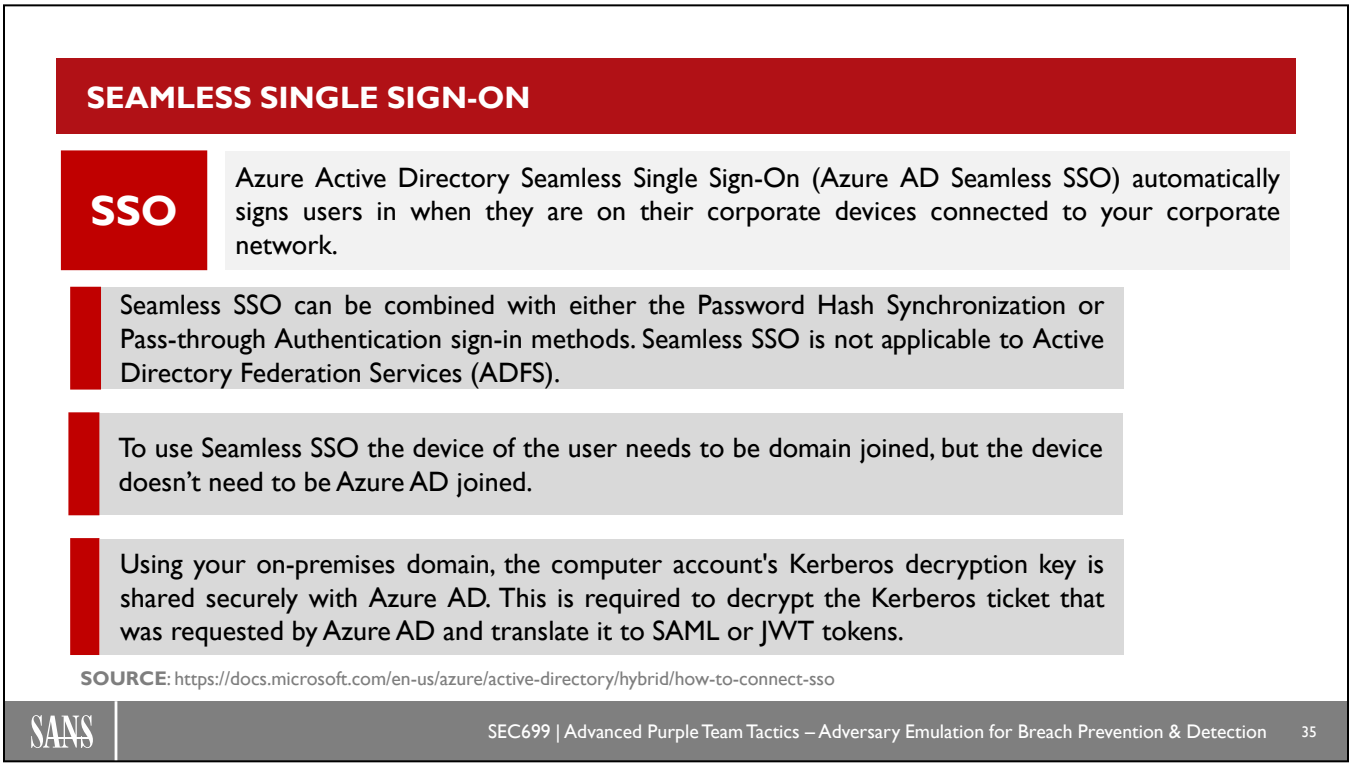
Active Directory Federation Services (ADFS) provides a simplified way for administrators to allow users to leverage on-premise identities (credentials) to access cloud applications.

In order to leverage ADFS, a federated trust between Azure AD and on-premises environment is to be set up.

Federated authentication within ADFS is claims-based authentication; this is another approach to authentication that removes authentication management from the application and makes it easier for you to manage accounts by centralizing authentication. Within a claim, you can find specific information that describes the given identity. Multiple claims can be requested. These are held in a token that also has a signature, so you know that the token is not tampered with in transit. When a user tries to log in with the federation authentication method, the user will be redirected toward the ADFS to perform a sign in.

ADFS trusts are based on private public key pairs (PKI Infrastructure), so you can't trust / set up a federation in case you have no known verified domain. Federated authentication will never work with the default Microsoft domain .onmicrosoft.com domain. Azure AD Connect can also create a trust to allow federated authentication.

Full Microsoft documentation can be found here:
https://docs.microsoft.com/en-us/azure/active-directory/hybrid/whatis-fed



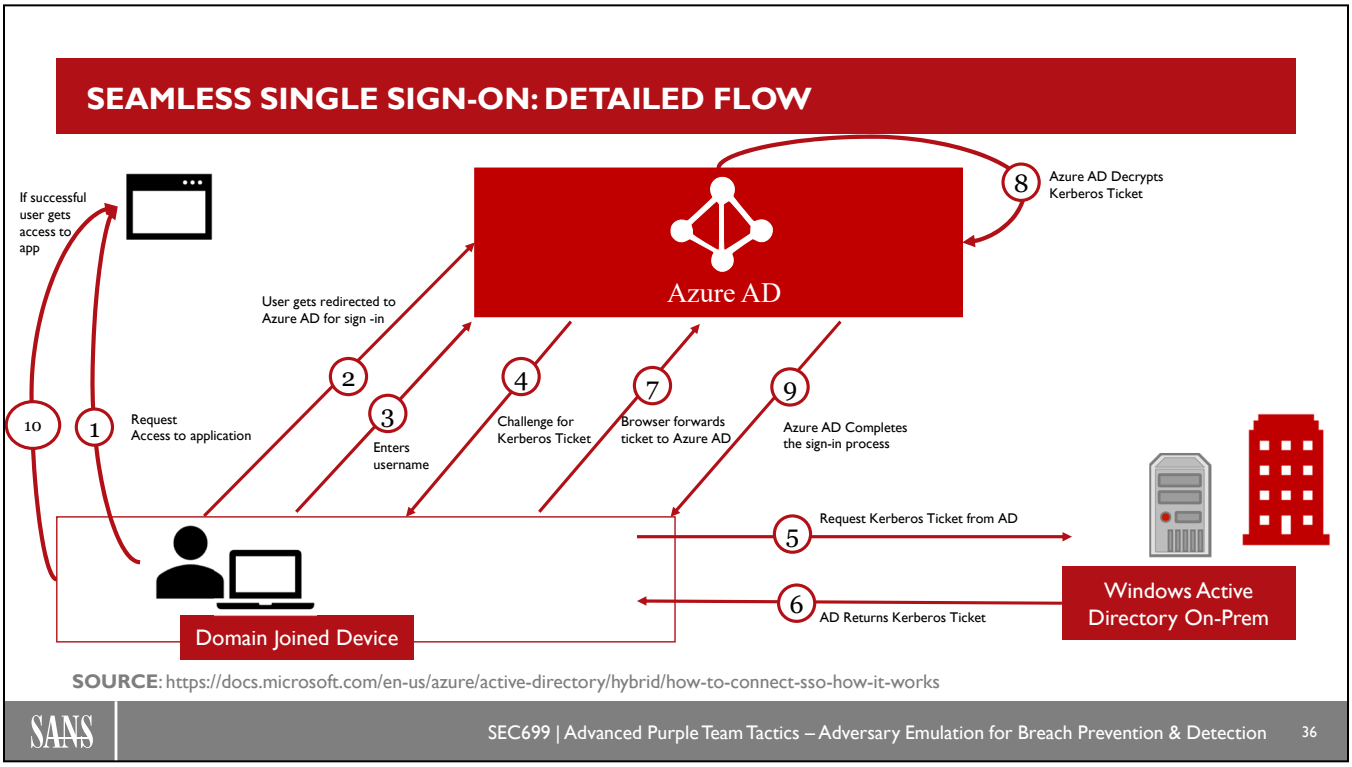
Seamless Single Sign-On

Azure Active Directory Seamless Single Sign-On (Azure AD Seamless SSO) automatically signs users in when they are on their corporate devices connected to your corporate network. Single Sign-on Azure AD uses Kerberos to identify the user on that corporate device; we will explain the single-sign-on flow more in depth. Note that Single Sign-on is not compatible with Active Directory Federation Services (ADFS) as the authentication is redirected toward your on-premise identity provider (domain controller).

Key highlights for Single Sign-On:

- Seamless SSO can be combined with either the Password Hash Synchronization or Pass-through Authentication sign-in methods. Seamless SSO is not applicable to Active Directory Federation Services (ADFS).
- To use Seamless SSO, the device of the user needs to be domain joined, but the device doesn't need to be Azure AD joined.
- Using your on-premises domain, the computer account's Kerberos decryption key is shared securely with Azure AD. This is required to decrypt the Kerberos ticket that was requested by Azure AD.

More information about this concept can be found here:
https://docs.microsoft.com/en-us/azure/active-directory/hybrid/how-to-connect-sso



Seamless Single Sign-On: Detailed Flow

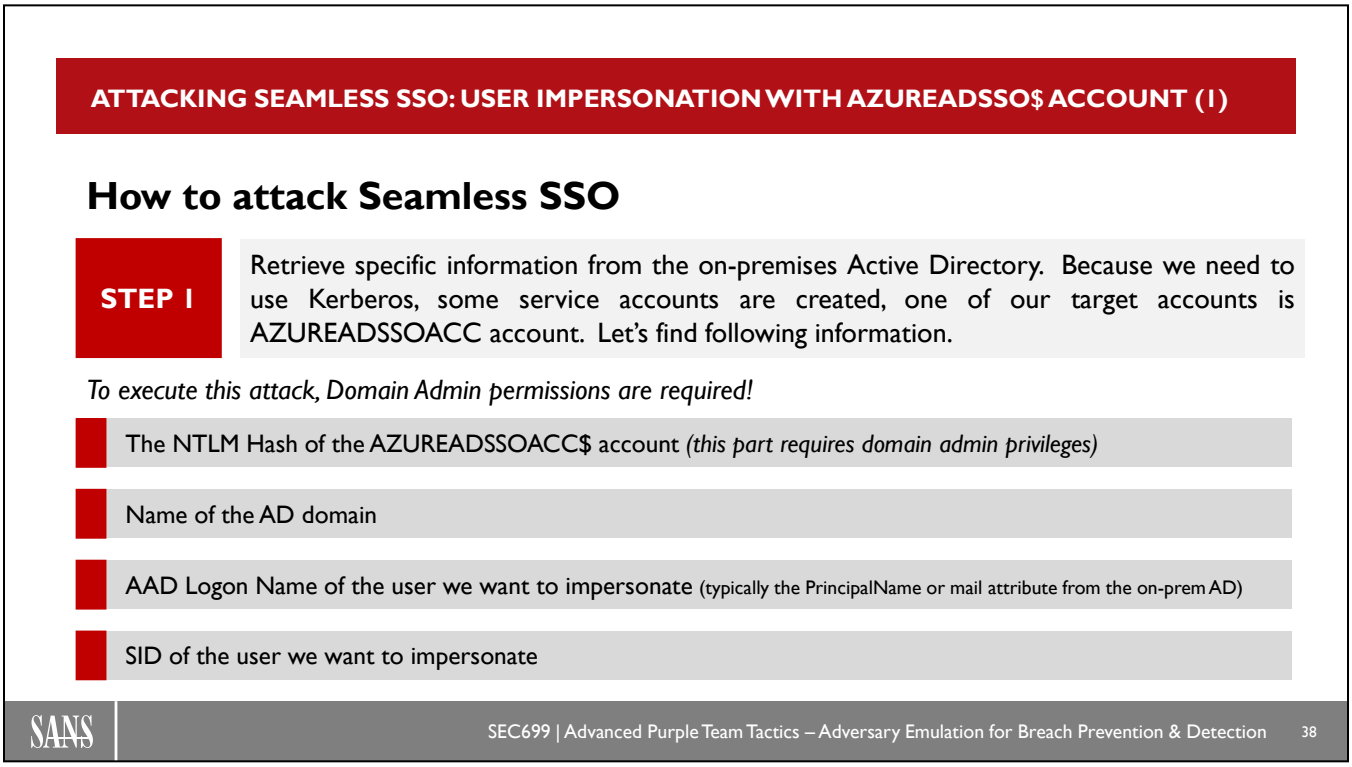
So how does Seamless Single Sign-on work with Windows Active Directory? As usual, Microsoft has provided detailed instructions on how Seamless Single Sign-On works in their online documentation. The following steps were taken from https://docs.microsoft.com/en-us/azure/active-directory/hybrid/how-to-connect-sso-how-it-works

The following steps describe this process in detail; also note that Azure AD is requesting tickets. In case you can generate your own Kerberos tickets, this would also allow a bad actor to grant access toward applications that are exposed via Azure AD.

1. The user tries to access a native application (for example, the Outlook client) from a domain-joined corporate device inside your corporate network.
2. If the user is not already signed in, the native application retrieves the username of the user from the device's Windows session.
3. The app sends the username to Azure AD and retrieves your tenant's WS-Trust MEX endpoint. This WS-Trust endpoint is used exclusively by the Seamless SSO feature, and is not a general implementation of the WS-Trust protocol on Azure AD.
4. The app then queries the WS-Trust MEX endpoint to see if integrated authentication endpoint is available. The integrated authentication endpoint is used exclusively by the Seamless SSO feature.
5. If step 4 succeeds, a Kerberos challenge is issued.
6. If the app is able to retrieve the Kerberos ticket, it forwards it up to Azure AD's integrated authentication endpoint.
7. Azure AD decrypts the Kerberos ticket and validates it.

8. Azure AD signs the user in, and issues a SAML token to the app.
9. The app then submits the SAML token to Azure AD's OAuth2 token endpoint.
10. Azure AD validates the SAML token, and issues to the app an access token and a refresh token for the specified resource, and an id token.
11. The user gets access to the app's resource.

Seamless SSO is opportunistic, which means if it fails, the sign-in experience falls back to its regular behavior—meaning the user needs to enter their password to sign in.



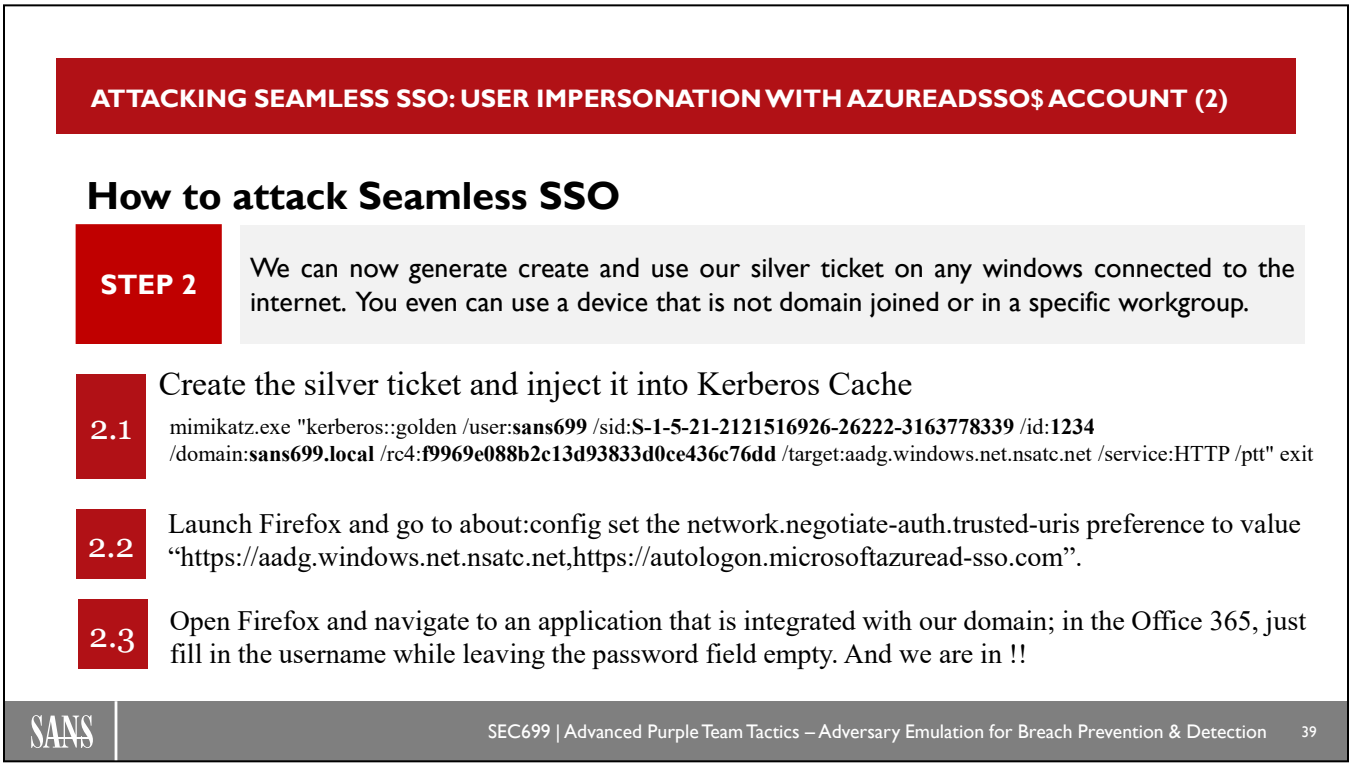
Attacking Seamless SSO: User Impersonation with AZUREADSSO$ Account (1)

An interesting attack technique against Seamless SSO was described by Michael Grafnetter in January 2017. In his blogpost "Impersonating Office 365 Users With Mimikatz" (https://www.dsinternals.com/en/impersonating-office-365-users-mimikatz/) he talks about abusing silver tickets from a specific account related to Azure AD to gain unauthorized access to Office 365 services. As SSO uses Kerberos, typical Kerberos attack strategies are useful here as well….

In his blog post, Michael explained that we can divide the attack in 2 main parts. In the first part, we need to obtain the following information:

1. NTLM password hash of the AZUREADSSOACC account
2. Name of the AD domain
3. AAD Logon Name of the user we want to impersonate (typically the PrincipalName or mail attribute from the on-prem AD)
4. SID of the user we want to impersonate

In case we collected this information, we can move on to step 2 and generate our silver tickets!



Attacking Seamless SSO: User impersonation with AZUREADSSO$ Account (2)

The second part of the attack goes as follows:

- Generate a silver ticket using mimikatz and providing the information we gathered previously. You will need to replace the /user, /sid /id / domain and /rc4 value with the previously obtained values:

  mimikatz.exe "kerberos::golden /user:sans699 /sid:S-1-5-21-2121516926-26222-3163778339 /id:1234

  /domain:sans699.local /rc4:f9969e088b2c13d93833d0ce436c76dd /target:aadg.windows.net.nsatc.net /service:HTTP /ptt" exit

- Once the ticket is loaded in memory, we can validate that it works using the following steps:
  - Launch Mozilla Firefox.
  - Go to about:config and set the network.negotiate-auth.trusted-uris preference to value "https://aadg.windows.net.nsatc.net,https://autologon.microsoftazuread-sso.com".
  - Navigate to any web application that is integrated with our AAD domain (e.g., Office365).
  - Once at the logon screen, fill in the username, while leaving the password field empty. Then press TAB or ENTER.

Course Roadmap

- Introduction & Key Tools
- Initial Access
- Lateral Movement
- Persistence
- **Azure AD & Emulation Plans**
- Adversary Emulation Capstone

SEC699.5

Azure AD
Azure AD Structure and Management
Azure AD Hybrid Authentication
Azure AD Authentication Methods
Azure AD Conditional Access
Introduction to Azure Identities
Azure AD Security Logging

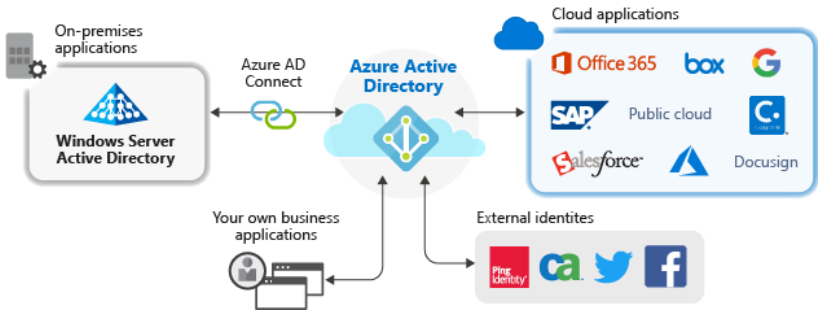
Executing emulation plans
APT-28 Emulation Plan
Exercise: APT-28 Emulation Plan
APT-34 Emulation Plan
Exercise: APT-34 Emulation Plan
Turla Emulation Plan
Exercise: Turla Emulation plan

This page intentionally left blank.

AZURE AD: ENTERPRISE APPLICATIONS

- Azure AD Gallery applications
- On-premises applications with Application Proxy
- Custom-developed applications
- Non-Gallery applications

Azure AD supports multiple enterprise applications to use a cloud-based identity server and to manage access toward all these applications



SOURCE: https://docs.microsoft.com/en-us/azure/active-directory/manage-apps/what-is-application-management

Azure AD: Enterprise Applications

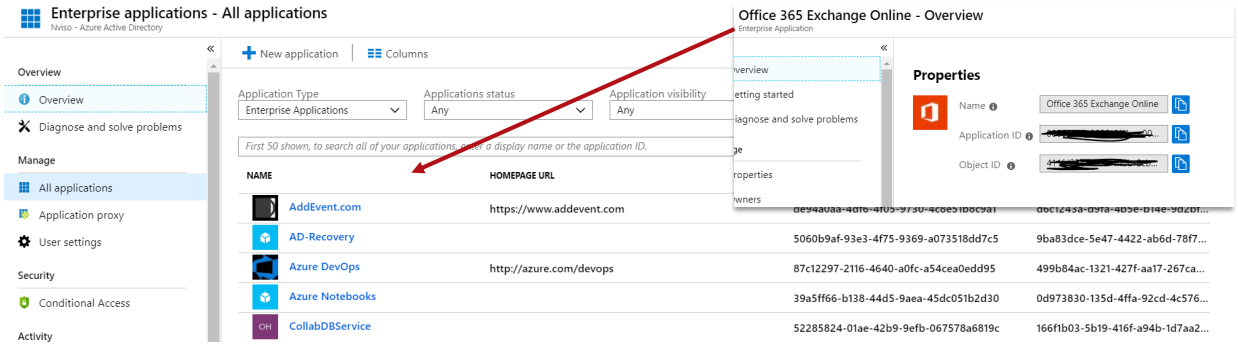
One of the core benefits of Azure AD is its use as a central authentication platform by a variety of enterprise applications. The following types of applications are supported with Azure Active Directory:

- Azure AD Gallery applications – Azure AD has a gallery that contains thousands of applications that have been pre-integrated for single sign-on with Azure AD. Some of the applications your organization uses are probably in the gallery.
- On-premises applications with Application Proxy – With Azure AD Application Proxy, you can integrate your on-premises web apps with Azure AD to support single sign-on. In such a setup, end users can access your on-premises web apps in the same way they access Office 365 and other SaaS apps.
- Custom-developed applications – When building your own line-of-business applications, you can integrate them with Azure AD to support single sign-on. By registering your application with Azure AD, you have control over the authentication policy for the application.
- Non-Gallery applications – Bring your own applications! Support single sign-on for other apps by adding them to Azure AD. You can integrate any web link you want, or any application that renders a username and password field, supports SAML or OpenID Connect protocols, or supports SCIM.

You can find tutorials, documentation, and concepts on the official Microsoft documentation page: https://docs.microsoft.com/en-us/azure/active-directory/manage-apps/

AZURE AD: ENTERPRISE APPLICATIONS – REGISTERING AN APP

All enterprise applications require registration; you can find the enterprise application page within the Azure AD portal. Each application has a unique object ID and application ID; Office 365 exchange online is also a registered application within Azure AD.

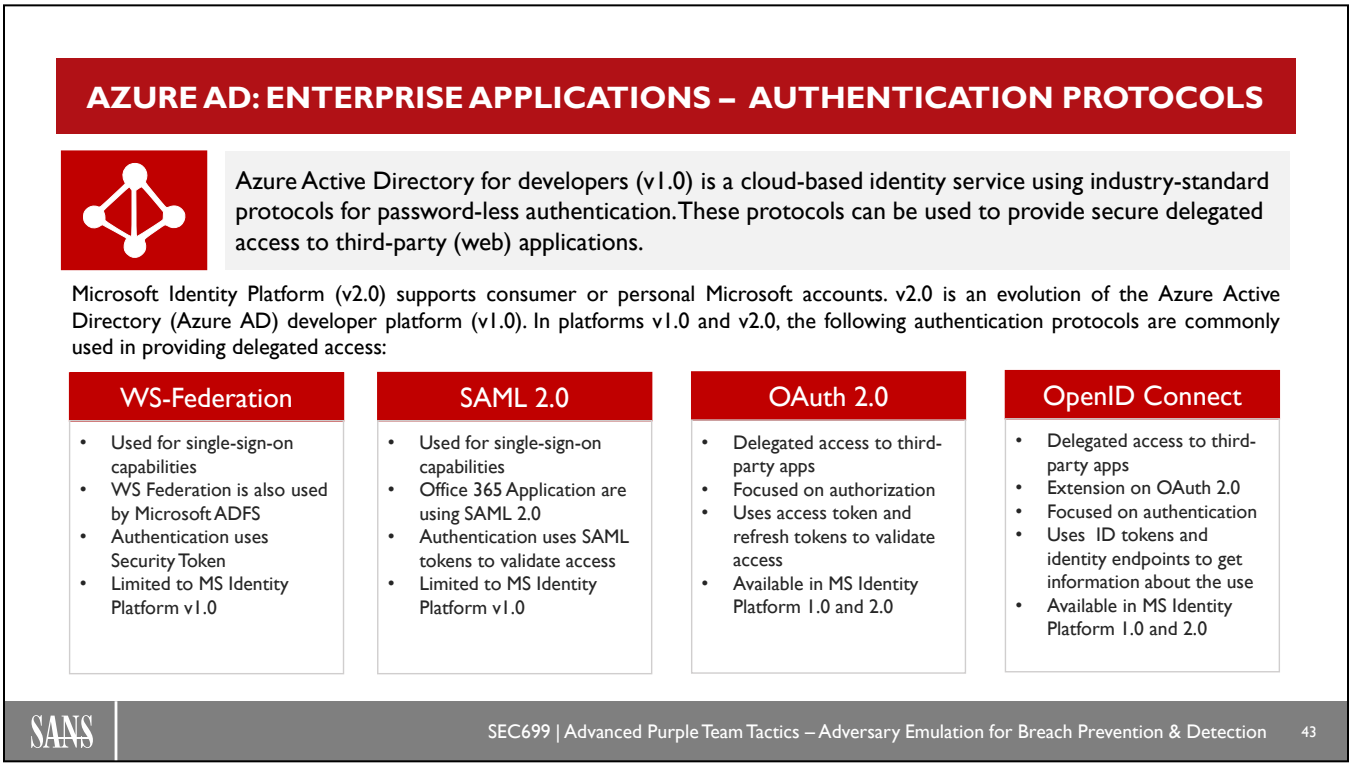


Azure AD: Enterprise Applications – Registering an App

To provide secure sign-in or single-sign-on capabilities to a specific application, you are required to register that application within Azure AD. This means you need to give information about the application and how it is going to use Azure Identity platform within Azure AD as identity service provider. The information required includes:

1. Name – Enter a meaningful application name that will be displayed to users of the app.
2. Supported account types – Select which accounts you would like your application to support. This can be accounts in this organizational directory only, accounts in any organizational directory or accounts in any organizational directory and personal Microsoft accounts.
3. Redirect URI (optional) – Select the type of app you're building, Web or Public client (mobile & desktop), and then enter the redirect URI (or reply URL) for your application. In case a secure transfer needs to be executed, Microsoft will use this URL to forward specific information to your application.
4. Once this is done, your app is registered; however, in case you need to request information from Azure AD, you need to define the Scope and permissions for that app. For example, you want to retrieve the groups that are allocated to a specific user, then it is required to add this permission to the scope of your app. The scope is typically combined with the OAuth2.0 protocol to validate authorization.

You can use the Microsoft Graph API to request specific information; all details can be found here: https://developer.microsoft.com/en-us/graph



Azure AD: Enterprise Applications – Authentication Protocols

Azure Active Directory for developers (v1.0) is a cloud-based identity service using industry-standard protocols for password-less authentication. These protocols can be used to provide secure delegated access to third-party (web) applications. Microsoft Identity Platform (v2.0) supports consumer or personal Microsoft accounts. v2.0 is an evolution of the Azure Active Directory (Azure AD) developer platform (v1.0). In platforms v1.0 and v2.0, the following authentication protocols are commonly used in providing delegated access:

WS-Federation

- Used for single-sign-on capabilities
- WS Federation is also used by Microsoft ADFS
- Authentication uses Security Token
- Limited to MS Identity platform v1.0

SAML2.0

- Used for single-sign-on capabilities
- Office 365 Application are using SAML 2.0
- Authentication uses SAML tokens to validate access
- Limited to MS Identity platform v1.0

OAuth 2.0

- Delegated access to third-party apps
- Focused on authorization (not authentication)
- Uses access token and refresh tokens to validate access
- Available in MS Identity platform 1.0 and 2.0

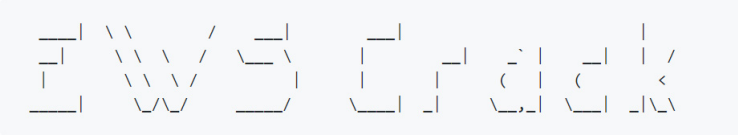
OpenID Connect

- Delegated access to third-party apps
- Extension on OAuth 2.0
- Focused on authentication (not authorization)
- Uses ID tokens and identity endpoints to get information about the use
- Available in MS Identity platform 1.0 and 2.0

In the next slides, we will go more in-depth and see how these protocols work.

ANOTHER INTERESTING TOOL: EWS CRACKER – BYPASSING MFA

EWS Cracker



Developed by Mike Siegel
https://github.com/mikesiegel/ews-crack

Example: python ews-crack.py --mode spray --filename users.txt --domain contoso.com --password <CommonPassword>

EWS stands for Exchange Web Services. This is a SOAP-based protocol used for free/busy scheduling, which is often leveraged by third-party mail clients. It allows a user to read email, send email, and test credentials.

Unfortunately, EWS only supports Basic Authentication. If you have multi-factor authentication through a third-party provider, such as Ping, Duo or Okta, EWS can be used to bypass MFA. It can also be used to bypass MDM solutions.

Another Interesting Tool: EWS Cracker – Bypassing MFA

EWS stands for Exchange Web Services. This is a SOAP-based protocol used for free/busy scheduling, which is often leveraged by third-party mail clients. It allows a user to read email, send email, and test credentials. Unfortunately, EWS only supports Basic Authentication. If you have multi-factor authentication through a third-party provider, such as Ping, Duo or Okta, EWS can be used to bypass MFA. It can also be used to bypass MDM solutions.

EWS Crack is a python script that can be used to attack AWS. The full source code can be found on: https://github.com/mikesiegel/ews-crack.

Furthermore, an interesting article on its typical use can be found here: https://www.reddit.com/r/netsec/comments/7qs6gb/ewscrack_bypass_2fa_and_mdm_in_office365_using_ews/

Course Roadmap

- Introduction & Key Tools
- Initial Access
- Lateral Movement
- Persistence
- **Azure AD & Emulation Plans**
- Adversary Emulation Capstone

SEC699.5

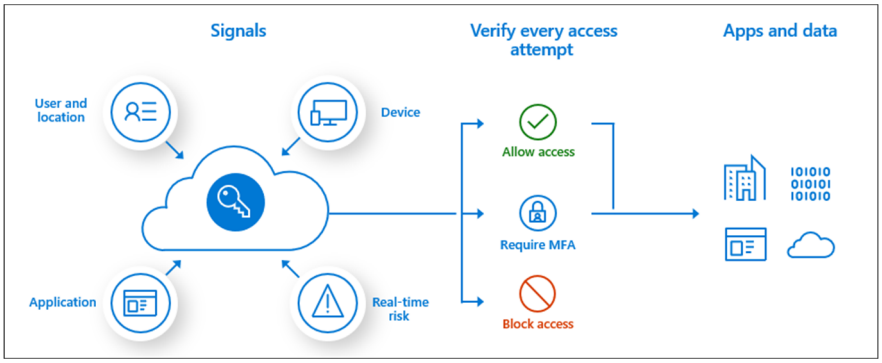
Azure AD
Azure AD Structure and Management
Azure AD Hybrid Authentication
Azure AD Authentication Methods
Azure AD Conditional Access
Introduction to Azure Identities
Azure AD Security Logging

Executing emulation plans
APT-28 Emulation Plan
Exercise: APT-28 Emulation Plan
APT-34 Emulation Plan
Exercise: APT-34 Emulation Plan
Turla Emulation Plan
Exercise: Turla Emulation plan

This page intentionally left blank.

INTRODUCING AZURE AD CONDITIONAL ACCESS

Conditional Access is the tool used by Azure Active Directory to bring signals together, to make decisions, and enforce organizational policies. You can use identity signals as part of your access control decisions.



- Signals
- Decisions
- Enforcement

SOURCE: https://docs.microsoft.com/en-us/azure/active-directory/conditional-access/overview

Introducing Azure AD Conditional Access

Conditional Access is the tool used by Azure Active Directory to bring signals together, to make decisions, and enforce organizational policies. You can use identity signals as part of your access control decisions. As a first example, let's investigate conditional access to protect our Azure AD identities. Conditional access can be activated in Azure AD in case you have a premium license P1 or P2.

Conditional Access policies at their simplest are if-then statements: If a user wants to access a resource, then they must complete an action. Example: A helpdesk manager wants to access the IT service desk application and is required to perform multi-factor authentication to access it. Conditional access policies are using signals to validate certain settings, decisions to actually allow access to an application or enforce other security controls, and these conditional access policies can be enforced in different applications. (e.g., SharePoint Online or on-prem applications).

What are typical signals within Conditional Access:

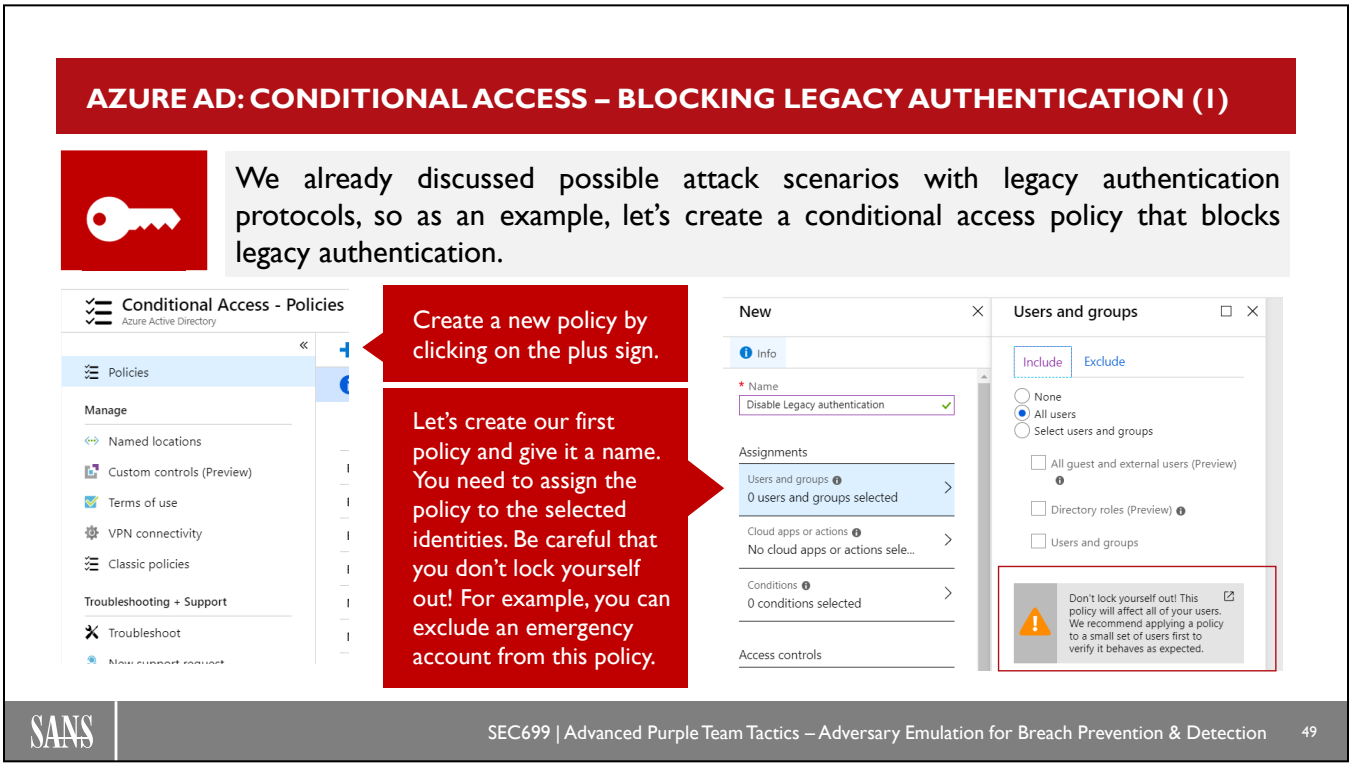
- User or group membership: For example, is the user a member of a priviliged group
- IP Location information: Can be based on specific Trusted IPs or IP ranges
- Device state and compliance: Devices of specific platforms or marked with a specific state can be used when enforcing Conditional Access. In most cases, the compliance state is checked with Intune which is the MDM service from Microsoft.
- Application details: Users attempting to access specific applications can trigger different Conditional Access policies.

- Real-time and calculated risk detection: Signals integration with Azure AD Identity Protection allows Conditional Access policies to identify risky sign-ins
- Microsoft Cloud App Security (MCAS): Enables user application access and sessions to be monitored and controlled in real time

What are common decisions with Conditional access:

- Block access
- Grant access (However additional controls can be requested):
    - Require multi-factor authentication
    - Require device to be marked as compliant (State is validated by Intune)
    - Require Hybrid Azure AD joined device
    - Require approved client app

Microsoft reference documentation:
https://docs.microsoft.com/en-us/azure/active-directory/conditional-access/overview

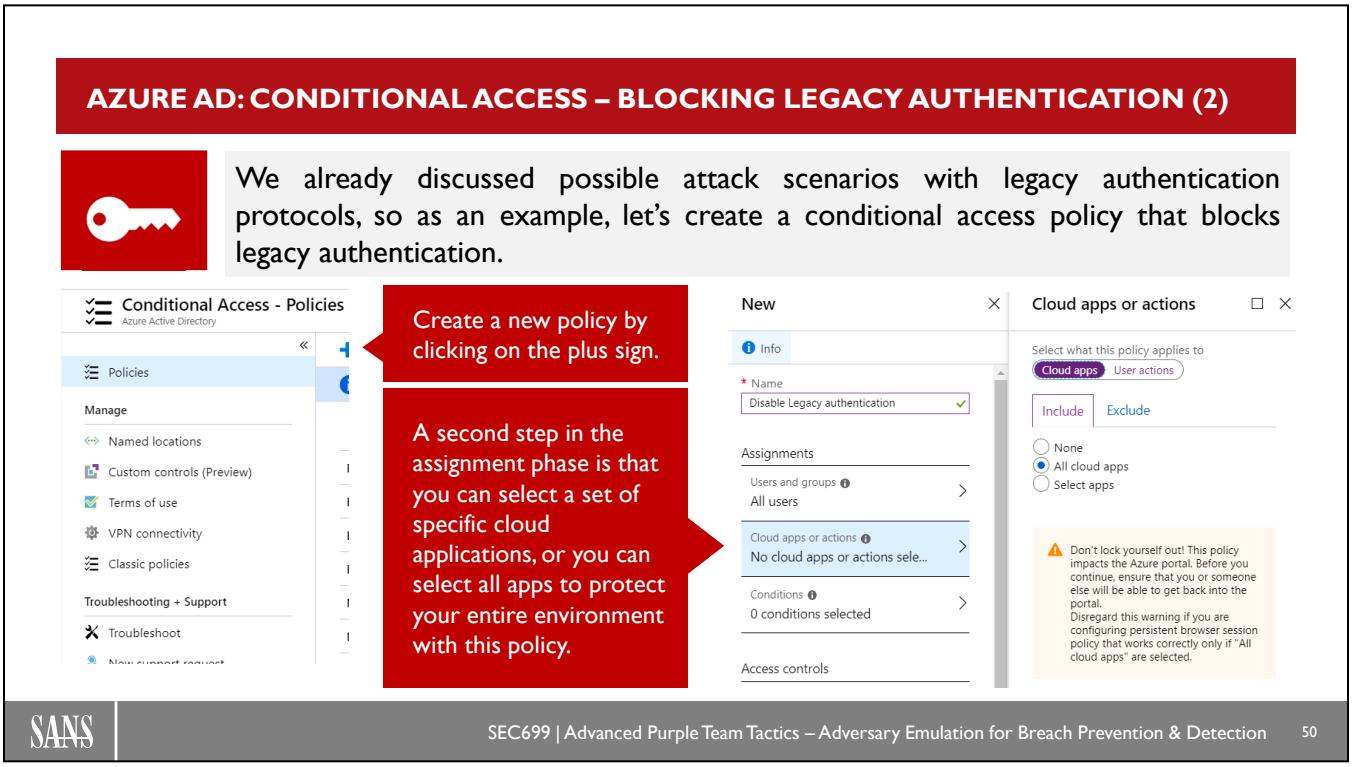


Azure AD: Conditional Access – Blocking Legacy Authentication (1)

We already discussed possible attack scenarios with legacy authentication protocols, so as an example, let's create a conditional access policy that blocks legacy authentication. In order to do so, please complete the following steps:

- Sign into the Azure portal as a global administrator, security administrator, or Conditional Access administrator;
- Browse to "Azure Active Directory" > "Conditional Access";
- Select "New policy";
- Give your policy a meaningful name;
- Under Assignments, select Users and groups
  - Under Include, select All users.
  - Under Exclude, select Users and groups and choose any accounts that must maintain the ability to use legacy authentication.
  - Select Done.

Additional documentation can be found here:
https://docs.microsoft.com/en-us/azure/active-directory/conditional-access/block-legacy-authentication



Azure AD: Conditional Access – Blocking Legacy Authentication (2)

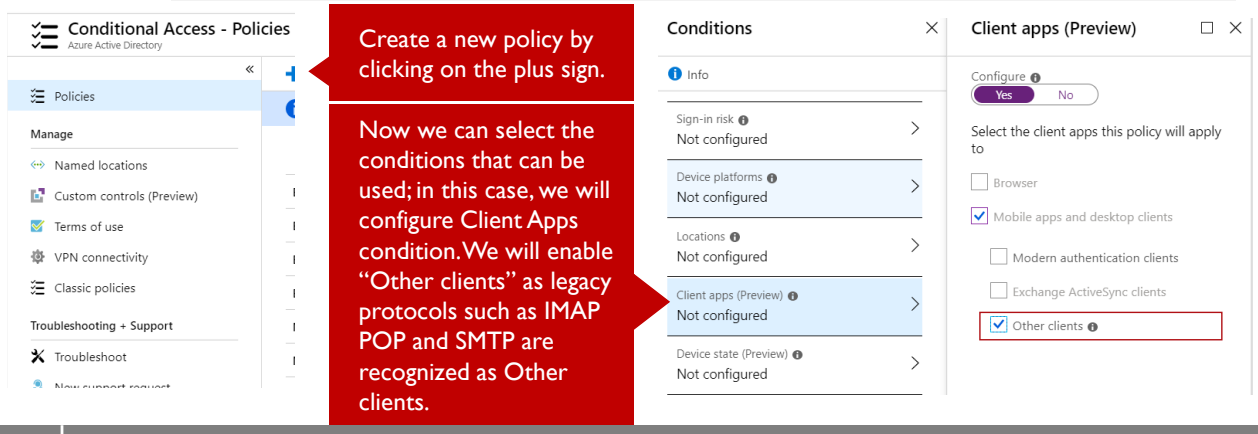
You may have already noticed that, in many Azure AD features, fine-grained configuration controls are available. This is no different for Conditional Access. Next to specifying to what users it applies, we can also specify what applications are affected by the block on legacy authentication protocols.

This can be done using the following steps:

- Under "Cloud apps or actions" > "Include", select "All cloud apps".
- If you must exclude specific applications from your policy, you can choose them from the "Exclude" tab under "elect excluded cloud apps" and choose "Select".
- Select "Done".

AZURE AD: CONDITIONAL ACCESS – BLOCKING LEGACY AUTHENTICATION (3)

We already discussed possible attack scenarios with legacy authentication protocols, so as an example, let's create a conditional access policy that blocks legacy authentication.

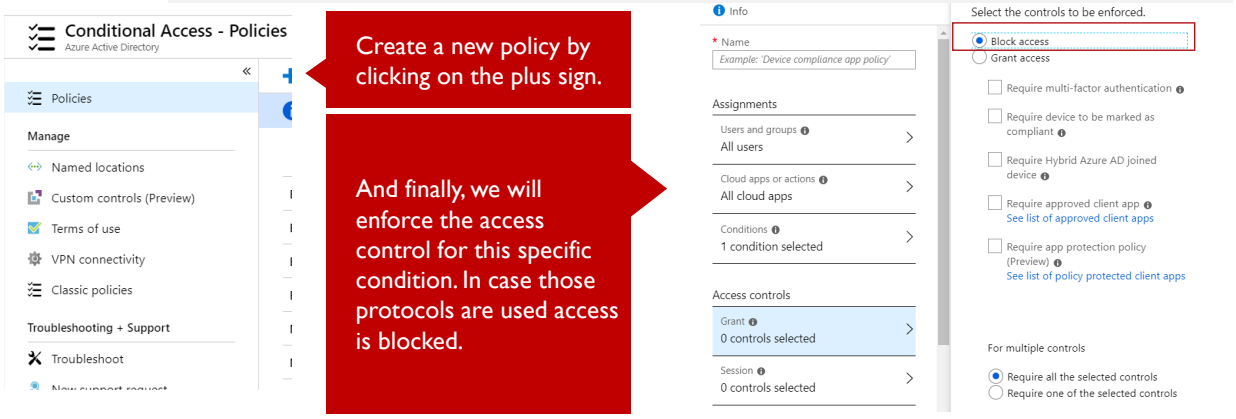


Azure AD: Conditional Access – Blocking Legacy Authentication (3)

Now that we have configured the scope of the conditional access policy (both users an applications were configured), we will decide what we actually want to block using the "Conditions" menu. Now we can select the conditions that can be used; in this case, we will configure Client Apps condition. We will enable "Other clients" as legacy protocols such as IMAP POP and SMTP are recognized as Other clients.

AZURE AD: CONDITIONAL ACCESS – BLOCKING LEGACY AUTHENTICATION (4)

We already discussed possible attack scenarios with legacy authentication protocols, so as an example, let's create a conditional access policy that blocks legacy authentication.

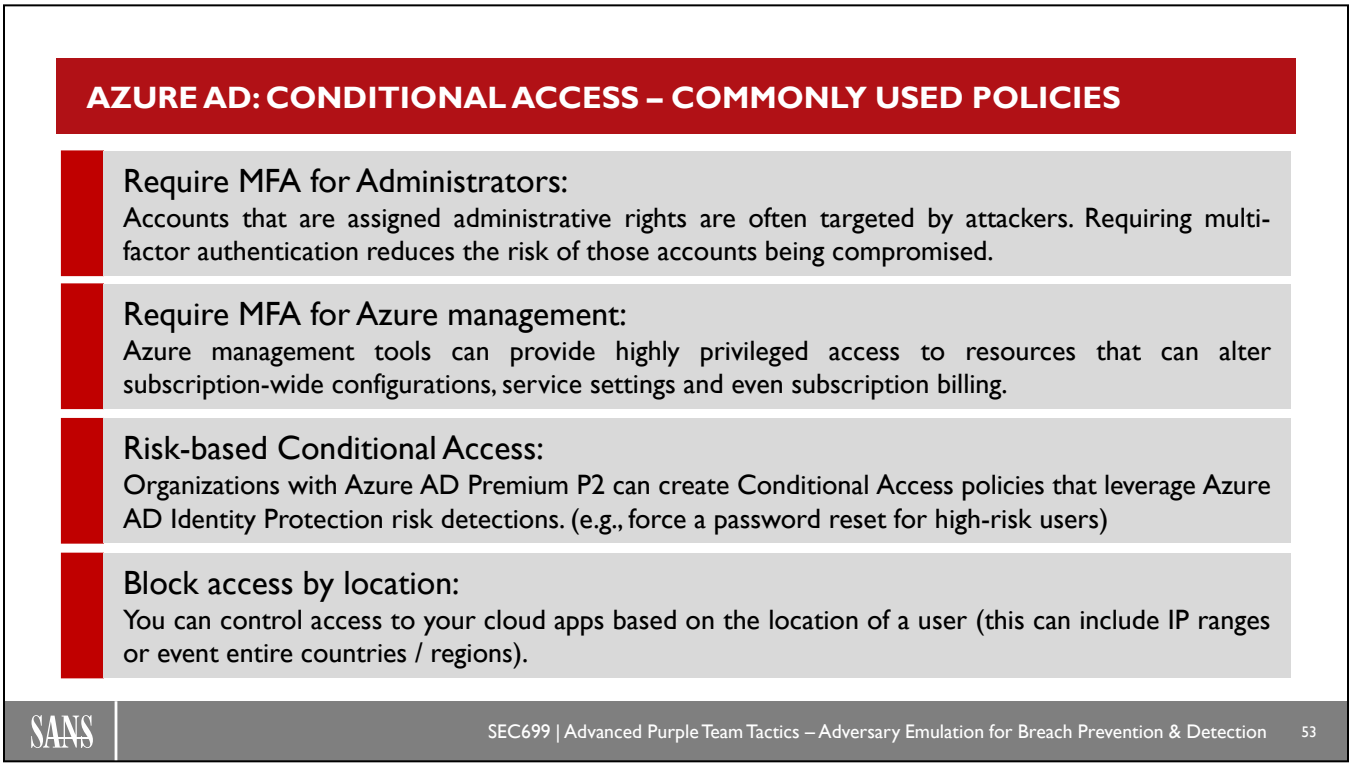


Azure AD: Conditional Access – Blocking Legacy Authentication (4)

Finally, we need to tell Azure AD that any access meeting this condition should be blocked. We can do this as follows:

- Under "Access controls" > "Grant", select "Block access".
  - Select "Select".
- Confirm your settings and set "Enable policy" to "On".
- Select "Create" to create and immediately enable your policy.

That's it! We have now created our first conditional access policy that will block access to your environment coming from legacy protocols.



Azure AD: Conditional Access – Commonly Used Policies

Some of the most commonly used conditional access policies include:

Require MFA for Administrators

Accounts that are assigned administrative rights are often targeted by attackers. Requiring multi-factor authentication reduces the risk of those accounts being compromised.

Require MFA for Azure management

Azure management tools can provide highly privileged access to resources that can alter subscription-wide configurations, service settings and even subscription billing.

Risk-based Conditional Access

Organizations with Azure AD Premium P2 can create Conditional Access policies that leverage Azure AD Identity Protection risk detections. (e.g., force a password reset for high-risk users)

Block access by location

You can control access to your cloud apps based on the location of a user (this can include IP ranges or event entire countries / regions).

Note that this list is limited, and many other policies can be created to further secure you identity service (e.g., critical applications can only be accessed by domain joined devices).

Course Roadmap

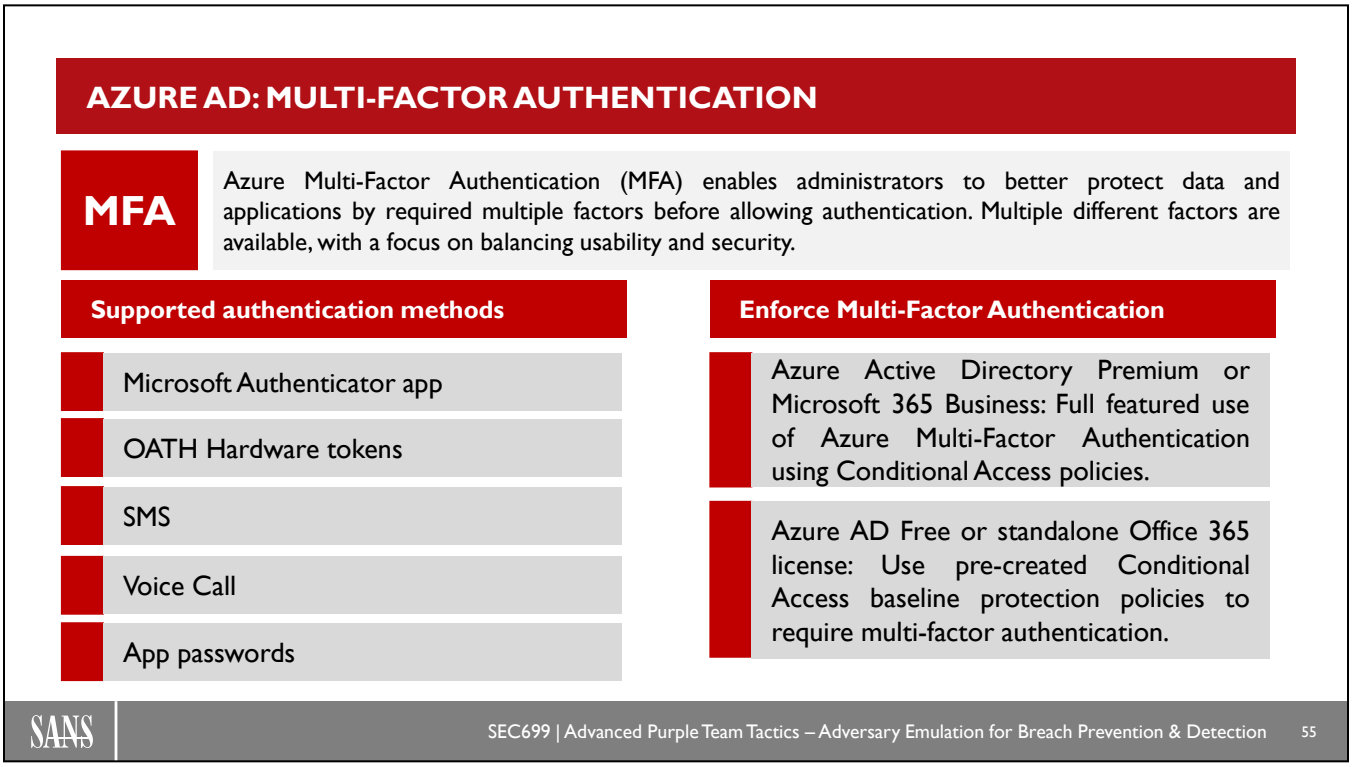
- Introduction & Key Tools
- Initial Access
- Lateral Movement
- Persistence
- **Azure AD & Emulation Plans**
- Adversary Emulation Capstone

SEC699.5

Azure AD
Azure AD Structure and Management
Azure AD Hybrid Authentication
Azure AD Authentication Methods
Azure AD Conditional Access
Introduction to Azure Identities
Azure AD Security Logging

Executing emulation plans
APT-28 Emulation Plan
Exercise: APT-28 Emulation Plan
APT-34 Emulation Plan
Exercise: APT-34 Emulation Plan
Turla Emulation Plan
Exercise: Turla Emulation plan

This page intentionally left blank.



Azure AD: Multi-Factor Authentication

Azure Multi-Factor Authentication (MFA) enables administrators to better protect data and applications by required multiple factors before allowing authentication. Multiple different factors are available, with a focus on balancing usability and security.

What type of "factors" are available?

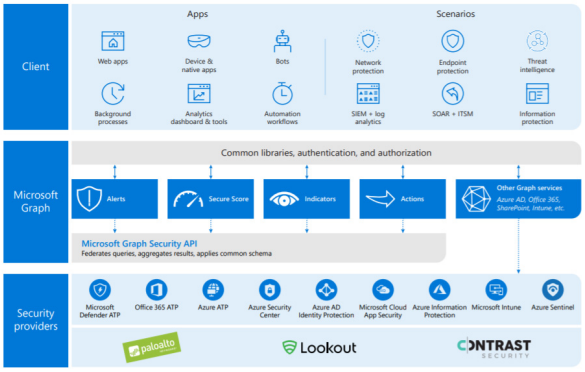
- Microsoft Authenticator app: The Microsoft Authenticator app provides an additional level of security to your Azure AD work or school account or your Microsoft account. The Microsoft Authenticator app is available for Android, iOS, and Windows Phone.
- OATH hardware tokens: OATH is an open standard that specifies how one-time password (OTP) codes are generated. Azure AD supports the use of OATH-TOTP SHA-1 tokens of the 30-second or 60-second variety.
- SMS Message: An SMS is sent to the mobile phone number that is linked to the identity in Azure AD containing a verification code. Enter the verification code provided in the sign-in interface to continue.
- Voice Call: An automated voice call is made to the phone number you provide. Answer the call and press # in the phone keypad to authenticate
- App Passwords: Certain non-browser apps do not support multi-factor authentication. If a user has been enabled for multi-factor authentication and attempts to use non-browser apps, they are unable to authenticate. An app password allows users to successfully authenticate. It's important to note, however, that when Multi-Factor Authentication is enforced through Conditional Access policies, you cannot create app passwords. You would thus need to enable MFA per user.

Additional information can be found via the Official Microsoft documentation page:
https://docs.microsoft.com/en-us/azure/active-directory/authentication/concept-mfa-howitworks

MICROSOFT GRAPH SECURITY API

API

Part of Microsoft Graph, the Microsoft Graph Security API integrates with security solutions from Microsoft and partners in a federated model. We will shortly introduce the basic concepts of the Microsoft Graph Security API.



Some Scenarios to use Microsoft Graph API:

- Streamline alert life-cycle management
- Automate creation of security incidents
- Use custom threat intelligence services
- Proactively manage security risks

Microsoft Graph Security API

As everything in larger cloud environments is based on REST API calls, Microsoft decided to create one centralized Graph Security API. Part of Microsoft Graph, the Microsoft Graph Security API integrates with security solutions from Microsoft and partners in a federated model. We will shortly introduce the basic concepts of the Microsoft Graph Security API.

How does Microsoft Graph Security API work?

The Graph API uses the same standards that were described earlier in this course: They share a common authentication framework based on OpenID Connect, OAuth 2.0, and a Web Representational State Transfer (REST) API with standard JavaScript Object Notation (JSON) response formats. In case you are going to use custom applications, you will need to register these applications to allow access to the Microsoft Graph API.

Let's discuss some common Scenarios published by Microsoft for using the Microsoft Graph API:

- Streamline alert life-cycle management for security monitoring
  An analyst signs into a security application integrated with the Microsoft Graph Security API. The analyst can now view high-severity security alerts across different security providers.
  Example REST call:

  GET https://graph.microsoft.com/v1.0/security/alerts?$filter=severity eq 'high'

- Automated creation of security incidents for security monitoring / incident response
  Integrate your existing ticketing system with security alerts that were triggered in Azure. In case a ticket is updated in your ticketing system, this information can be forwarded to the Microsoft Graph API using REST calls.

Example REST call:

PATCH https://graph.microsoft.com/v1.0/security/alerts/{alert_id}

Content-type: application/json

{ "assignedTo": "ErikVanBuggenhout", "vendorInformation": { "provider": "String", "vendor": "String" } }

- Use custom threat intelligence services
  A threat was discovered; you want to upload information on this threat to Azure. The threat indicator can be sent integrated \ using the Microsoft Graph Security API. The Microsoft solution can then alert the organization if the file is detected.
  Example REST call:

  POST beta/security/tiIndicators/

  Content-type: application/json

  *{ "action": "alert", "confidence": 0, "description": "MD5 hash on watch while system vulnerabilities being addressed", "expirationDateTime": "2019- 03-01T21:43:37.5031462+00:00", "externalId": "CUSTOM-MD5-Indicator", "fileHashType": "MD5", "fileHashValue": "**fe8a8226a4cfd0deffe209069a7e64d908b74de8**", "severity": 0, "targetProduct": "Azure Sentinel", "threatType": "WatchList", "tlpLevel": "green"}*

- Proactively manage security risks
  By querying the Microsoft Graph Security API secure score entity, the organization can quickly retrieve the most recent summarized Microsoft Secure Score. For example, a list can be generated to validate the security improvements that have a high impact on the end-users.
  Example REST call:

  GET /security/secureScoreControlProfiles?$filter=userImpact eq 'High'

Additional documentation on the Microsoft Graph Security API can be found here:
https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RWm9G4

AZURE ROLE BASED ACCESS CONTROL (RBAC)

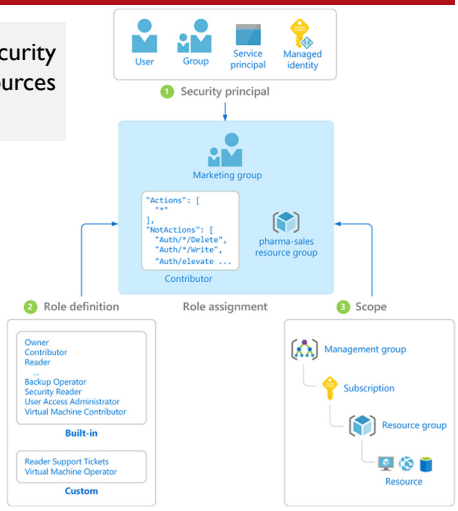
IAM

Access Control in Azure uses identities in Azure AD (security principal) to give permissions (role definition) on Azure resources (scope)

A security principal is an object in Azure AD that represents a user, group, service principal, or managed identity that is requesting access to Azure resources.

A role definition is a collection of permissions. It lists the operations that can be performed, such as read, write, and delete.

A scope can be assigned to a role assignment. The scope can be a subscription or a specific resource group within that subscription.



SOURCE: https://docs.microsoft.com/en-us/azure/azure-australia/role-privileged

Azure Role Based Access Control (RBAC)

Azure Role Based Access Control has a strong link with Azure AD. As within Azure, we can define several resources such as containers, virtual servers, databases, and many more. In many cases, we see that these resources are managed with a local or built-in account; this makes user management very complex and increases the risk of a compromise. Role based access will enable Azure AD as identity service for specific resources.

A security principal is an object in Azure AD that represents a user, group, service principal, or managed identity that is requesting access to Azure resources. Microsoft provides the following definitions for these concepts:

- User: An individual who has a profile in Azure Active Directory. You can also assign roles to users in other tenants.
- Group: A set of users created in Azure Active Directory. When you assign a role to a group, all users within that group have that role.
- Service principal: A security identity used by applications or services to access specific Azure resources. You can think of it as a user identity (username and password or certificate) for an application.
- Managed identity: An identity in Azure Active Directory that is automatically managed by Azure. You typically use managed identities when developing cloud applications to manage the credentials for authenticating to Azure services.

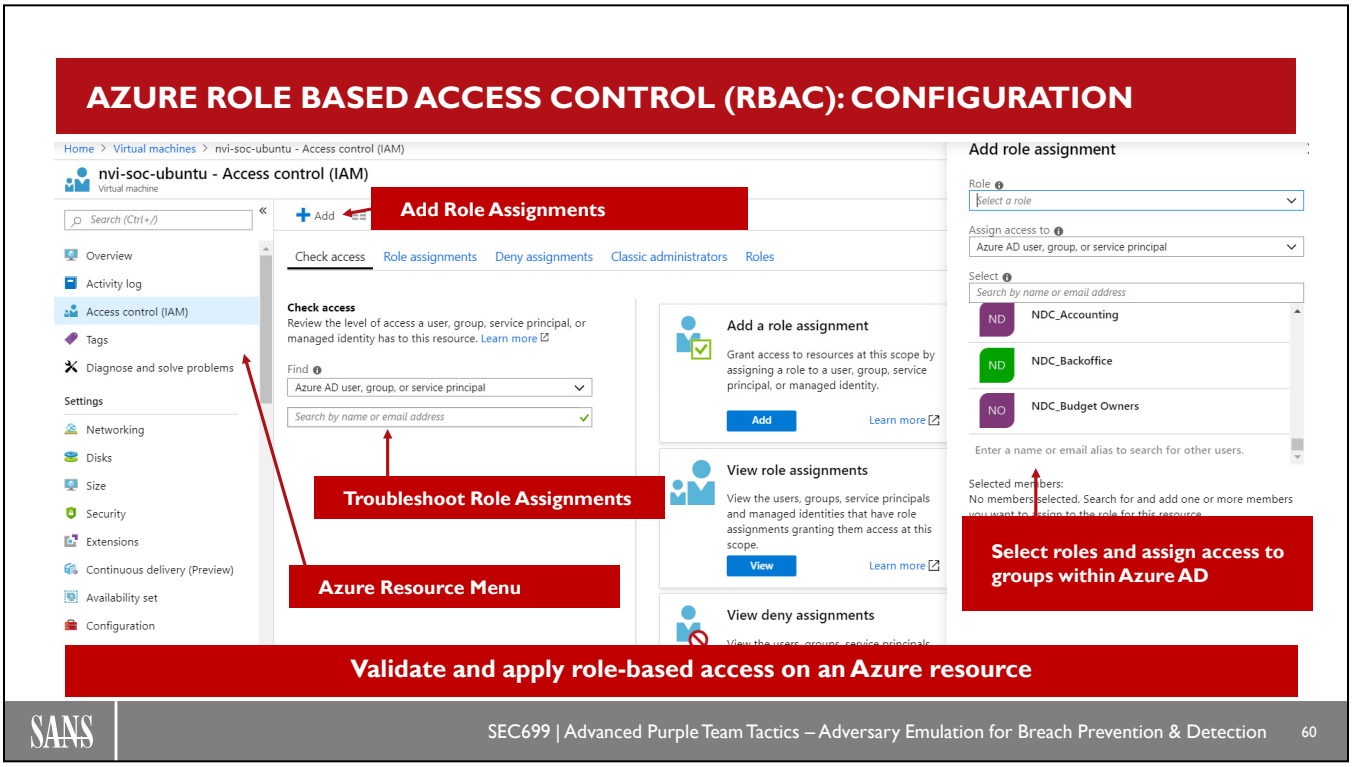
Source: https://docs.microsoft.com/en-us/azure/azure-australia/role-privileged

Role definition, the following list has four fundamental built-in roles:

- Owner: Has full access to all resources including the right to delegate access to others.
- Contributor: Can create and manage all types of Azure resources but can’t grant access to others.
- Reader: Can view existing Azure resources.
- User Access Administrator: Lets you manage user access to Azure resources.

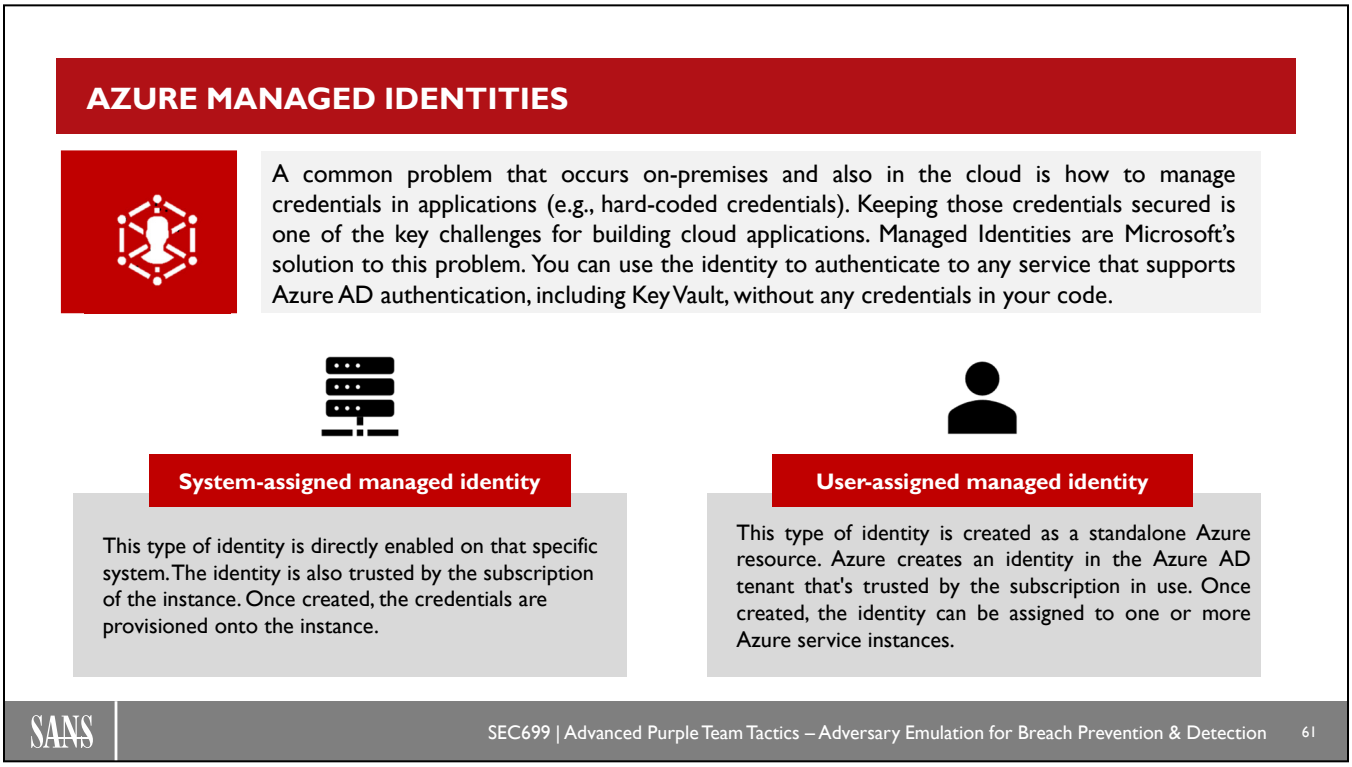
A scope:

- If you assign the Owner role to a user at the management group scope, that user can manage everything in all subscriptions in the management group.
- If you assign the Reader role to a group at the subscription scope, the members of that group can view every resource group and resource in the subscription.
- If you assign the Contributor role to an application at the resource group scope, it can manage resources of all types in that resource group, but not other resource groups in the subscription.



Azure Role Based Access Control (RBAC): Configuration

The slide above demonstrates an example configuration for an Azure resource, in this case, a Virtual Machine (labelled “nvi-soc-ubuntu”). The different menu entries show the simplicity of management in Azure AD.



Azure Managed Identities

A common problem that occurs on-premises and also in the cloud is how to manage credentials in applications (e.g., hard-coded credentials). Keeping those credentials secured is one of the key challenges for building cloud applications. Managed Identities are Microsoft's solution to this problem. You can use the identity to authenticate to any service that supports Azure AD authentication, including Key Vault, without any credentials in your code.

We can make a distinction between two different managed identity types:

System-assigned managed identity

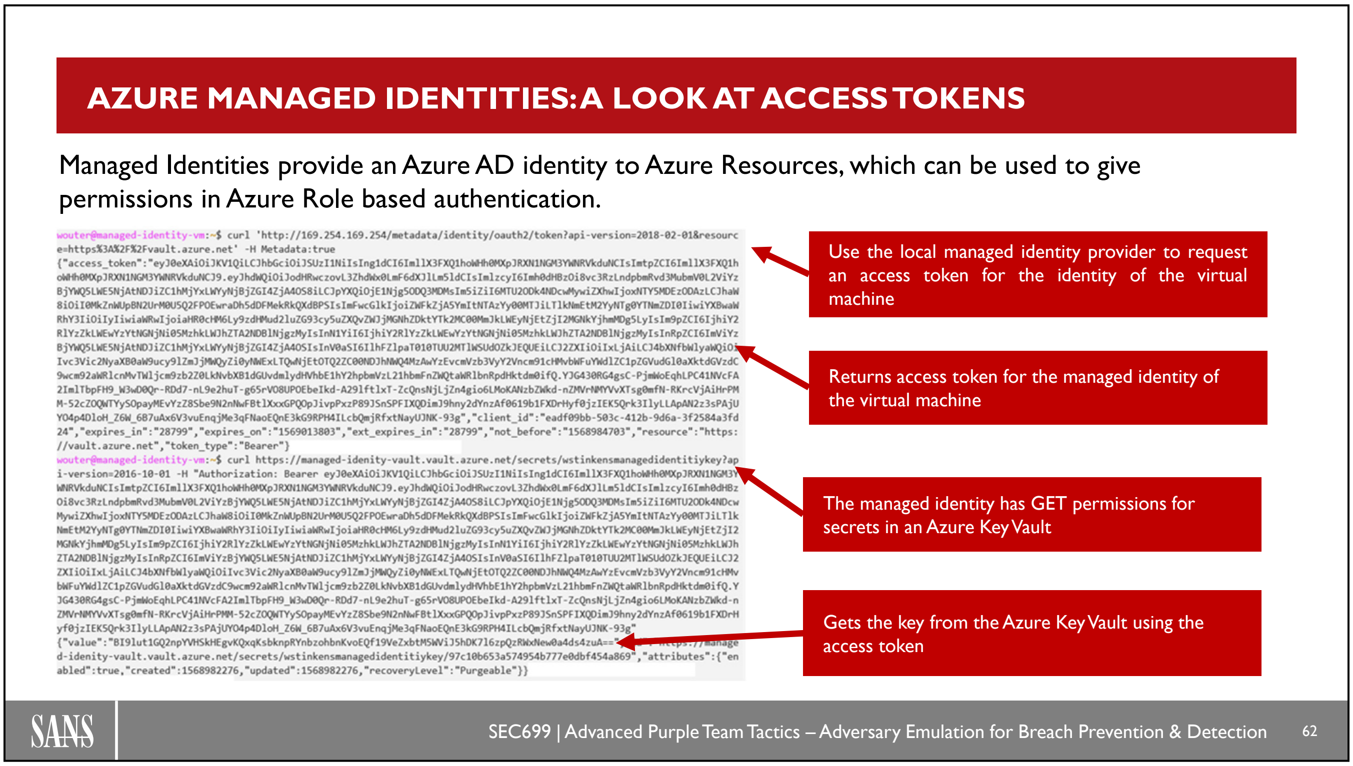
This type of identity is directly enabled on a specific system. The identity is also trusted by the subscription of the instance. Once created, the credentials are provisioned onto the instance.

User-assigned managed identity

This type of identity is created as a standalone Azure resource. Azure creates an identity in the Azure AD tenant that is trusted by the subscription in use. Once created, the identity can be assigned to one or more Azure service instances.

Imagine you have a web application that needs to connect to a database service "Azure SQL". We will first need to grant the web server (System Assigned Identity) access to the Azure SQL Server. You will also need to enable Azure AD authentication for the SQL server. You can select the "System Assigned Managed identity". As a second step, you must create a user in the database that represents the web server's system assigned identity.

Once the user is known, you can get an access token using the VM's system-assigned managed identity and use it to call Azure SQL. A full tutorial on this example scenario is described at https://docs.microsoft.com/en-us/azure/active-directory/managed-identities-azure-resources/tutorial-windows-vm-access-sql.



Azure Managed Identities: A Look at Access Tokens

This slide gives you an example of what the access token actually looks like and how managed identities are integrated within Azure.

The following steps occur when a specific Azure service needs to validate if that particular identity has access to the resource:

1. Any azure resource can request the local managed identity provider for an access token for the identity of a specific virtual machine;

2. The identity provider returns the access token for the managed identity of that virtual machine;

3. Once validated, the managed identity can use his permission, for example, to request secrets in Azure Key Vault;

4. Azure key vault is compatible with managed identities and validates the token and permissions; in case all permissions are correct, the managed identity is able to retrieve keys from the Azure Vault.

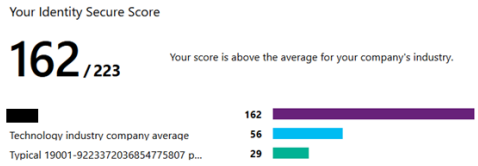
INTRODUCING THE IDENTITY SECURE SCORE

Azure AD Identity Secure Score is an indicator showing the alignment with Microsoft's best practice recommendations for Azure AD security configuration.

Benchmark Azure AD security configuration against peers

Recommends security improvements based on user impact and implementation cost

View security configuration evolution over time



NAME	SCORE IMPA...	USER IMPACT	IMPLEMENTATION COST
Require MFA for Azure AD privileged roles	0	Low	Low
Require MFA for all users	20	Moderate	Moderate
Do not allow users to grant consent to unmanaged applicati...	0	Moderate	Low
Designate fewer than 5 global admins	0	Low	Low
Designate more than one global admin	0	Low	Low
Use limited administrative roles	0	Low	Low
Do not expire passwords	0	Moderate	Low
Delete/block accounts not used in last 30 days	1	Moderate	Low
Enable policy to block legacy authentication	20	Moderate	Moderate
Turn on sign-in risk policy	0	Moderate	Moderate
Turn on user risk policy	13	Moderate	Moderate
Register all users for multi-factor authentication	7	High	High
Enable Password Hash Sync if hybrid	0	Low	Low
Enable self-service password reset	0	Moderate	Moderate

Introducing the Identity Secure Score

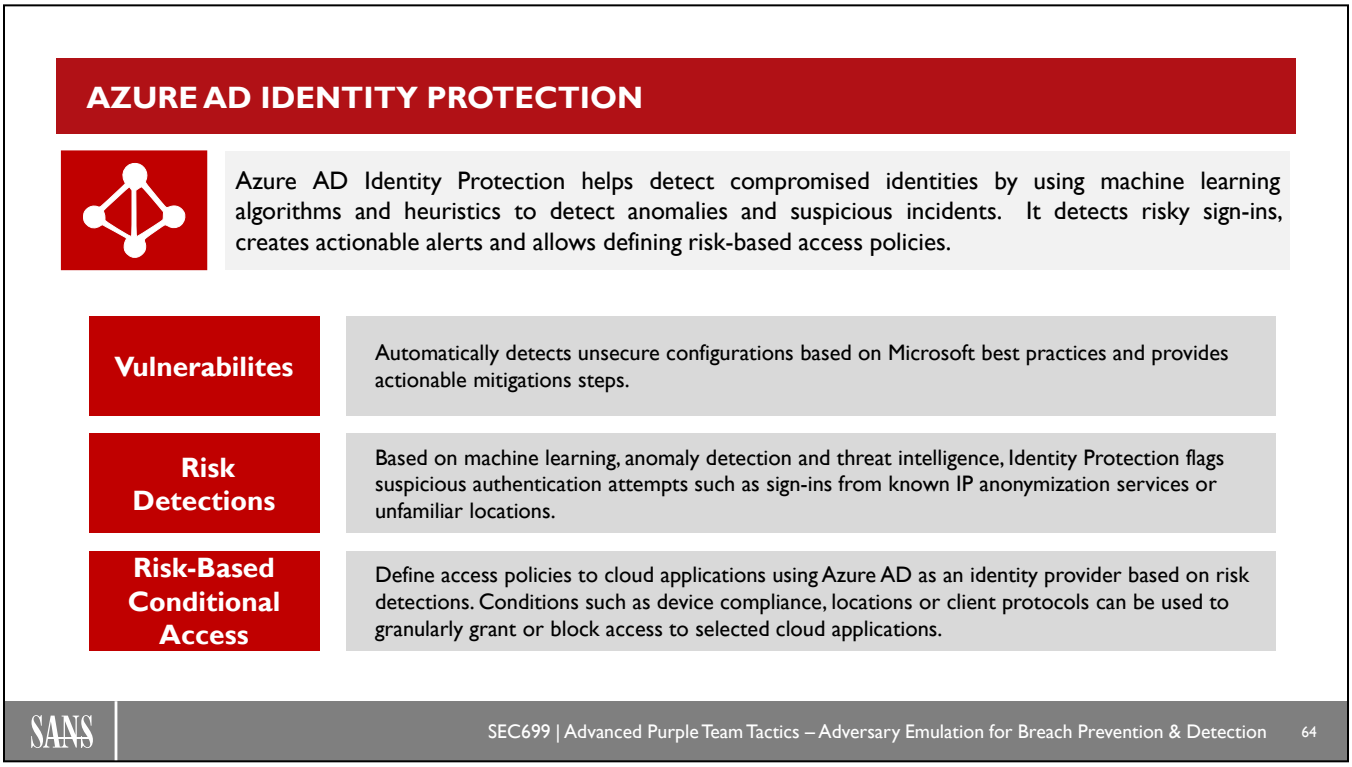
We all love a little competition and Microsoft has integrated several scoring mechanisms that you can use to compare against industry peers. The identity secure score is available in all editions of Azure AD; however, some recommendations that are required to improve your score are only available with premium licenses.

Every 48 hours, Azure reviews your security configuration and compares your settings with the recommended best practices. Based on the outcome of this evaluation, a new score is calculated for your directory. Each recommendation is measured based on your Azure AD configuration. If you are using additional third-party products instead of Microsoft products to "comply" with best practices, you can indicate this configuration in the settings of an improvement action.

Typical recommendations are:

- Activate MFA
- Use limited administrative roles
- Turn-on additional features such as sign-in risk policy

It is an ideal starting point to validate your current exposure and develop a plan to further improve your identity and access management mechanisms.



Azure AD Identity Protection

Azure AD Identity Protection helps detect compromised identities by using machine learning algorithms and heuristics to detect anomalies and suspicious incidents. It detects risky sign-ins, creates actionable alerts and allows defining risk-based access policies.

Microsoft provides the following identity protection capabilities:

Vulnerabilities & Risky Accounts

Automatically detects unsecure configurations based on Microsoft best practices and provides actionable mitigations steps. Calculates risk levels of both individual sign-ins and users.

Risk Detections

Based on machine learning, anomaly detection and threat intelligence, Identity Protection flags suspicious authentication attempts such as sign-ins from known IP anonymization services or unfamiliar locations.

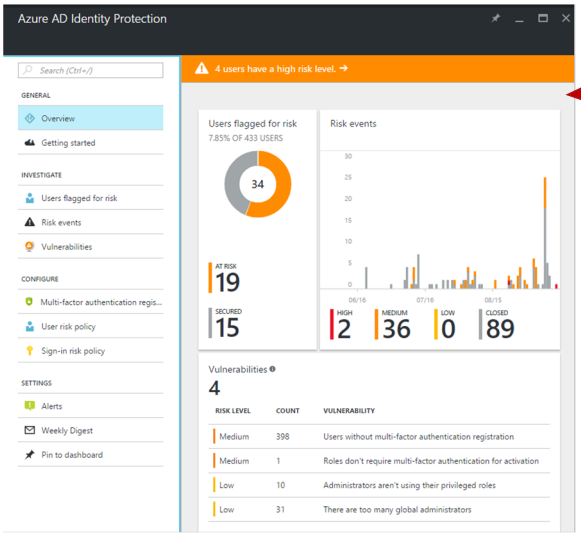
Risk-based Conditional Access

Define access policies to cloud applications using Azure AD as an identity provider based on risk detections. Conditions such as device compliance, locations or client protocols can be used to granularly grant or block access to selected cloud applications.

Reference:

https://docs.microsoft.com/en-us/azure/active-directory/identity-protection/overview-identity-protection

AZURE AD IDENTITY PROTECTION: DASHBOARD



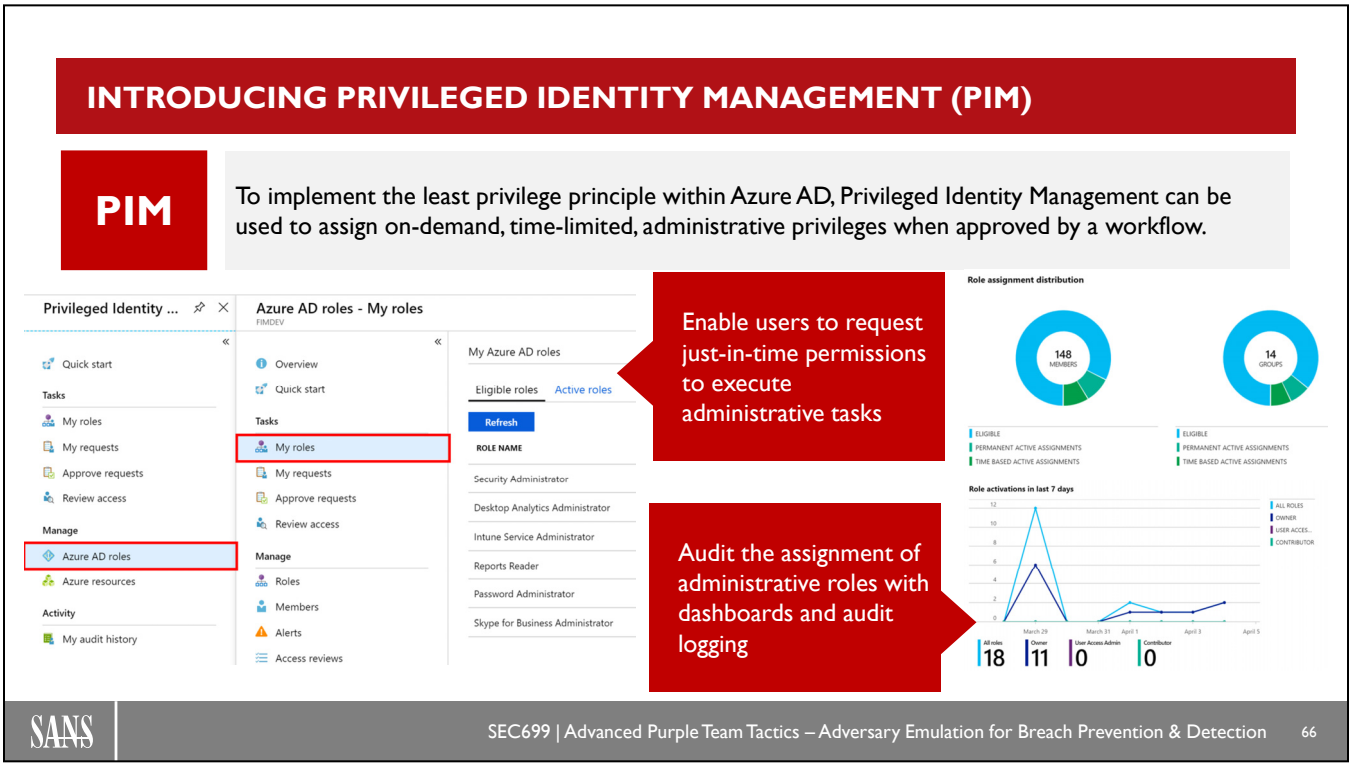
Includes an overview of users that are flagged for risk, meaning anomalies were detected during sign-ins (e.g., an anonymized VPN endpoint was used for this sign-in)

Looks at the events and calculates a severity based on several factors. It will look at the sign-in events and vulnerabilities within your environment. In case a risky sign-in was detected on an identity that has a high vulnerability (e.g., leaked credentials), this will be marked as a high-risk event.

A list of vulnerabilities is displayed on the dashboard; you have one generic overview to investigate and remediate the specific vulnerabilities. You can also see how many identities are impacted by this vulnerability within Azure AD.

Azure AD Identity Protection: Dashboard

This slide gives you a typical example of an Azure Identity dashboard, on which you can immediately identify the Risk Detections, Events, and Vulnerabilities that were found in this Azure AD configuration.



Introducing Privileged Identity Management (PIM)

To implement the least privilege principle within Azure AD, Privileged Identity Management (PIM) can be used to assign on-demand, time-limited, administrative privileges when approved by a workflow.

In PIM, the following features are available (from: https://docs.microsoft.com/en-us/azure/active-directory/privileged-identity-management/pim-configure)

- Provide just-in-time privileged access to Azure AD and Azure resources – this will limit the timeframe that a privileged account is exposed (e.g., a dynamic FW rule)
- Assign time-bound access to resources using start and end dates – this will limit the timeframe that a privileged account has access
- Require approval to activate privileged roles – You can create your own approval flows in case privileged access is required
- Enforce multi-factor authentication to activate any role
- Use justification to understand why users activate – You can integrate message into your flow and have an audit trail
- Get notifications when privileged roles are activated – Continuously monitor the usage of high privileged accounts
- Conduct access reviews to ensure users still need roles
- Download audit history for internal or external audit

Course Roadmap

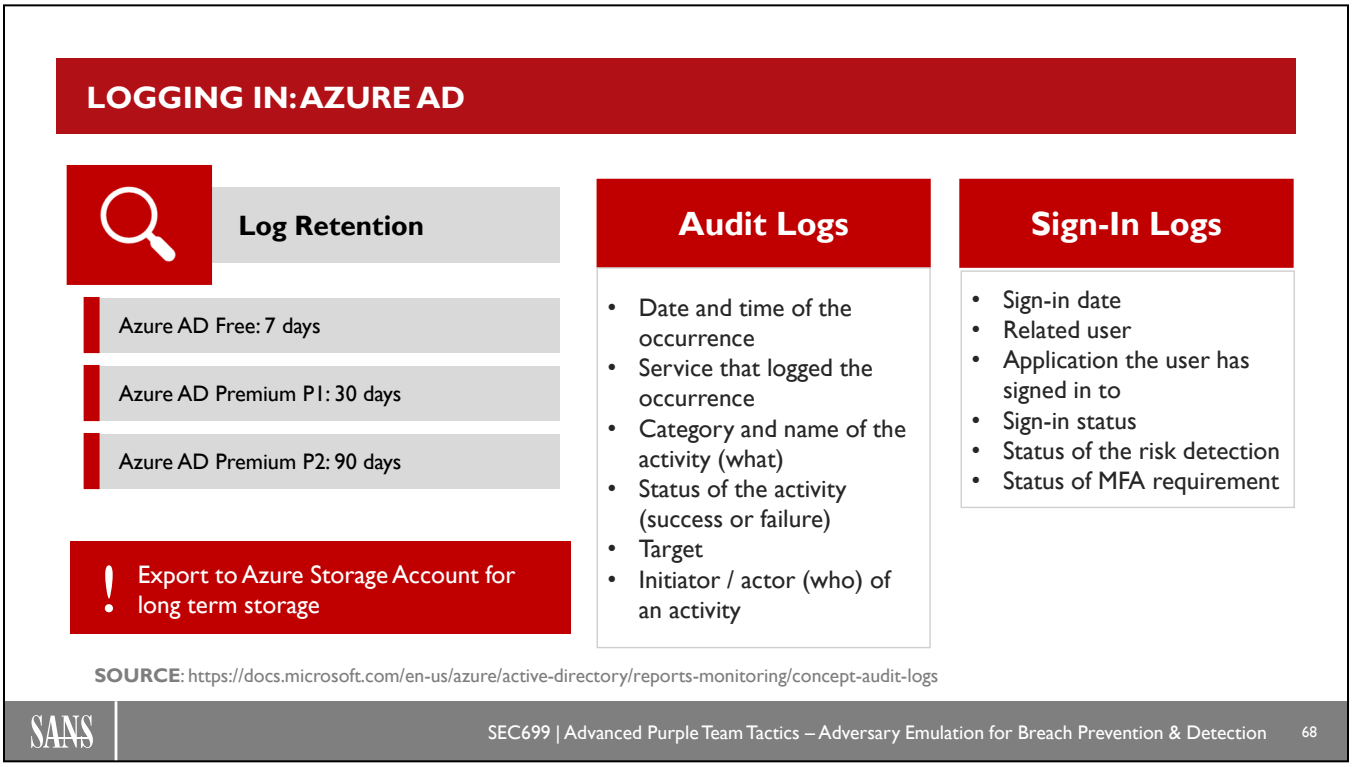
- Introduction & Key Tools
- Initial Access
- Lateral Movement
- Persistence
- **Azure AD & Emulation Plans**
- Adversary Emulation Capstone

SEC699.5

Azure AD
Azure AD Structure and Management
Azure AD Hybrid Authentication
Azure AD Authentication Methods
Azure AD Conditional Access
Introduction to Azure Identities
Azure AD Security Logging

Executing emulation plans
APT-28 Emulation Plan
Exercise: APT-28 Emulation Plan
APT-34 Emulation Plan
Exercise: APT-34 Emulation Plan
Turla Emulation Plan
Exercise: Turla Emulation plan

This page intentionally left blank.



Logging in: Azure AD

By default, Azure AD has logging enabled; however, the retention period depends on your license.

It is recommended to export these logs to an Azure storage account. When the logs are exported, you can define your own retention period and, in case an incident occurred before the default retention period, you will be able to analyze older events via that storage account. Note that you are not able to export logs if you are using the Azure AD free license.

The following logs are available:

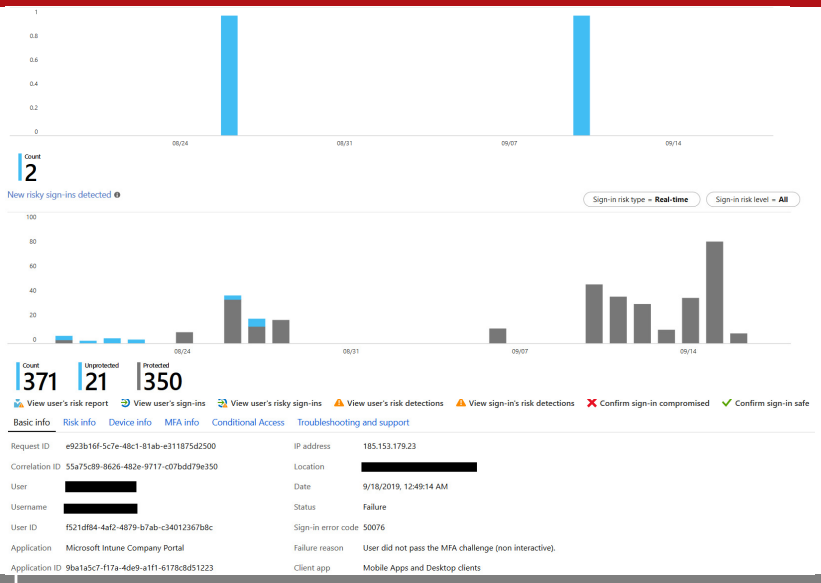
Audit Logs, which include the following details:

- Date and time of the occurrence
- Service that logged the occurrence
- Category and name of the activity (what)
- Status of the activity (success or failure)
- Target
- Initiator / actor (who) of an activity

Sign-in Logs, which include the following details:

- Sign-in date
- Related user
- Application the user has signed in to
- Sign-in status
- Status of the risk detection
- Status of MFA requirement

AN EXAMPLE OF AN AZURE AD RISK DETECTION INVESTIGATION (1)



In Azure's Identity Protection Risky Sign-In, we can further investigate and triage authentication attempts that were flagged as suspicious.

Investigation by an analyst is required because of the high false positive rate and limited tuning options.

An Example of an Azure AD Risk Detection Investigation (1)

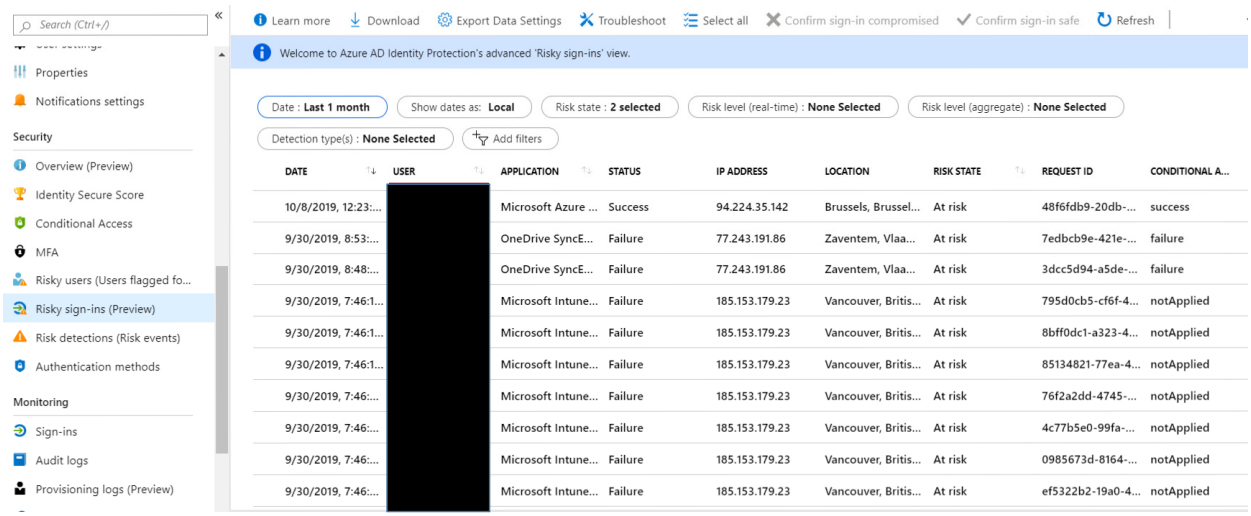
In Azure's Identity Protection Risky Sign-In, we can further investigate and triage authentication attempts that were flagged as suspicious. Investigation by an analyst is required because of the high false positive rate and limited tuning options available.

Note that the initial overview provides some information already:

- IP address of the sign-in: "185.153.179.23"
- Outcome of the attempt: "Failure"
- Reason for the outcome: "User did not pass the MFA challenge (non interactive)"
- The client app: "Mobile Apps and Desktop clients"

We do, however, need some additional information to make a correct assessment!

AN EXAMPLE OF AN AZURE AD RISK DETECTION INVESTIGATION (2)



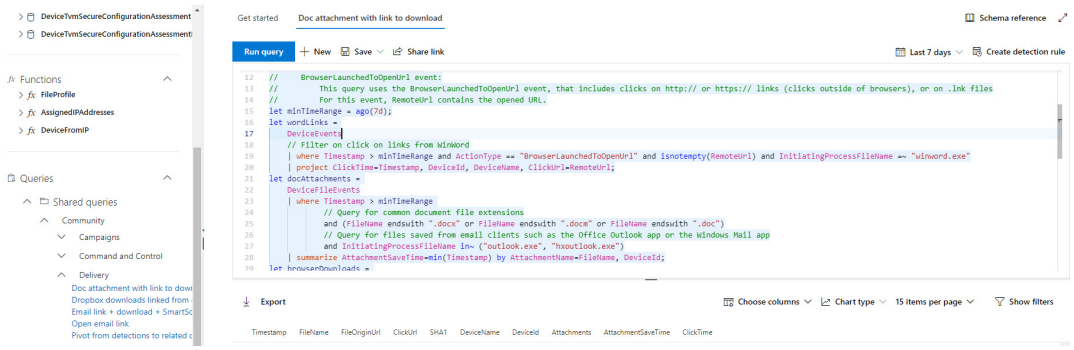
An Example of an Azure AD Risk Detection Investigation (2)

When taking a step back and reviewing all of the risky sign-ins, we immediately notice multiple failures coming from the same IP address in Vancouver (British Columbia, Canada). Our example organization (NVISO) is European without a presence in Canada; thus, such attempts are, indeed, suspicious. In this specific case, we can identify a password guessing attack.

ADVANCED DETECTION CAPABILITIES: DEFENDER FOR ENDPOINT

Defender for Endpoint is Microsoft's EDR tool that is available for Windows, Linux, and MacOS clients. It uses an online, cloud, endpoint from which alerts can be investigated and handled, but also advanced manual threat hunts can be launched.

Advanced hunting



SOURCE: https://securitycenter.windows.com/hunting

Advanced Detection Capabilities: Defender for Endpoint

Defender for Endpoint is Microsoft's EDR tool that is available for Windows, Linux, and MacOS clients. It uses an online, cloud, endpoint from which alerts can be investigated and handled, but also advanced manual threat hunts can be launched. The threat hunts are executed by running queries across all installed hosts. Security analysts can write their own rules or can leverage prebuilt community rules. The screenshot on the slide shows an example of a rule called "Doc attachment with link to download".

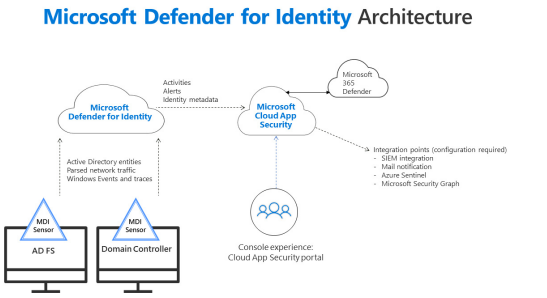
Should the hunt be successful, there's also a handy function that allows you to convert this query in real-time detection.

If you have the right license and access, the hunting feature for Defender for Endpoint can be found at https://securitycenter.windows.com/hunting.

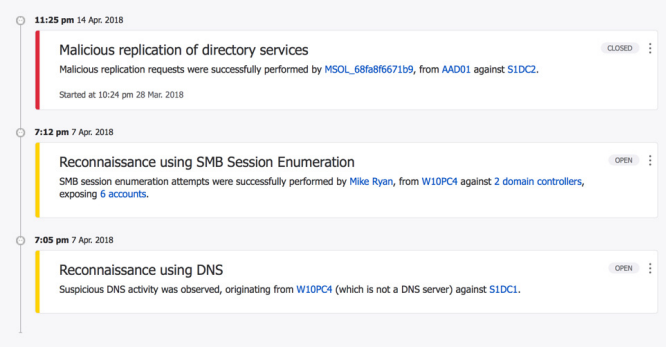
ADVANCED DETECTION CAPABILITIES: DEFENDER FOR IDENTITY

Defender for Identity (previously Microsoft ATA / Azure ATP) allows you to monitor both an on-premise and Azure AD for advanced attacks. It achieves this dual view by also installing an agent on the local domain controllers.

Defender for Identity has a key focus on detecting authentication attacks (e.g., Kerberos ticket manipulation), so the golden ticket attack could be detected when this feature is enabled!



SOURCE: https://docs.microsoft.com/en-us/defender-for-identity/what-is



Advanced Detection Capabilities: Azure ATP

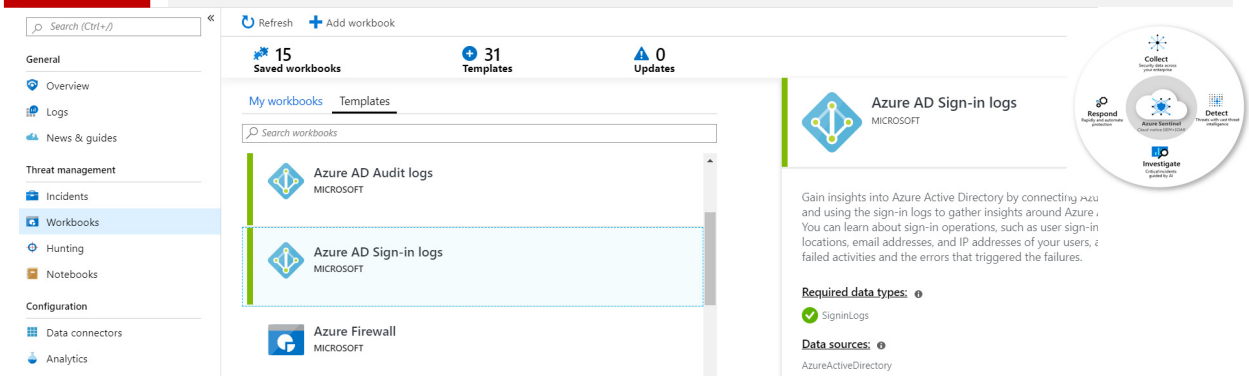
We previously touched upon Kerberos and typical AD lateral movement strategies and noted that this is quite often difficult to detect. As a reminder, Kerberos is a stateless protocol which does not keep track of assigned tickets, hence attacks where tickets are crafted by adversaries (e.g., golden or silver tickets) are hard to detect.

Defender for Identity (previously Microsoft ATA / Azure ATP) allows you to monitor both an on-premise and Azure AD for advanced attacks. It achieves this dual view by also installing an agent on the local domain controllers. The agents on the domain controllers both look at the local Windows event log and monitor network traffic.

As Defender for Identity tracks all relevant authentication information (e.g., issuing of tickets and authentications against the DC), it can detect crafted ticket and thus detect, for example, a golden ticket attack.

ADVANCED DETECTION CAPABILITIES: AZURE SENTINEL

Azure AD logs can be analyzed by Azure Sentinel which is a cloud-native SIEM created by Microsoft. The service is only available in a pay-as-you go model and is calculated on the amount of logs ingested by Sentinel.

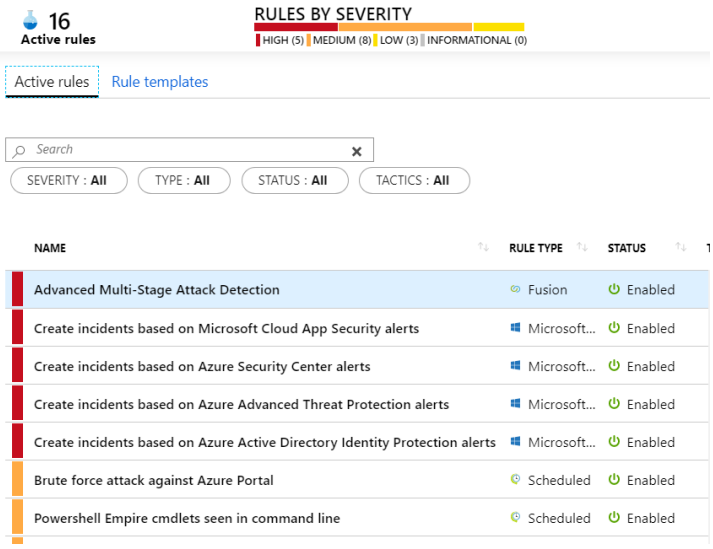


Advanced Detection Capabilities: Azure Sentinel

Azure Sentinel is a “cloud-native” SIEM offered by Microsoft that aims to replace typical on-prem SIEM solutions. Azure Sentinel is based on log analytics workspaces and data connectors; it will look at these logs and map them against use cases (called “Analytics”) that are enabled. The main goal is to provide a flexible solution that can easily integrate both with other cloud services (such as Azure AD) and on-prem resources (e.g., local firewall logs).

More information on Azure Sentinel can be found here:
https://docs.microsoft.com/en-us/azure/sentinel/overview

ADVANCED DETECTION CAPABILITIES: AZURE SENTINEL – EXAMPLE RULES



Use cases are defined in the analytics page, analytics are used to correlate alerts into incidents.

By default, Azure Sentinel has built-in correlation rules; these can be enabled and used for your Azure environment. New correlation rules will be added on a frequent basis.

You can create your own correlation rules and try to fine-tune the detection rate to minimize false positives. For identity protection, you have the Azure identity protection alerts and many others.

Advanced Detection Capabilities: Azure Sentinel – Example Rules

Use cases are defined in the analytics page, analytics are used to correlate alerts into incidents. By default, Azure Sentinel has built-in correlation rules; these can be enabled and used for your Azure environment. New correlation rules will be added on a frequent basis. You can create your own correlation rules and try to fine-tune the detection rate to minimize false positives. For identity protection, you have the Azure identity protection alerts and many others.

Course Roadmap

- Introduction & Key Tools
- Initial Access
- Lateral Movement
- Persistence
- **Azure AD & Emulation Plans**
- Adversary Emulation Capstone

SEC699.5

Azure AD
Azure AD Structure and Management
Azure AD Hybrid Authentication
Azure AD Authentication Methods
Azure AD Conditional Access
Introduction to Azure Identities
Azure AD Security Logging

Executing emulation plans
APT-28 Emulation Plan
Exercise: APT-28 Emulation Plan
APT-34 Emulation Plan
Exercise: APT-34 Emulation Plan
Turla Emulation Plan
Exercise: Turla Emulation plan

This page intentionally left blank.

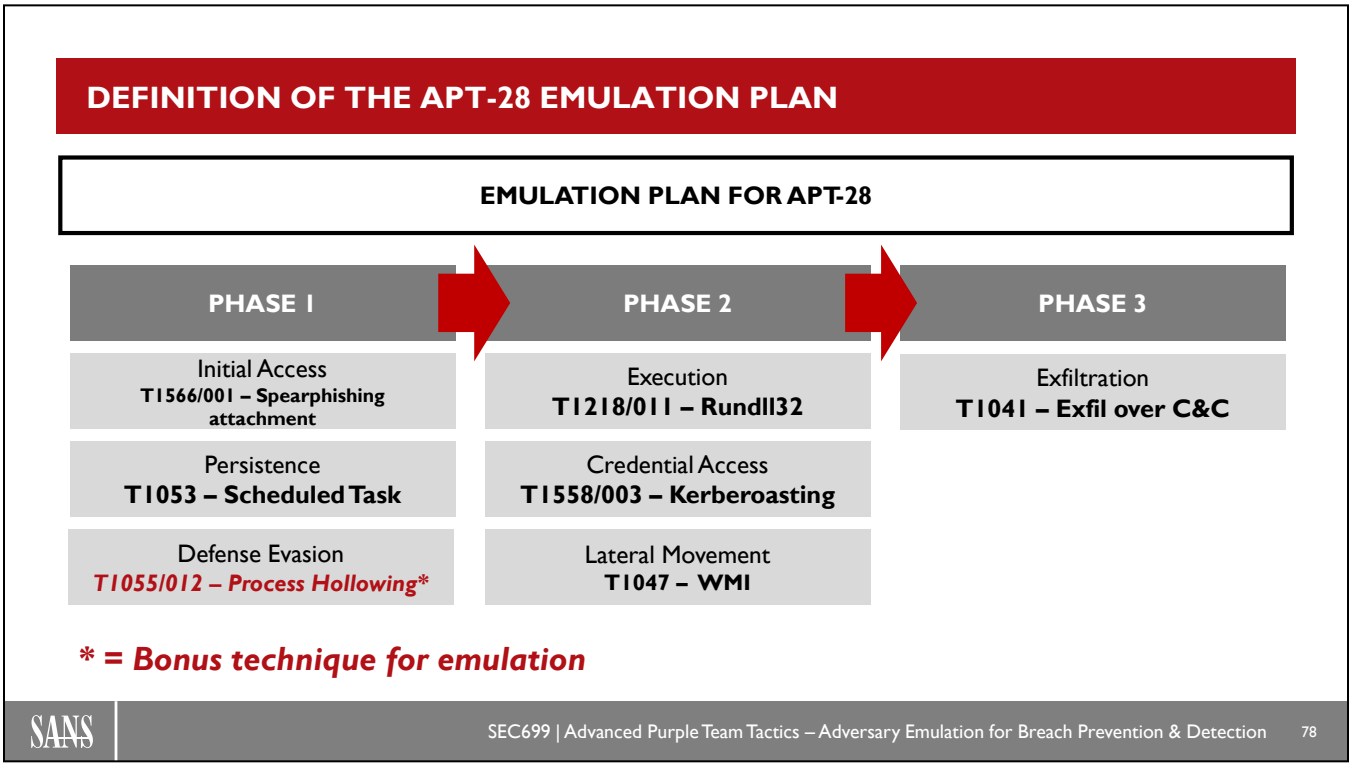
APT28: Introduction and Common Techniques

APT-28 (also known as Sofacy or Fancy bear) is a Russian cyber espionage group. It has been associated by several cybersecurity firms with the Russian government (some attribute it more specifically to the Russian military intelligence agency GRU). In 2018, an indictment by the US Special Counsel identified APT-28 as two GRU units known as Unit 26165 and Unit 74455.

MITRE's ATT&CK report on the group can be found on https://attack.mitre.org/groups/G0007/.

With regards to targets, APT-28 is known to have a global approach, but a specific focus on USA and (Western) Europe. They mostly target (military) government agencies. Reported victims have included DNC (Democratic National Committee), OSCE (Organization for Security and Co-Operation in Europe), and NATO.

Further references:
https://www2.fireeye.com/rs/fireye/images/rpt-apt28.pdf
https://www.crowdstrike.com/blog/who-is-fancy-bear/



Definition of the APT-28 Emulation Plan

We will now draft an emulation plan that mimics techniques known to be abused by APT-28. We will define the following steps as part of the plan:

Phase 1

- Initial Access – T1566/001 – Spearphishing attachment
- Persistence – T1053 – Scheduled Task
- Defense Evasion – T1055/012 – Process Hollowing (Bonus technique)

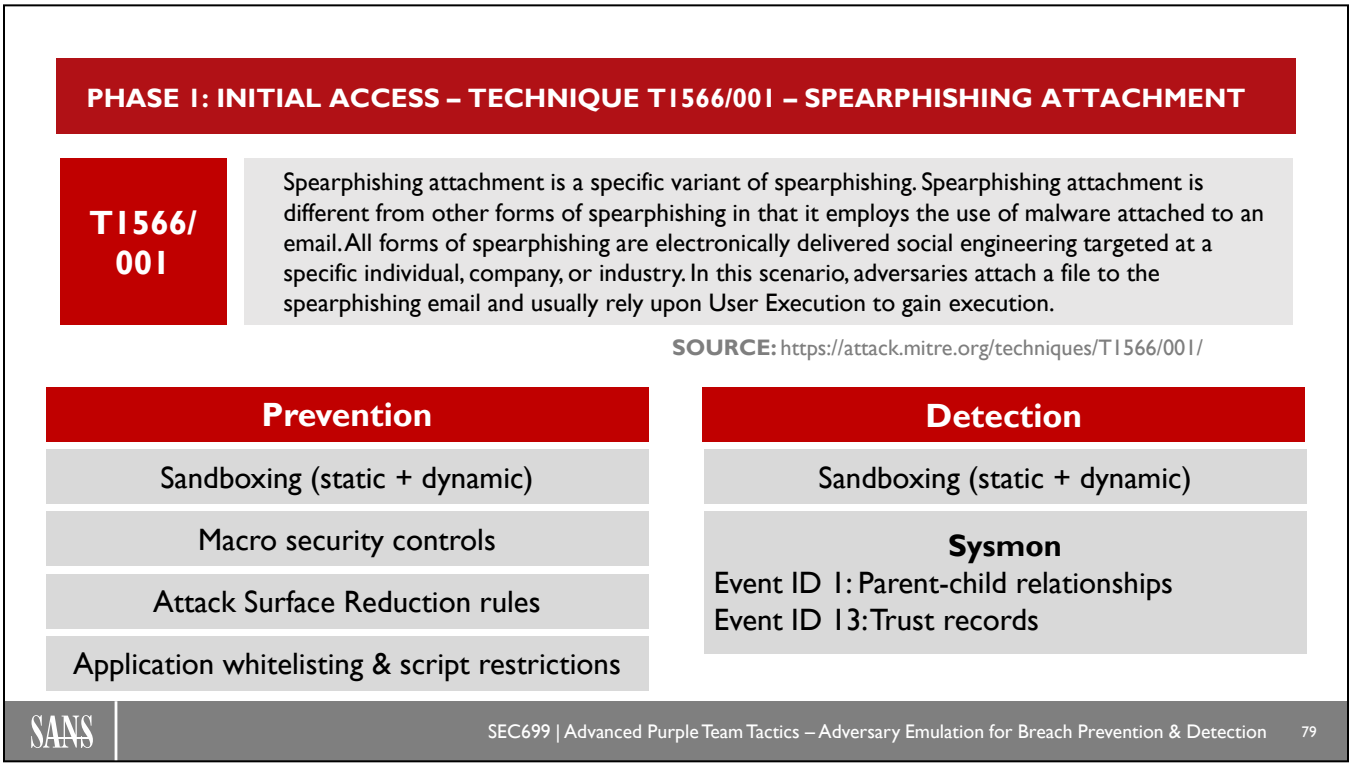
Phase 2

- Execution – T1218/011 – Rundll32
- Privilege Escalation – T1558/003 – Kerberoasting
- Lateral Movement – T1047 – Windows Management Instrumentation

Phase 3

- Execution – T1041 – Exfil over C&C

We have added short technique descriptions in the next few slides (extracted from MITRE ATT&CK). We will further prepare and execute the emulation plan in our Workbook!



Phase 1: Initial Access – Technique T1566/001 – Spearphishing Attachment

From MITRE's ATT&CK framework:

"Spearphishing attachment is a specific variant of spearphishing. Spearphishing attachment is different from other forms of spearphishing in that it employs the use of malware attached to an email. All forms of spearphishing are electronically delivered social engineering targeted at a specific individual, company, or industry. In this scenario, adversaries attach a file to the spearphishing email and usually rely upon User Execution to gain execution."

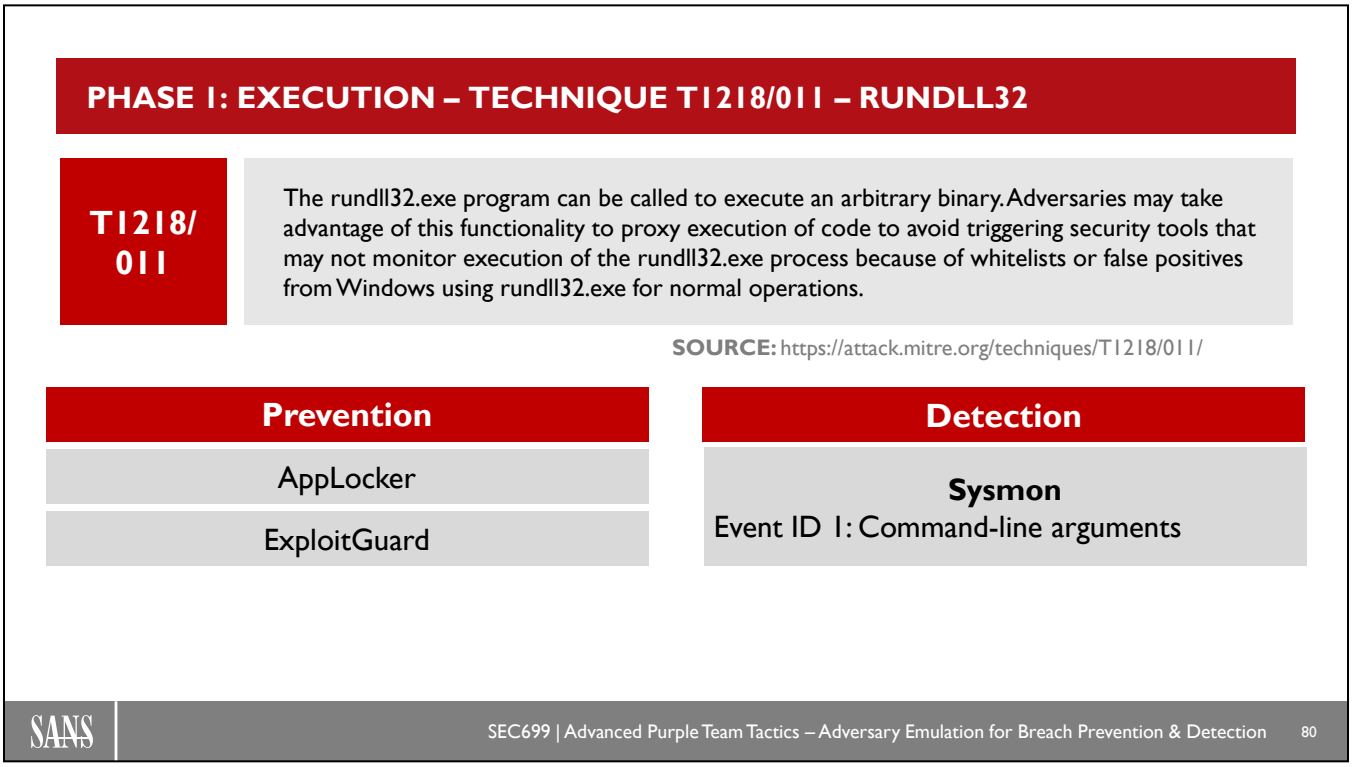
How could we prevent this?

- Sandboxing of attachments to block suspicious / malicious payloads
- Macro security controls
- Attack Surface Reduction rules
- Application whitelisting & script restrictions

How could we detect this?

- Sandboxing of attachments to detect suspicious / malicious payloads
- Using Sysmon, we could search for the following:
  - Event ID 1 (Process Creation): Analyze parent-child relationships to find weird instances
  - Event ID 13 (RegistryEvent Value Set): Analyze registry values for trust records that are being set, which indicated Office documents trusted for macro execution

The full ATT&CK entry can be found at https://attack.mitre.org/techniques/T1193/.



Phase 1: Execution – Technique T1218/011 – Rundll32
From MITRE's ATT&CK framework:

"The rundll32.exe program can be called to execute an arbitrary binary. Adversaries may take advantage of this functionality to proxy execution of code to avoid triggering security tools that may not monitor execution of the rundll32.exe process because of whitelists or false positives from Windows using rundll32.exe for normal operations."

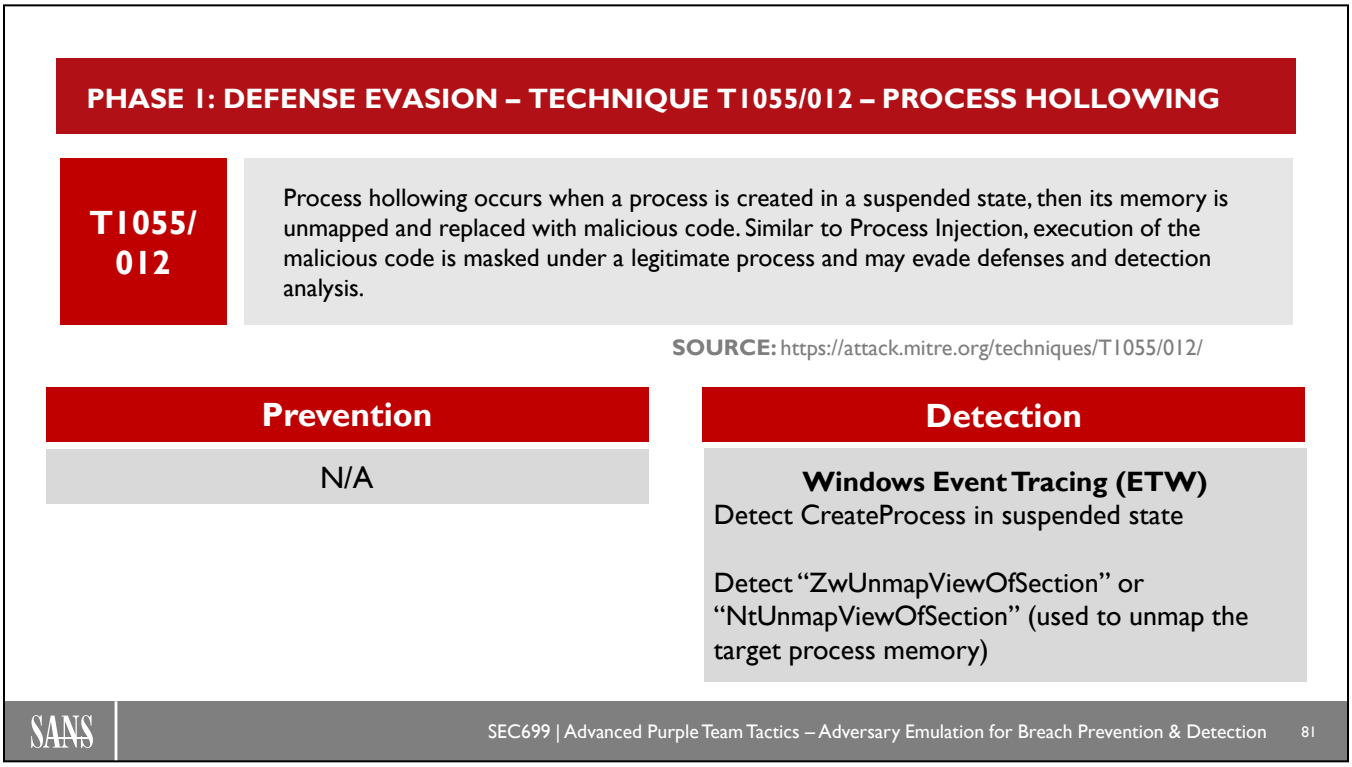
How could we prevent this?

- AppLocker (restrictions on DLL execution)
- ExploitGuard (restrictions on DLL loading / execution)

How could we detect this?

- Using Sysmon, we could search for the following:
  - Event ID 1 (Process Creation): Analyze command-line arguments passed to rundll32.exe

The full ATT&CK entry can be found at https://attack.mitre.org/techniques/T1085/.



Phase 1: Defense Evasion – Technique T1055/012 – Process Hollowing
From MITRE's ATT&CK framework:

"Process hollowing occurs when a process is created in a suspended state then its memory is unmapped and replaced with malicious code. Similar to Process Injection, execution of the malicious code is masked under a legitimate process and may evade defenses and detection analysis."

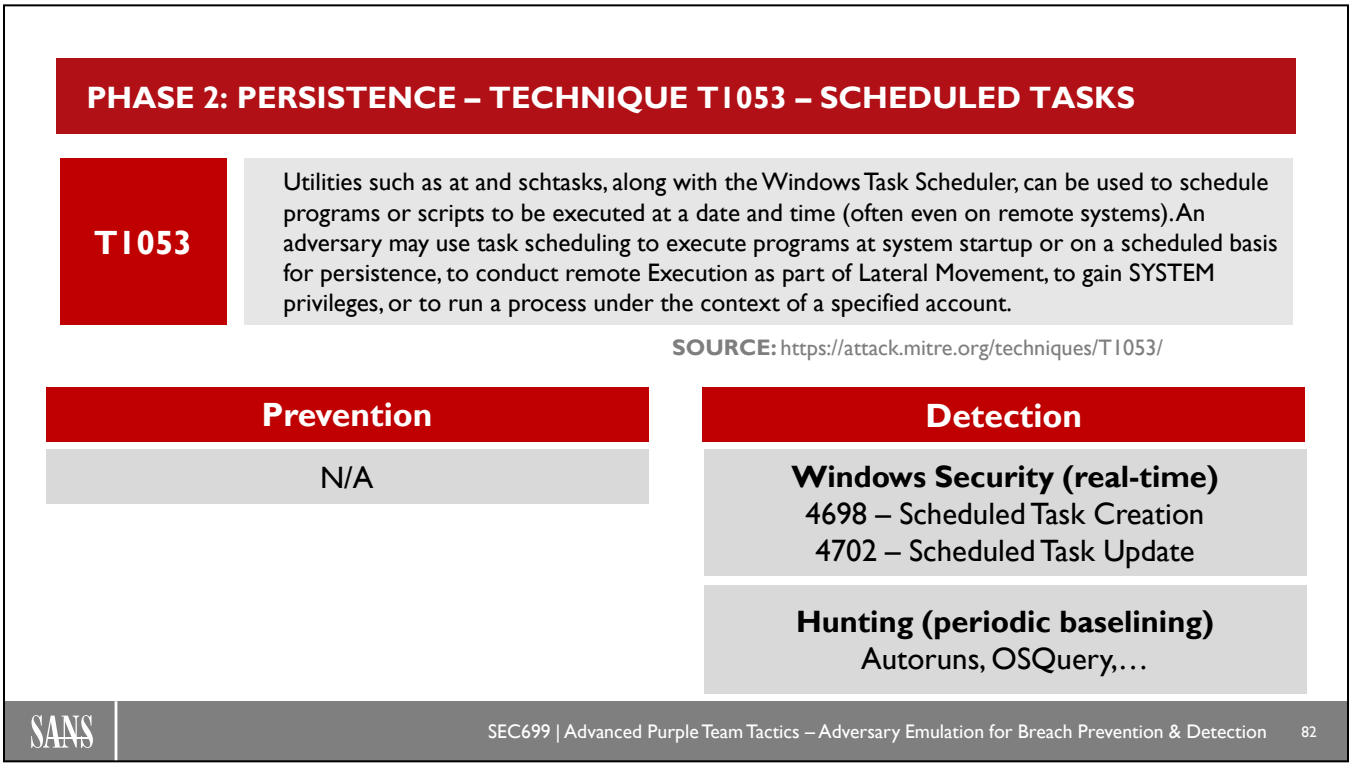
How could we prevent this?

- N/A => This is a built-in feature in Windows and can thus not be disabled

How could we detect this?

- This is a rather stealth attack strategy, which will require in-depth visibility
  - Windows Event Tracing (ETW) can be used to detect the steps of process hollowing
    - Processes being created in a suspended state
    - The use of specifics calls to unmap memory
  - Note that several EDR vendors have built-in techniques to detect this

The full ATT&CK entry can be found at https://attack.mitre.org/techniques/T1093/.



Phase 2: Persistence – Technique T1053 – Scheduled Tasks
From MITRE's ATT&CK framework:

"Utilities such as at and schtasks, along with the Windows Task Scheduler, can be used to schedule programs or scripts to be executed at a date and time (often even on remote systems). An adversary may use task scheduling to execute programs at system startup or on a scheduled basis for persistence, to conduct remote Execution as part of Lateral Movement, to gain SYSTEM privileges, or to run a process under the context of a specified account."

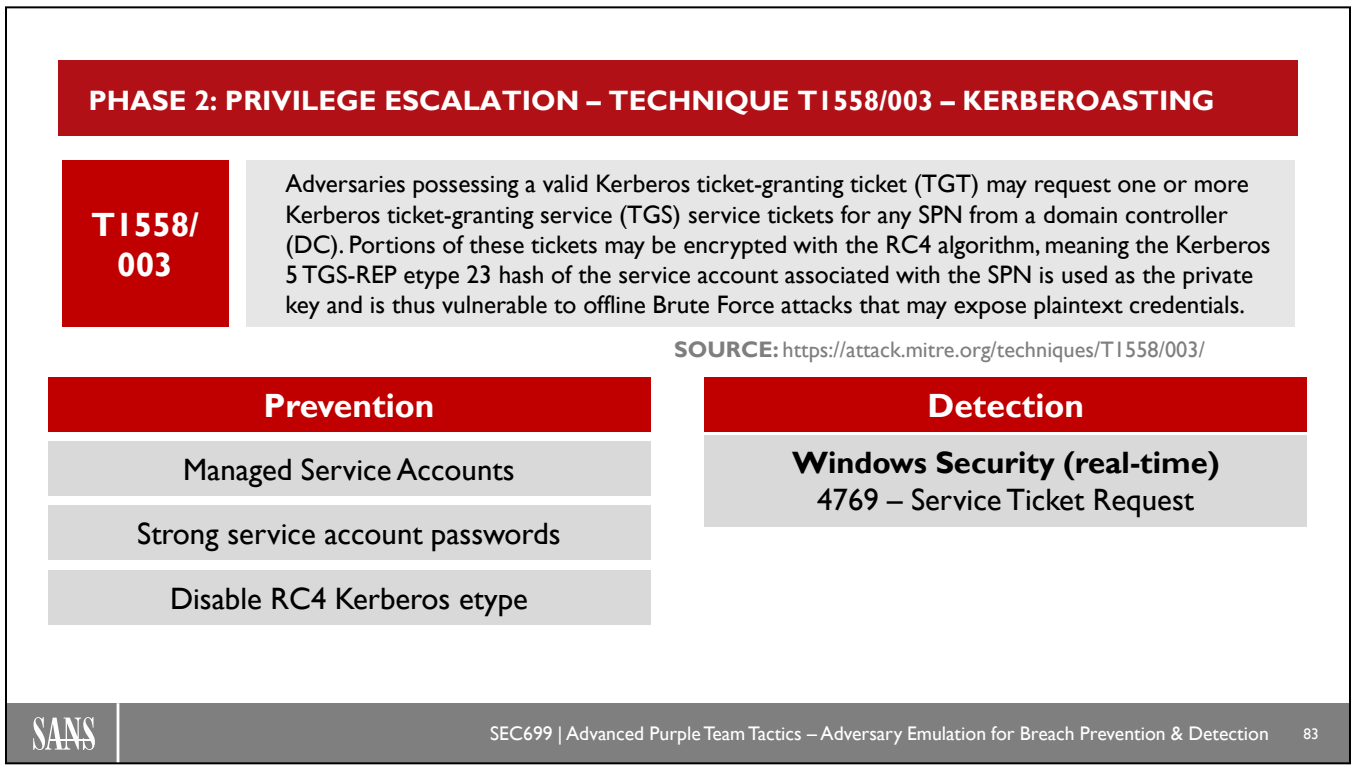
How could we prevent this?

- N/A => Task scheduling is a built-in feature on most Operating Systems and can thus typically not be disabled

How could we detect this?

- Real-time detection in Windows
  - Look for event ID 4698 – Scheduled Task Creation
  - Look for event ID 4702 – Scheduled Task Update

- Hunting
  - Periodic baselining as described per the slide. The students taking this 600 level course should have a good understanding of what baselining is.



Phase 2: Privilege Escalation – Technique T1558/003 – Kerberoasting
From MITRE's ATT&CK framework:

"Adversaries possessing a valid Kerberos ticket-granting ticket (TGT) may request one or more Kerberos ticket-granting service (TGS) service tickets for any SPN from a domain controller (DC). Portions of these tickets may be encrypted with the RC4 algorithm, meaning the Kerberos 5 TGS-REP etype 23 hash of the service account associated with the SPN is used as the private key and is thus vulnerable to offline Brute Force attacks that may expose plaintext credentials."

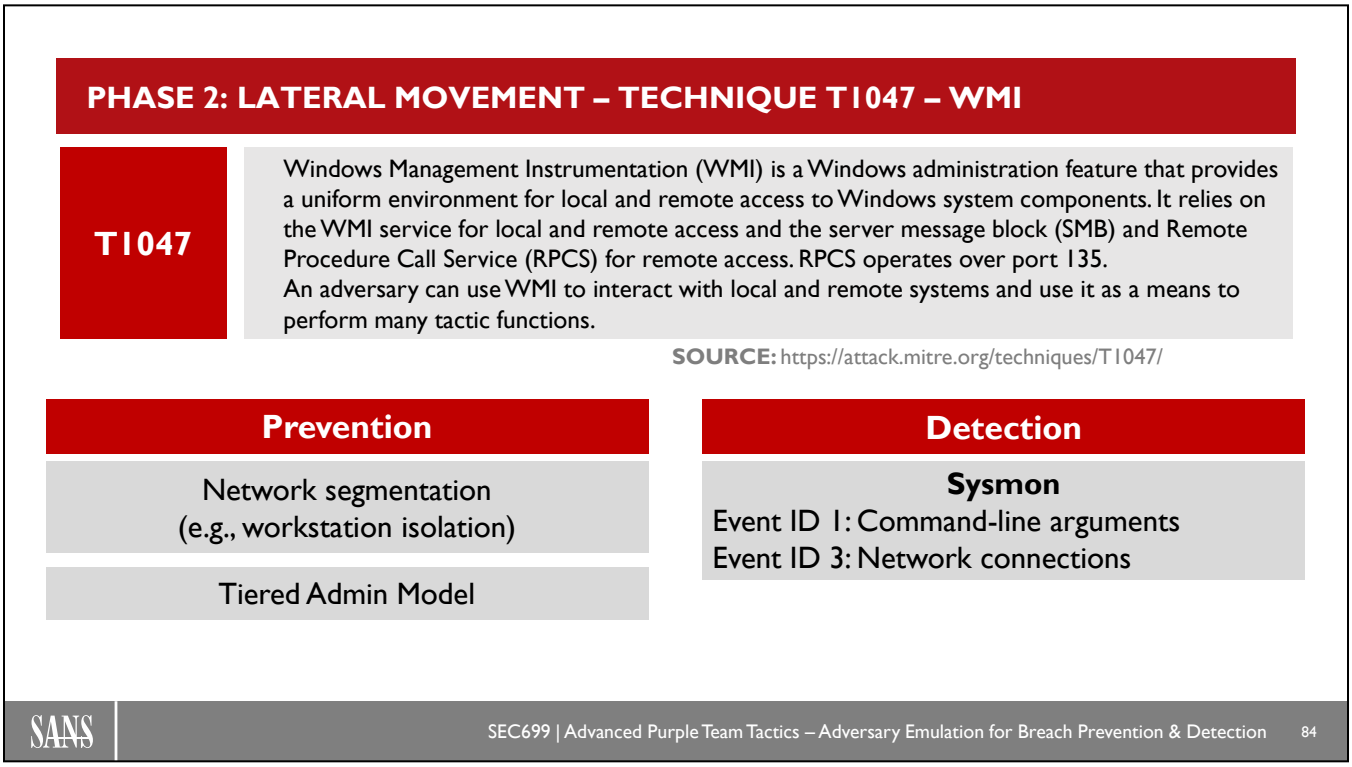
How could we prevent this?

- As Kerberoasting relies on brute force attacks against service accounts, ensure that service accounts are configured with a strong password. Microsoft provides "Managed Service Accounts" to achieve this
- Disable the RC4 encryption type to slow down / complicate brute force attacks

How could we detect this?

- Look for Windows event ID 4769 (Service Ticket Request)
  - If one account is requesting service tickets for a variety of service accounts in a short period, this should be suspicious (especially if the encryption type is RC4)

The full ATT&CK entry can be found at https://attack.mitre.org/techniques/T1208/.



Phase 2: Lateral Movement – Technique T1047 – WMI
From MITRE's ATT&CK framework:

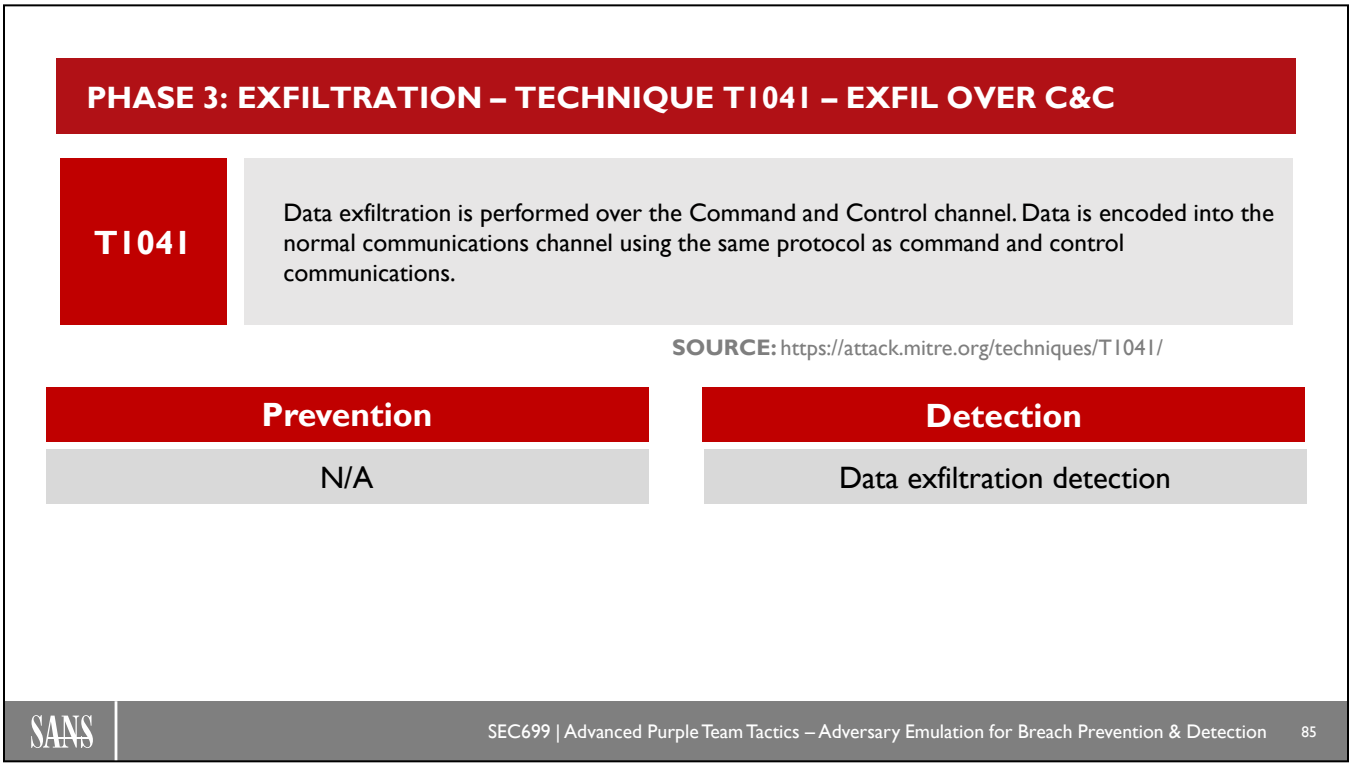
"Windows Management Instrumentation (WMI) is a Windows administration feature that provides a uniform environment for local and remote access to Windows system components. It relies on the WMI service for local and remote access and the server message block (SMB) and Remote Procedure Call Service (RPCS) for remote access. RPCS operates over port 135. An adversary can use WMI to interact with local and remote systems and use it as a means to perform many tactic functions, such as gathering information for Discovery and remote Execution of files as part of Lateral Movement."

How could we prevent this?

- WMI is a built-in feature on Windows and is thus not advised to try disabling it. We can, however, implement network segmentation (e.g., prevent workstation-to-workstation traffic)
- The Microsoft Tiered Admin Model should be implemented in order to restrict lateral movement strategies

How could we detect this?

- Real-time detection in Windows using Sysmon:
  - Event ID 1: Look for suspicious WMI invocations
  - Event ID 3: Look for suspicious network connections (e.g., workstation to workstation)



Phase 3: Exfiltration – Technique T1041 – Exfil over C&C
From MITRE's ATT&CK framework:

"Data exfiltration is performed over the Command and Control channel. Data is encoded into the normal communications channel using the same protocol as command and control communications."

How could we prevent this?

- Not applicable: If an adversary has a C&C channel set up, they can exfiltrate data over this channel.

How could we detect this?

- In order to detect data exfiltration over the C&C channel, we could try detecting volume anomalies.

The full ATT&CK entry can be found at https://attack.mitre.org/techniques/T1041/.

Course Roadmap

- Introduction & Key Tools
- Initial Access
- Lateral Movement
- Persistence
- **Azure AD & Emulation Plans**
- Adversary Emulation Capstone

SEC699.5

Azure AD
Azure AD Structure and Management
Azure AD Hybrid Authentication
Azure AD Authentication Methods
Azure AD Conditional Access
Introduction to Azure Identities
Azure AD Security Logging

Executing emulation plans
APT-28 Emulation Plan
Exercise: APT-28 Emulation Plan
APT-34 Emulation Plan
Exercise: APT-34 Emulation Plan
Turla Emulation Plan
Exercise: Turla Emulation plan

This page intentionally left blank.

EXERCISE: MANUAL EXECUTION OF APT-28 EMULATION PLAN

Please refer to the workbook for further instructions on the exercise!

This page intentionally left blank.

Course Roadmap

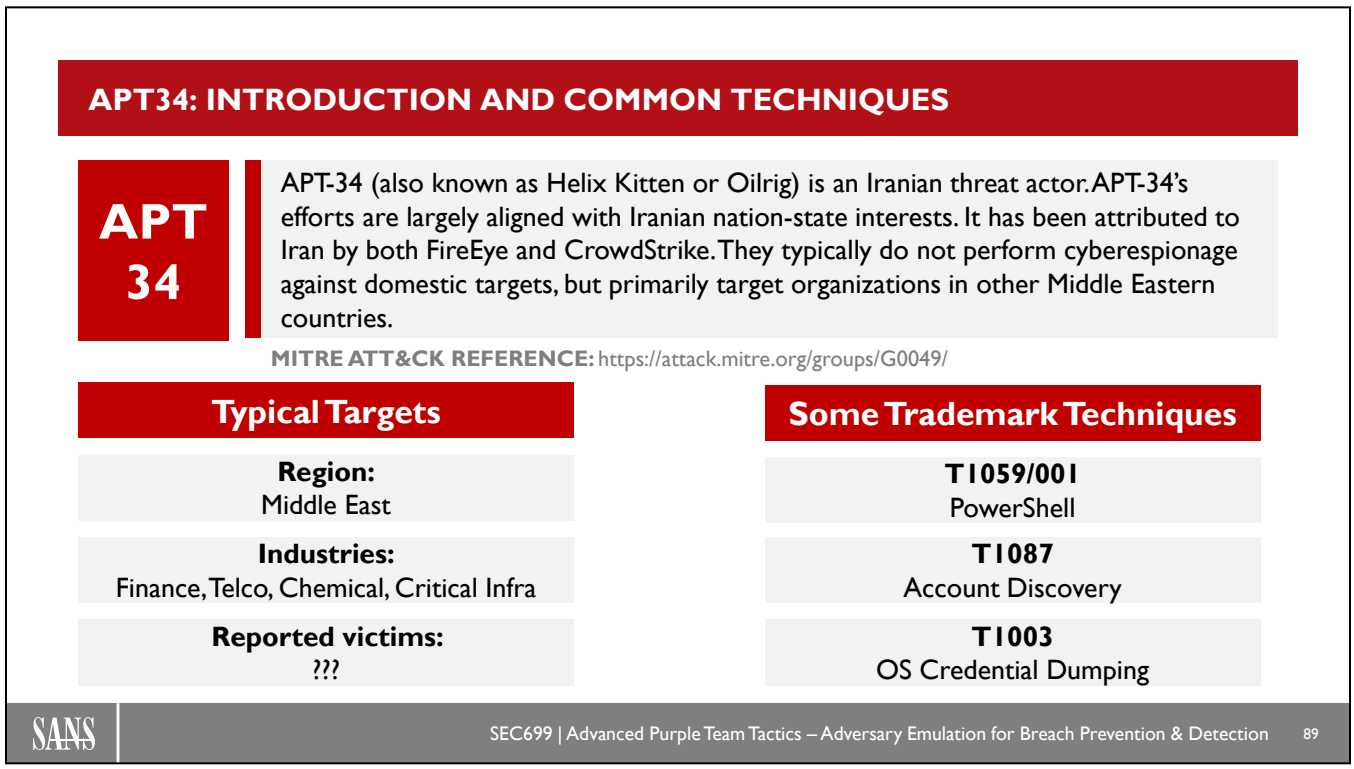
- Introduction & Key Tools
- Initial Access
- Lateral Movement
- Persistence
- **Azure AD & Emulation Plans**
- Adversary Emulation Capstone

SEC699.5

Azure AD
Azure AD Structure and Management
Azure AD Hybrid Authentication
Azure AD Authentication Methods
Azure AD Conditional Access
Introduction to Azure Identities
Azure AD Security Logging

Executing emulation plans
APT-28 Emulation Plan
Exercise: APT-28 Emulation Plan
APT-34 Emulation Plan
Exercise: APT-34 Emulation Plan
Turla Emulation Plan
Exercise: Turla Emulation plan

This page intentionally left blank.

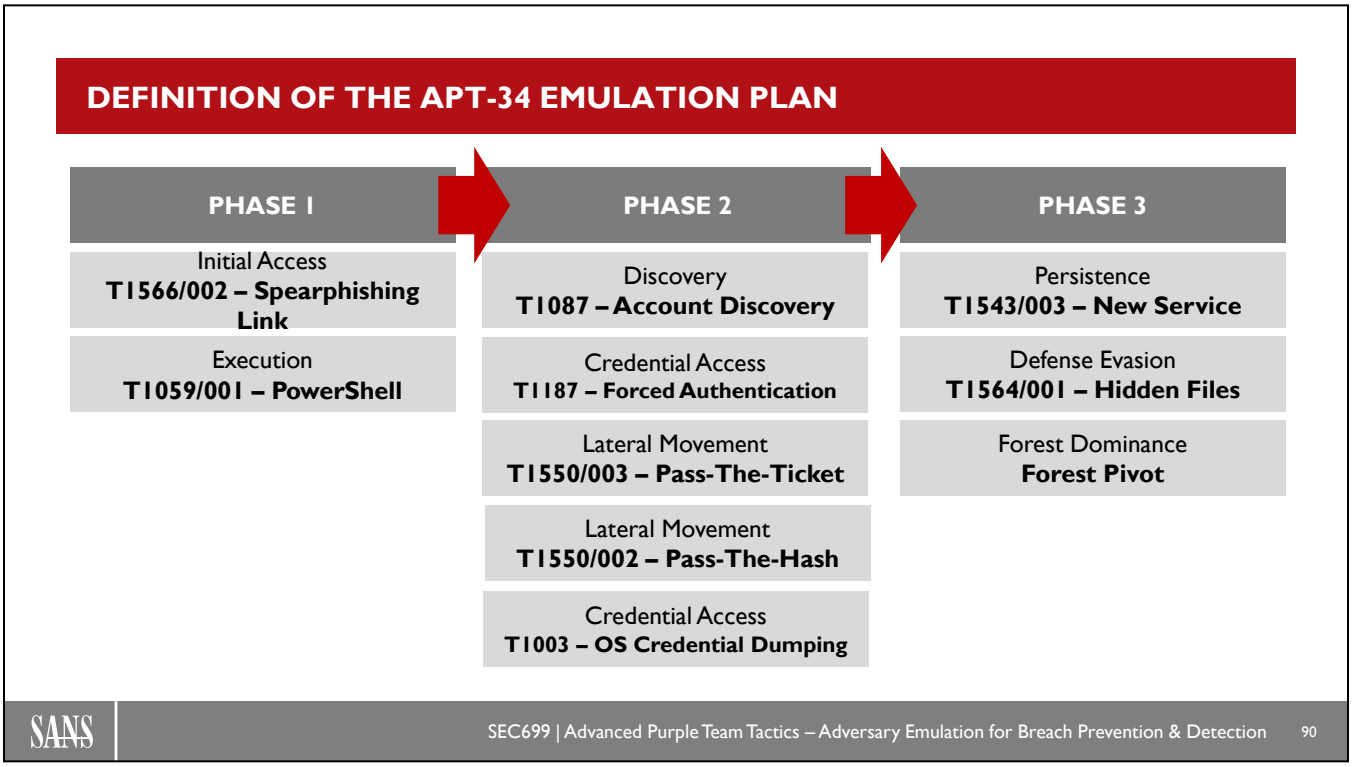


APT34: Introduction and Common Techniques
APT-34 (also known as Helix Kitten or Oilrig) is an Iranian threat actor. APT-34's efforts are largely aligned with Iranian nation-state interests. It has been attributed to Iran by both FireEye and CrowdStrike. They typically do not perform cyberespionage against domestic targets, but primarily target organizations in other Middle Eastern countries.

MITRE's ATT&CK report on the group can be found on https://attack.mitre.org/groups/G0049/.

With regards to targets, APT-34 primarily targets organizations in other Middle Eastern countries. Furthermore, the industries they target include Financials, Telecommunications, Chemical, and Critical Infrastructure. Due to the nebulous state of Iranian threat actors, it's hard to identify specific organizations that were the victim of APT-34.

References:
https://css.ethz.ch/content/dam/ethz/special-interest/gess/cis/center-for-securities-studies/pdfs/20190507_MB_HS_IRN%20V1_rev.pdf
https://www.fireeye.com/current-threats/apt-groups.html
https://www.crowdstrike.com/blog/meet-crowdstrikes-adversary-of-the-month-for-november-helix-kitten/



Definition of the APT-34 Emulation Plan

We will now draft an emulation plan that mimics techniques known to be abused by APT-34. We will define the following steps as part of the plan:

Phase 1

- Initial Access – T1566/002 – Spearphishing Link
- Execution – T1059/001 – PowerShell

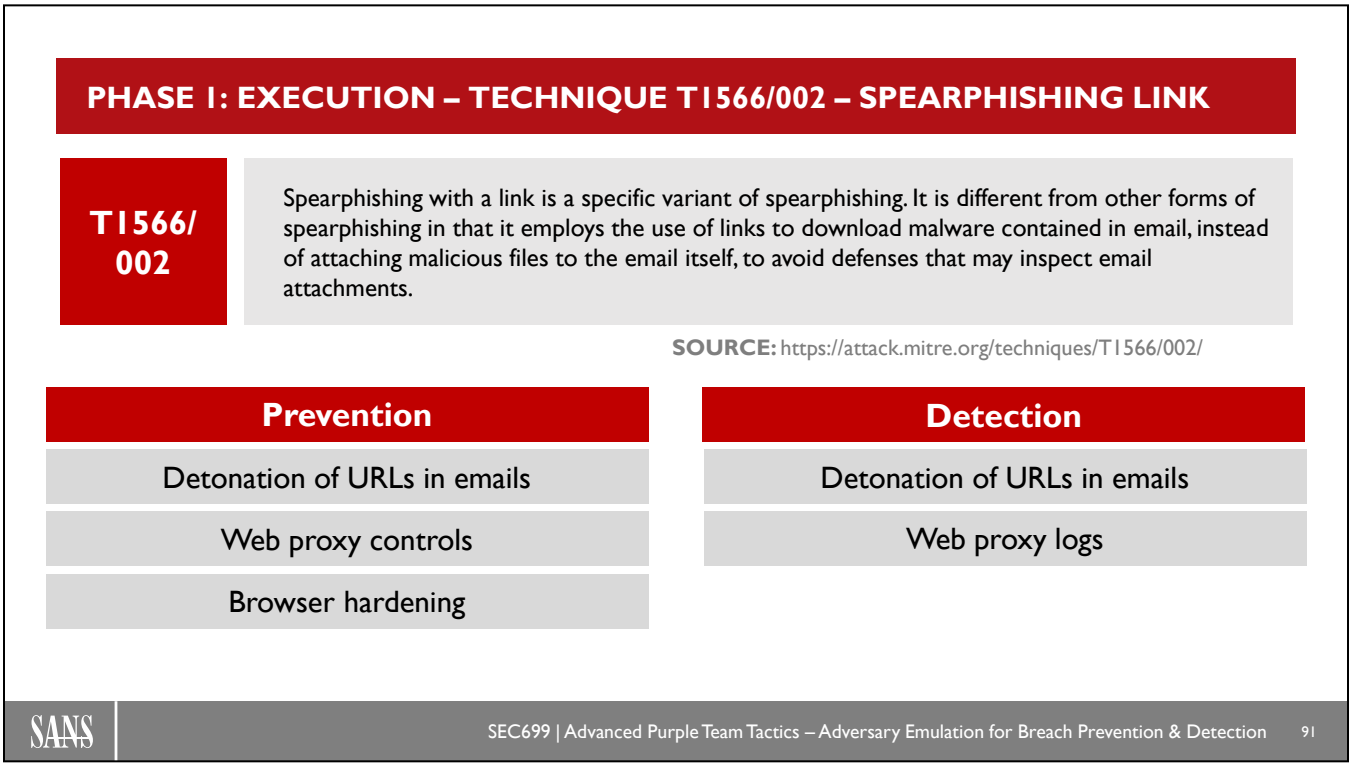
Phase 2

- Discovery – T1087 – Account Discovery
- Credential Access – T1187 – Forced Authentication
- Lateral Movement – T1550/003 – Pass-The-Ticket
- Lateral Movement – T1550/002 – Pass-The-Hash
- Credential Access – T1003 – OS Credential Dumping

Phase 3

- Persistence – T1543/003 – New Service
- Defense Evasion – T1564/001 – Hidden Files
- Forest Dominance – Forest Pivot

We have added short technique descriptions in the next few slides (extracted from MITRE ATT&CK). We will further prepare and execute the emulation plan in our Workbook!



Phase 1: Execution – Technique T1566/002 – Spearphishing Link

From MITRE's ATT&CK framework:

"Spearphishing with a link is a specific variant of spearphishing. It is different from other forms of spearphishing in that it employs the use of links to download malware contained in email, instead of attaching malicious files to the email itself, to avoid defenses that may inspect email attachments."

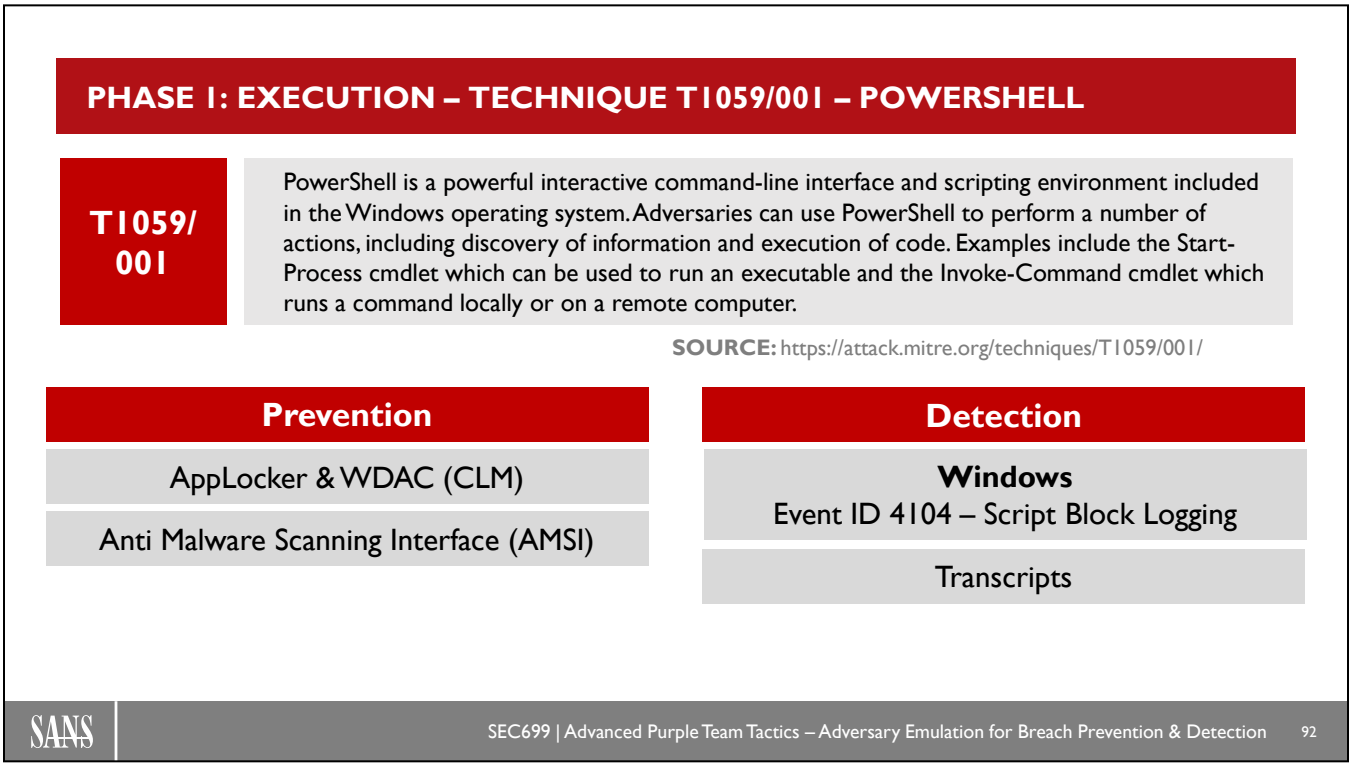
How could we prevent this?

- Detonation of URLs in emails
- Web proxy controls

How could we detect this?

- Detonation of URLs in emails
- Web proxy logs

The full ATT&CK entry can be found at https://attack.mitre.org/techniques/T1192/.



Phase 1: Execution – Technique T1059/001 – PowerShell

From MITRE's ATT&CK framework:

"PowerShell is a powerful interactive command-line interface and scripting environment included in the Windows operating system. Adversaries can use PowerShell to perform a number of actions, including discovery of information and execution of code. Examples include the Start-Process cmdlet which can be used to run an executable and the Invoke-Command cmdlet which runs a command locally or on a remote computer."

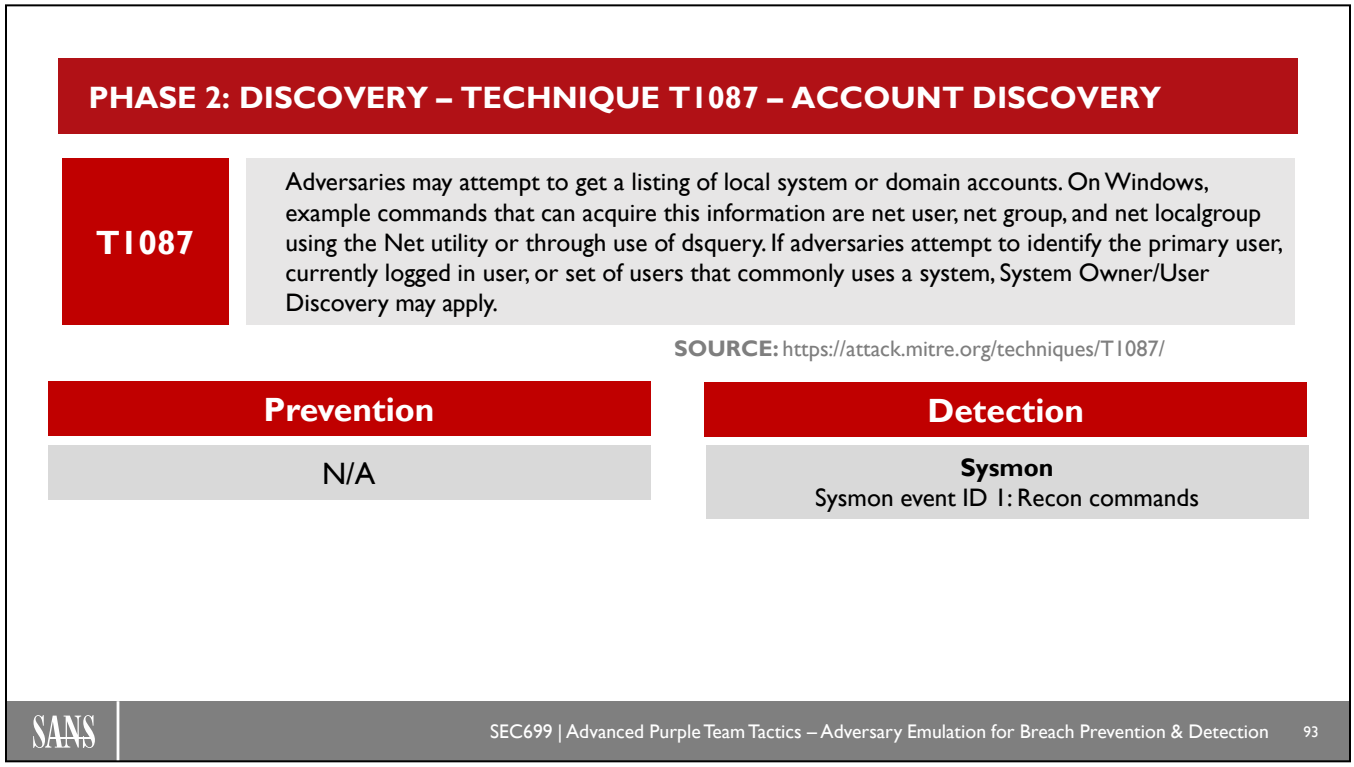
How could we prevent this?

- AppLocker and Windows Defender Application Control (to enforce PowerShell Constrained Language Mode)
- The Anti Malware Scanning Interface (AMSI) feature

How could we detect this?

- The Windows event ID 4104 (PowerShell Script Block Logging)
- PowerShell transcript logs

The full ATT&CK entry can be found at https://attack.mitre.org/techniques/T1059/001/.



Phase 2: Discovery – Technique T1087 – Account Discovery

From MITRE's ATT&CK framework:

"Adversaries may attempt to get a listing of local system or domain accounts. On Windows, example commands that can acquire this information are net user, net group, and net localgroup using the Net utility or through use of dsquery. If adversaries attempt to identify the primary user, currently logged in user, or set of users that commonly uses a system, System Owner/User Discovery may apply."

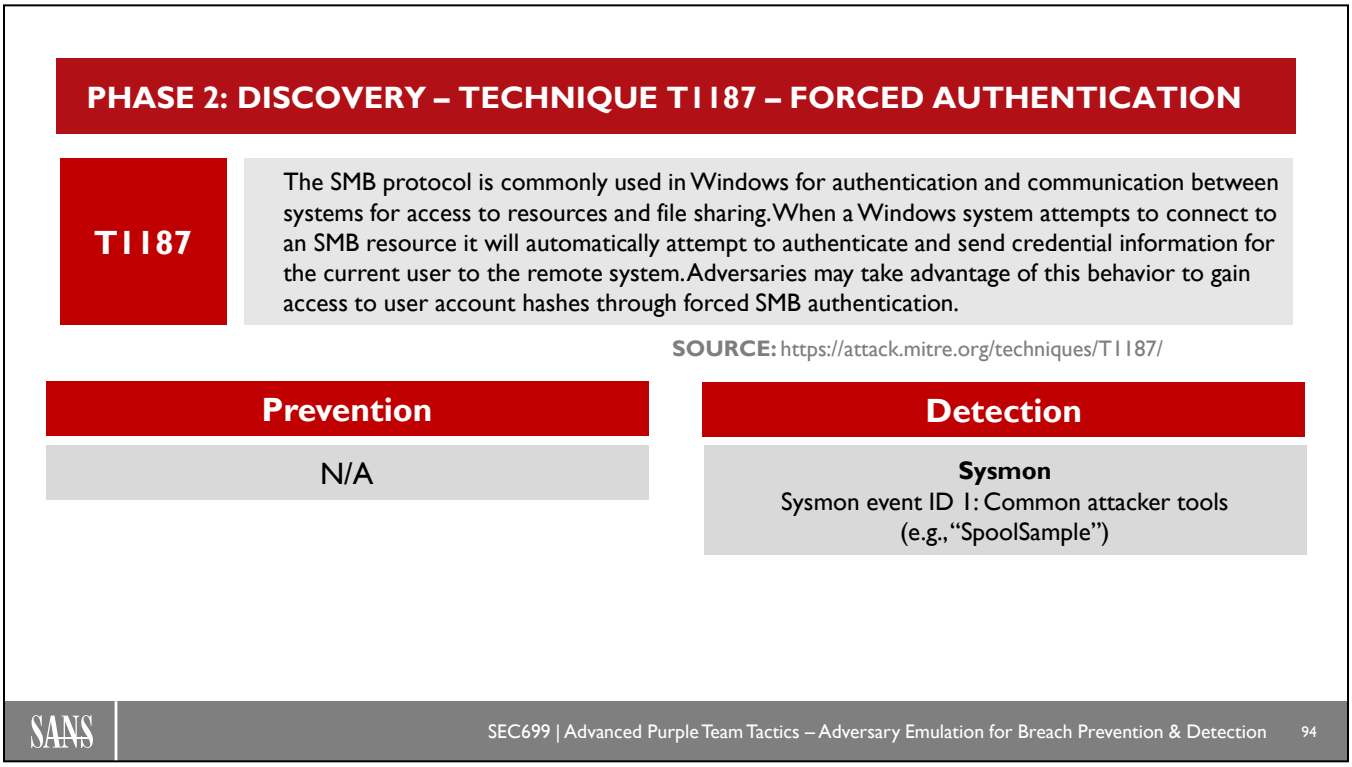
How could we prevent this?

- Not Applicable: Reconnaissance of the Windows environment is not something that can be blocked

How could we detect this?

- Sysmon event ID 1: Using Sysmon process creation logging, we could look for typical commands used in domain enumeration

The full ATT&CK entry can be found at https://attack.mitre.org/techniques/T1087/.



Phase 2: Discovery – Technique T1187 – Forced Authentication

From MITRE's ATT&CK framework:

"The SMB protocol is commonly used in Windows for authentication and communication between systems for access to resources and file sharing. When a Windows system attempts to connect to an SMB resource it will automatically attempt to authenticate and send credential information for the current user to the remote system. Adversaries may take advantage of this behavior to gain access to user account hashes through forced SMB authentication."

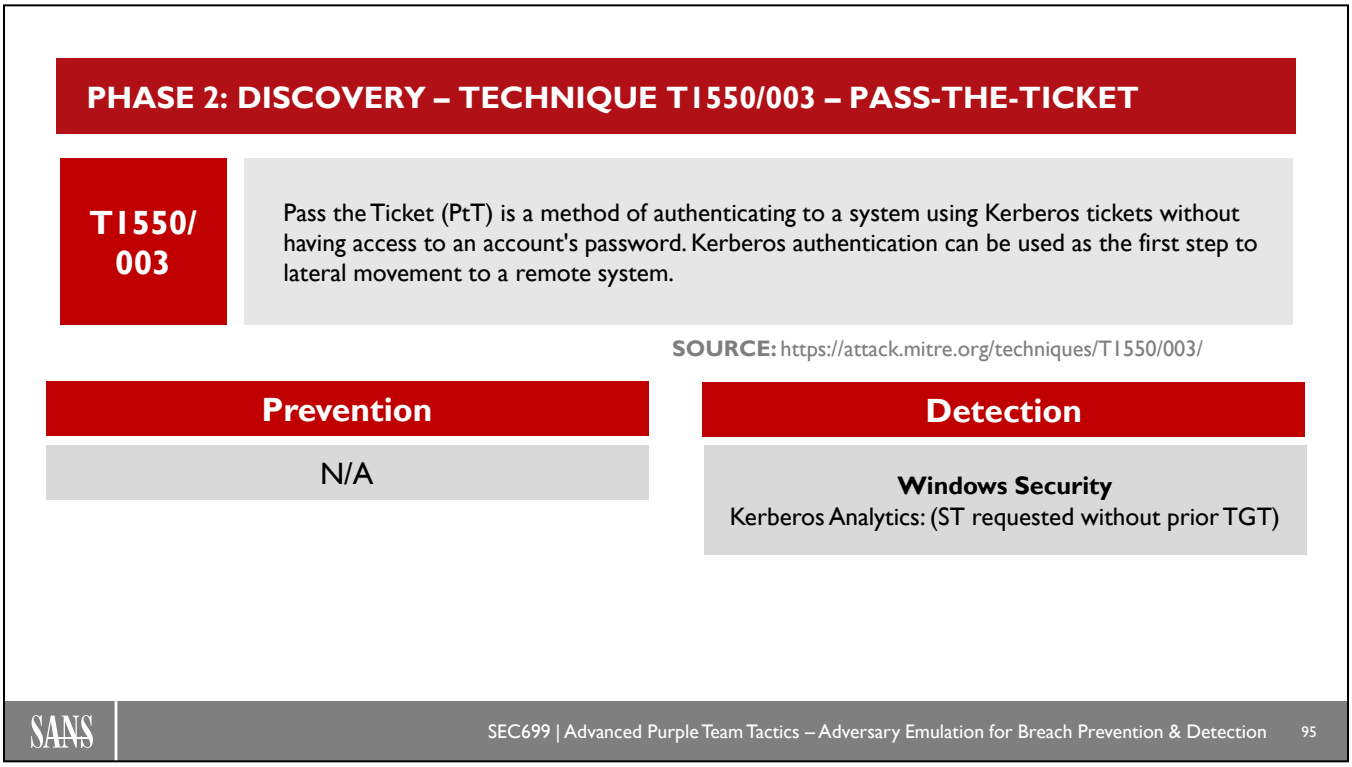
How could we prevent this?

- Not Applicable: This is a built-in feature of the Windows OS

How could we detect this?

- Sysmon event ID 1: Using Sysmon process creation logging, we could look for typical attacker tools being used (such as SpoolSample).

The full ATT&CK entry can be found at https://attack.mitre.org/techniques/T1187/.



Phase 2: Discovery – Technique T1550/003 – Pass-The-Ticket

From MITRE's ATT&CK framework:

"Pass the ticket (PtT) is a method of authenticating to a system using Kerberos tickets without having access to an account's password. Kerberos authentication can be used as the first step to lateral movement to a remote system."

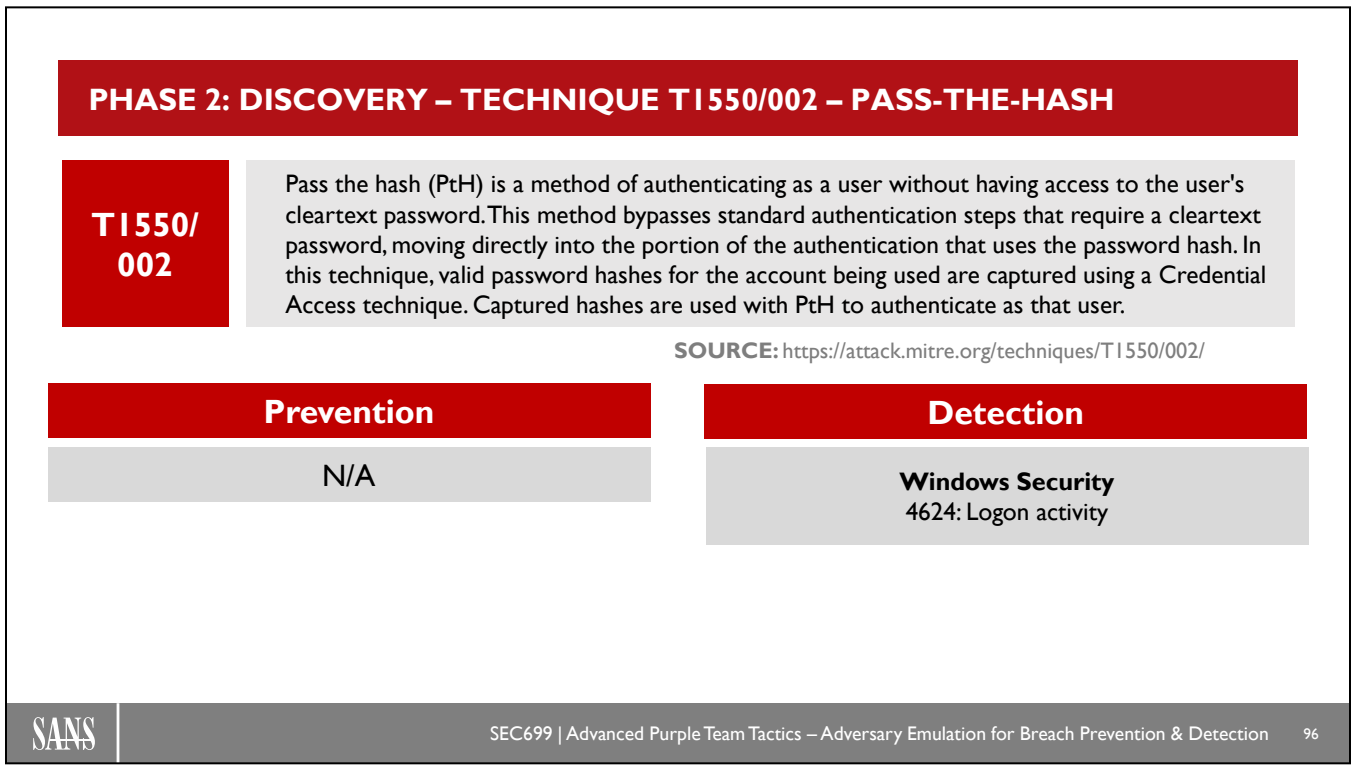
How could we prevent this?

- Not Applicable: This is a built-in feature of Kerberos

How could we detect this?

- Kerberos Analytics: Theoretically, it would be possible to identify service ticket request if no prior TGT was requested. This would, however, mean we would need some form of "stateful" Kerberos. Microsoft Defender for Identity (previous called Microsoft ATA or Azure ATP) can be useful for this specific use case, but it is a commercial tool.

The full ATT&CK entry can be found at https://attack.mitre.org/techniques/T1097/.



Phase 2: Discovery – Technique T1550/002 – Pass-The-Hash

From MITRE's ATT&CK framework:

"Pass the hash (PtH) is a method of authenticating as a user without having access to the user's cleartext password. This method bypasses standard authentication steps that require a cleartext password, moving directly into the portion of the authentication that uses the password hash. In this technique, valid password hashes for the account being used are captured using a Credential Access technique. Captured hashes are used with PtH to authenticate as that user."

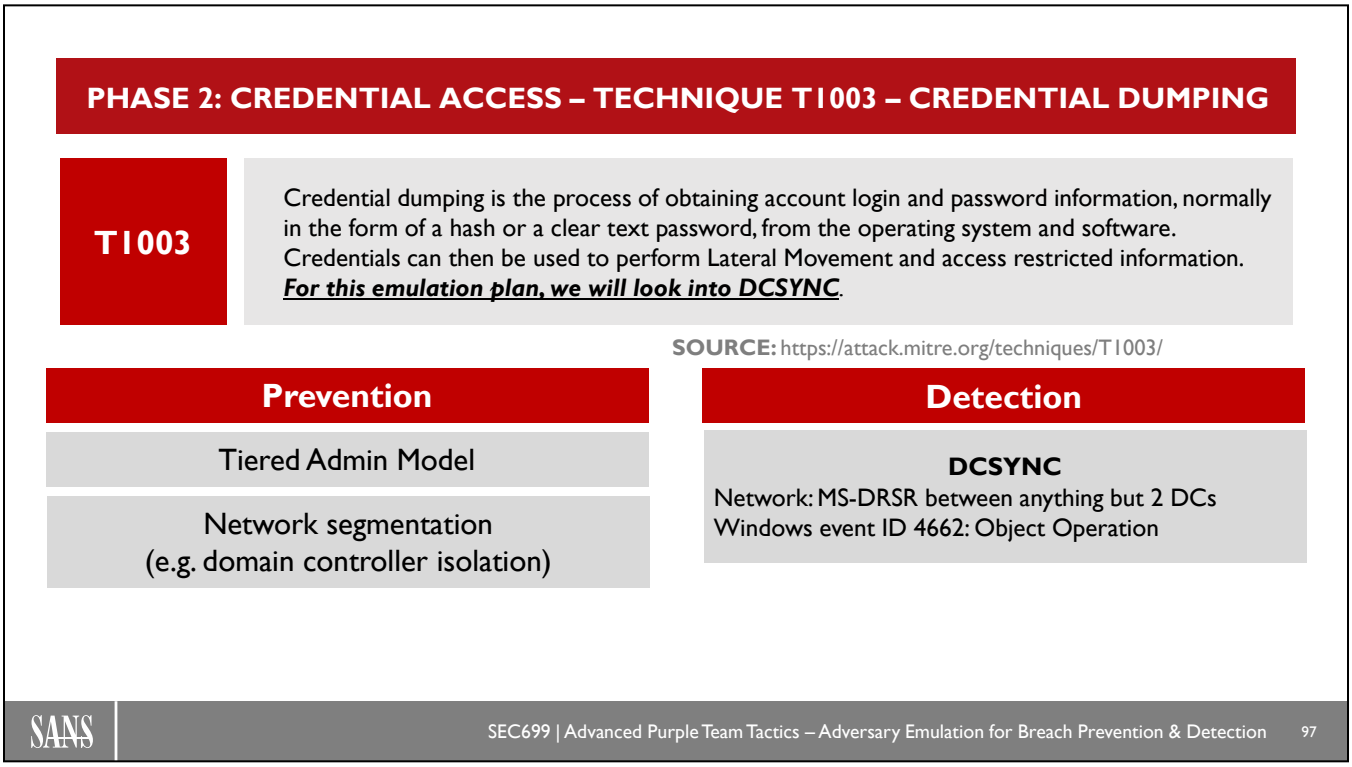
How could we prevent this?

- Not Applicable: This is a native feature in Windows

How could we detect this?

- Windows Security logs: We can review event ID 4624 (Successful login) for traces of this behavior. Please refer to the example SIGMA rules in the next slides!

The full ATT&CK entry can be found at https://attack.mitre.org/techniques/T1075/.



Phase 2: Credential Access – Technique T1003 – Credential Dumping

From MITRE's ATT&CK framework:

"Credential dumping is the process of obtaining account login and password information, normally in the form of a hash or a clear text password, from the operating system and software. Credentials can then be used to perform Lateral Movement and access restricted information."

For this emulation plan, we will look into the DCYNC attack. Other credential dumping techniques will be discussed later.

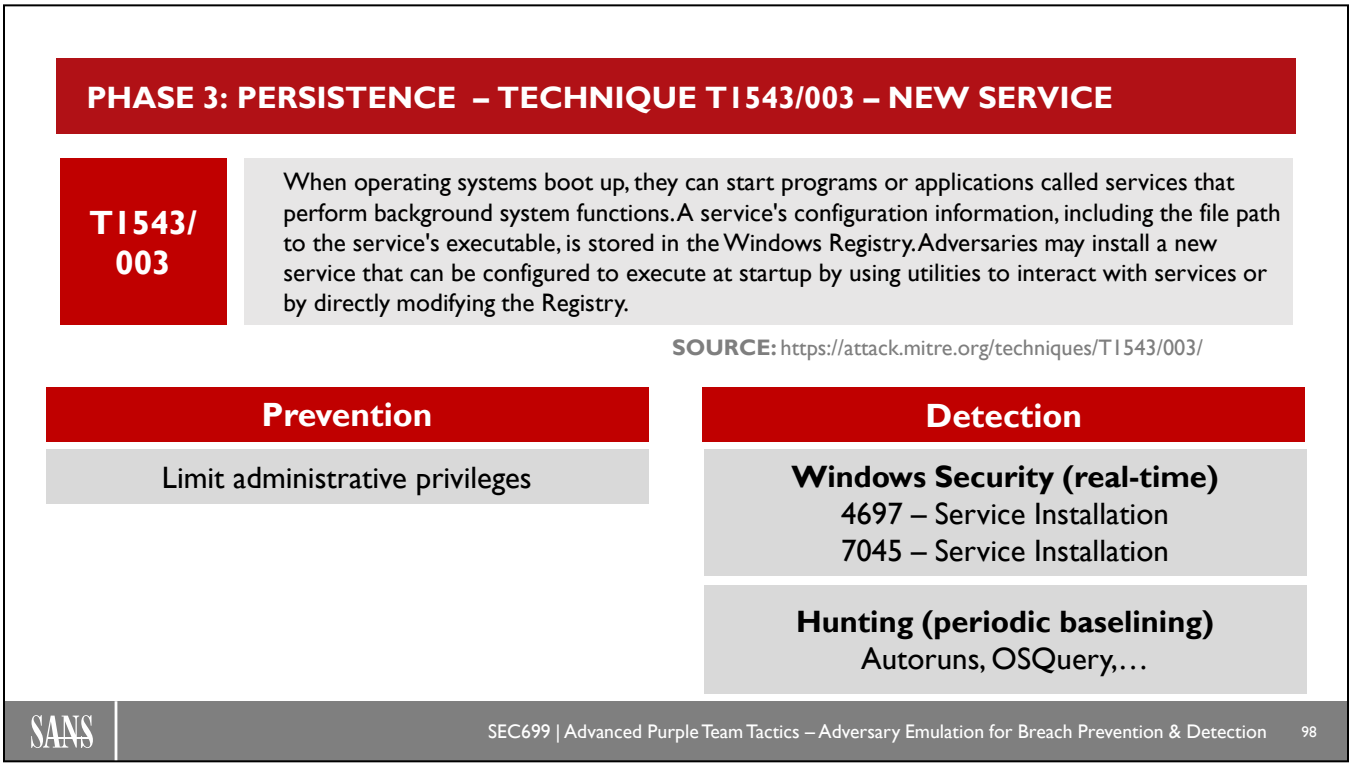
How could we prevent this?

- The Microsoft Tiered Admin Model should be implemented in order to prevent people from getting domain administrative privileges
- The network should be segmented to isolate domain controllers in their own proper network zone

How could we detect this?

- To detect DCSYNC on the network, we could look for MS-DRSR traffic between anything but 2 domain controllers. This would require proper isolation and monitoring of domain controllers
- Using the Windows event logs, we can look for specific values in event ID 4662. Please refer to the next slide for an example of how this can be done.

The full ATT&CK entry can be found at https://attack.mitre.org/techniques/T1003/.



Phase 3: Persistence – Technique T1543/003 – New Service

From MITRE's ATT&CK framework:

"When operating systems boot up, they can start programs or applications called services that perform background system functions. A service's configuration information, including the file path to the service's executable, is stored in the Windows Registry. Adversaries may install a new service that can be configured to execute at startup by using utilities to interact with services or by directly modifying the Registry."

How could we prevent this?

- Limit administrative privileges

How could we detect this?

- Real-time detection in Windows
  - Look for event ID 4697 – Service Installation
  - Look for event ID 7045 – Service Installation
- Hunting
  - Periodic baselining of scheduled tasks (Autoruns, OSQuery,…)

The full ATT&CK entry can be found at https://attack.mitre.org/techniques/T1050/.

PHASE 3: DEFENSE EVASION – TECHNIQUE T1564/001 – HIDDEN FILES

T1564/001

To prevent normal users from accidentally changing special files on a system, most operating systems have the concept of a 'hidden' file. These files don't show up when a user browses the file system with a GUI or when using normal commands on the command line. Adversaries can use this to their advantage to hide files and folders anywhere on the system for persistence and evading a typical user or system analysis that does not incorporate investigation of hidden files.

SOURCE: https://attack.mitre.org/techniques/T1564/001/

Prevention	Detection
N/A	**Sysmon** Sysmon event ID 1: "attrib +h"

Phase 3: Defense Evasion – Technique T1564/001 – Hidden Files

From MITRE's ATT&CK framework:

"To prevent normal users from accidentally changing special files on a system, most operating systems have the concept of a 'hidden' file. These files don't show up when a user browses the file system with a GUI or when using normal commands on the command line. Adversaries can use this to their advantage to hide files and folders anywhere on the system for persistence and evading a typical user or system analysis that does not incorporate investigation of hidden files."

How could we prevent this?

- As this is a built-in feature of the OS that does not require administrative privileges, it cannot be prevented

How could we detect this?

- Using Sysmon, we could search for the following:
  - Event ID 1 (Process Creation): Identify executions of "attrib +h"

https://github.com/Neo23x0/sigma/blob/master/rules/windows/process_creation/win_attrib_hiding_files.yml

The full ATT&CK entry can be found at https://attack.mitre.org/techniques/T1158/.

Course Roadmap

- Introduction & Key Tools
- Initial Access
- Lateral Movement
- Persistence
- **Azure AD & Emulation Plans**
- Adversary Emulation Capstone

SEC699.5

Azure AD
Azure AD Structure and Management
Azure AD Hybrid Authentication
Azure AD Authentication Methods
Azure AD Conditional Access
Introduction to Azure Identities
Azure AD Security Logging

Executing emulation plans
APT-28 Emulation Plan
Exercise: APT-28 Emulation Plan
APT-34 Emulation Plan
Exercise: APT-34 Emulation Plan
Turla Emulation Plan
Exercise: Turla Emulation plan

This page intentionally left blank.

EXERCISE: MANUAL EXECUTION OF APT-34 EMULATION PLAN

Please refer to the workbook for further instructions on the exercise!

This page intentionally left blank.

Course Roadmap

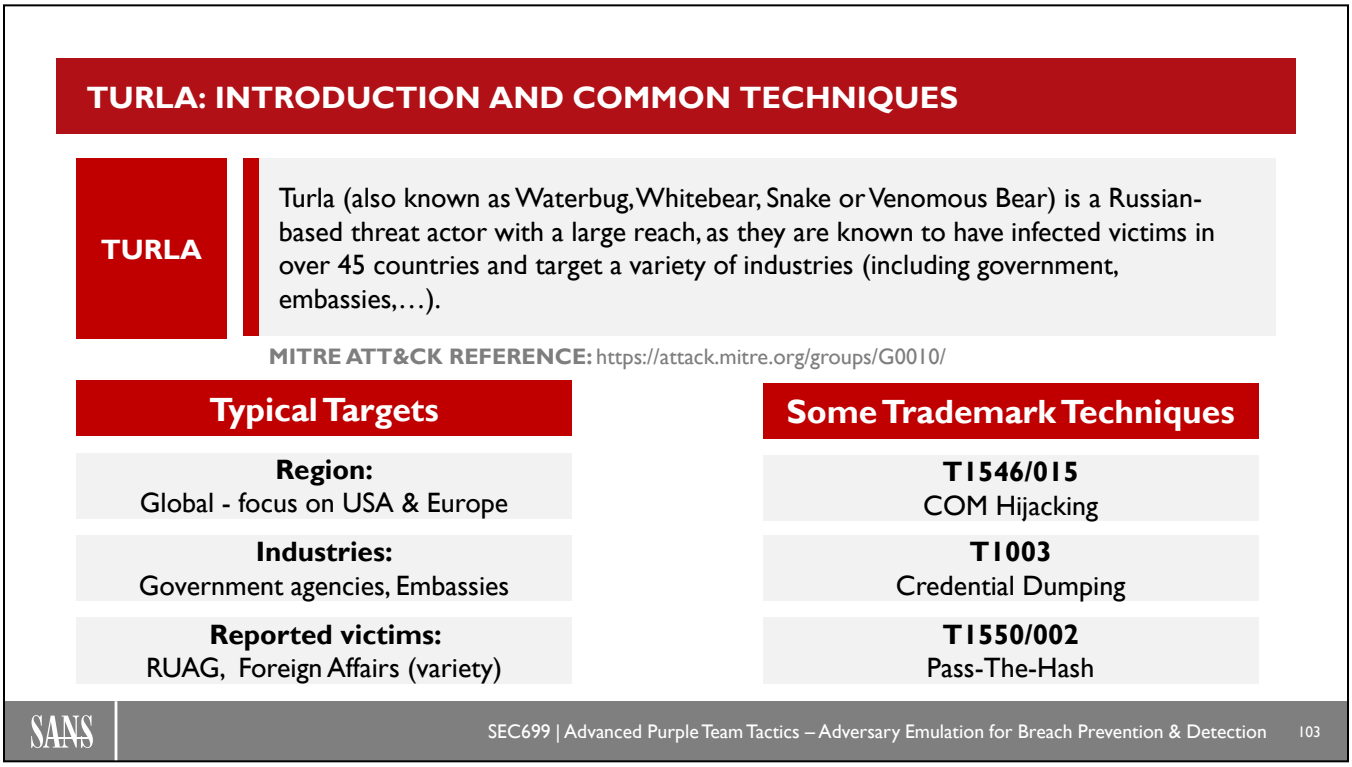
- Introduction & Key Tools
- Initial Access
- Lateral Movement
- Persistence
- **Azure AD & Emulation Plans**
- Adversary Emulation Capstone

SEC699.5

Azure AD
Azure AD Structure and Management
Azure AD Hybrid Authentication
Azure AD Authentication Methods
Azure AD Conditional Access
Introduction to Azure Identities
Azure AD Security Logging

Executing emulation plans
APT-28 Emulation Plan
Exercise: APT-28 Emulation Plan
APT-34 Emulation Plan
Exercise: APT-34 Emulation Plan
Turla Emulation Plan
Exercise: Turla Emulation plan

This page intentionally left blank.



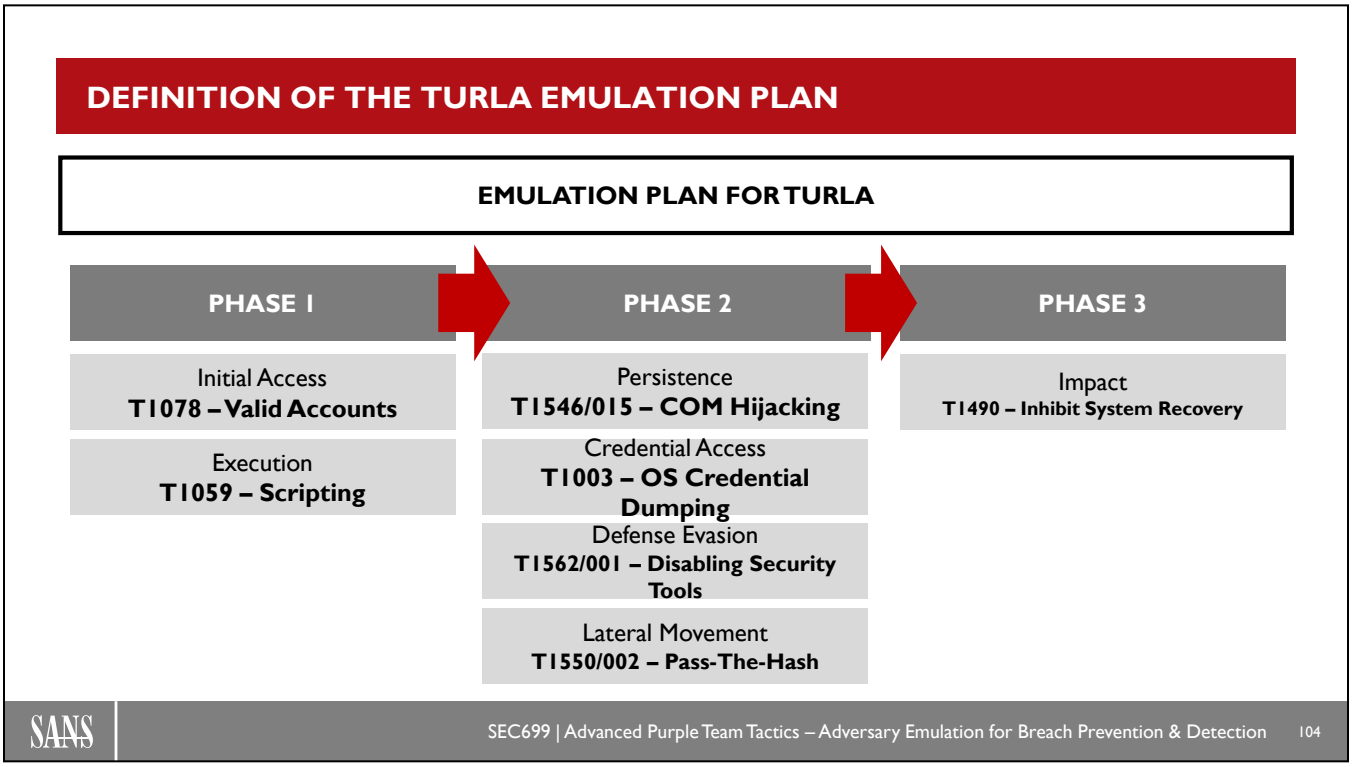
APT28: Introduction and Common Techniques

Turla (also known as Waterbug, Whitebear, Snake or Venomous Bear) is a Russian-based threat actor with a large reach, as they are known to have infected victims in over 45 countries and target a variety of industries (including government, embassies,…).

MITRE's ATT&CK report on the group can be found on https://attack.mitre.org/groups/G0010/.

With regards to targets, Turla is known to have a global approach, but a specific focus on USA and (Western) Europe. They mostly target government agencies and embassies. Reported victims have included RUAG and the Foreign Affairs departments of a number of European countries.

Further references:
https://malpedia.caad.fkie.fraunhofer.de/actor/turla_group
https://www.crowdstrike.com/blog/meet-crowdstrikes-adversary-of-the-month-for-march-venomous-bear/



Definition of the Turla Emulation Plan

We will now draft an emulation plan that mimics techniques known to be abused by Turla. We will define the following steps as part of the plan:

Phase 1

- Initial Access – T1078 – Valid Accounts
- Execution – T1059 – Scripting

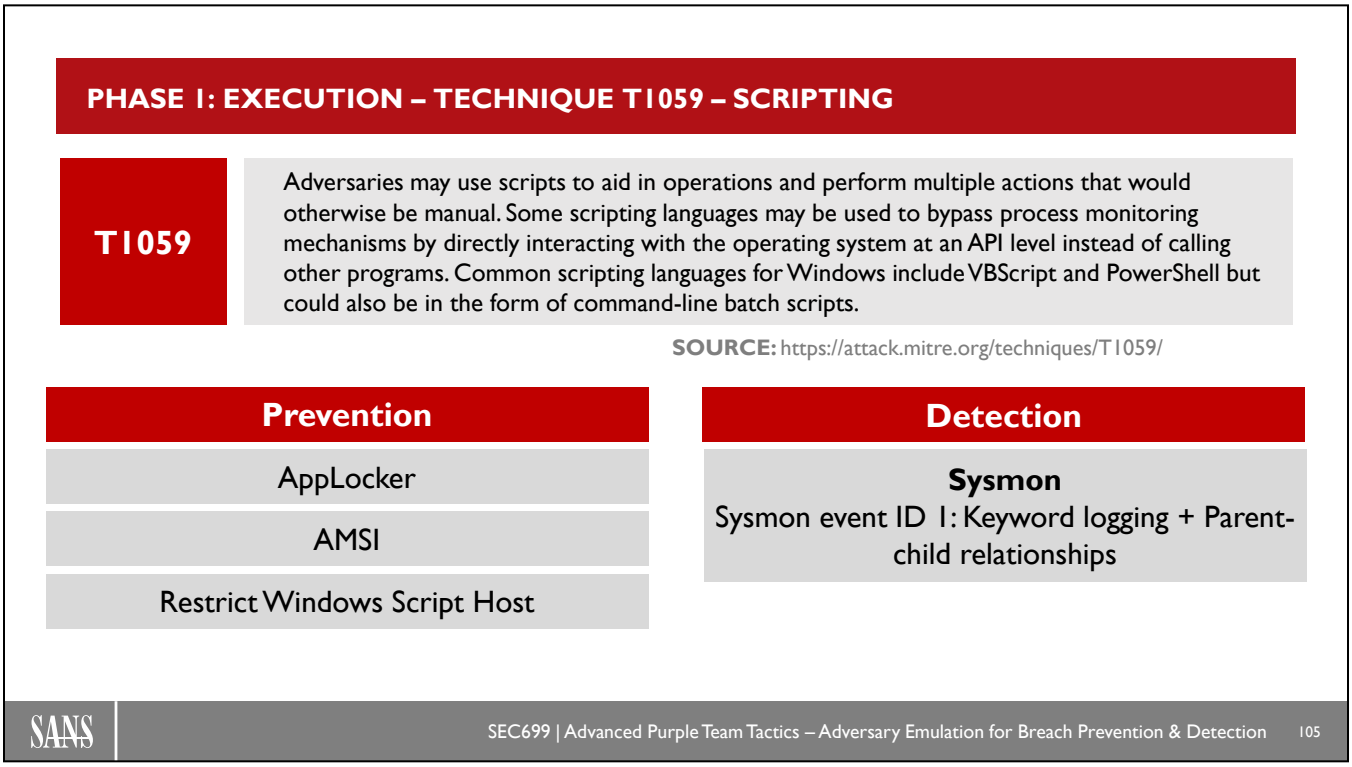
Phase 2

- Persistence - T1546/015 – COM Hijacking
- Credential Access - T1003 – OS Credential Dumping
- Defense Evasion - T1562/001 – Disabling Security Tools
- Lateral Movement – T1550/002 – Pass-The-Hash

Phase 3

- Impact – T1490 – Inhibit System Recovery

We have added short technique descriptions in the next few slides (extracted from MITRE ATT&CK). We will further prepare and execute the emulation plan in our Workbook!



Phase 1: Execution – Technique T1059 – Scripting

From MITRE's ATT&CK framework:

"Adversaries may use scripts to aid in operations and perform multiple actions that would otherwise be manual. Some scripting languages may be used to bypass process monitoring mechanisms by directly interacting with the operating system at an API level instead of calling other programs. Common scripting languages for Windows include VBScript and PowerShell but could also be in the form of command-line batch scripts."

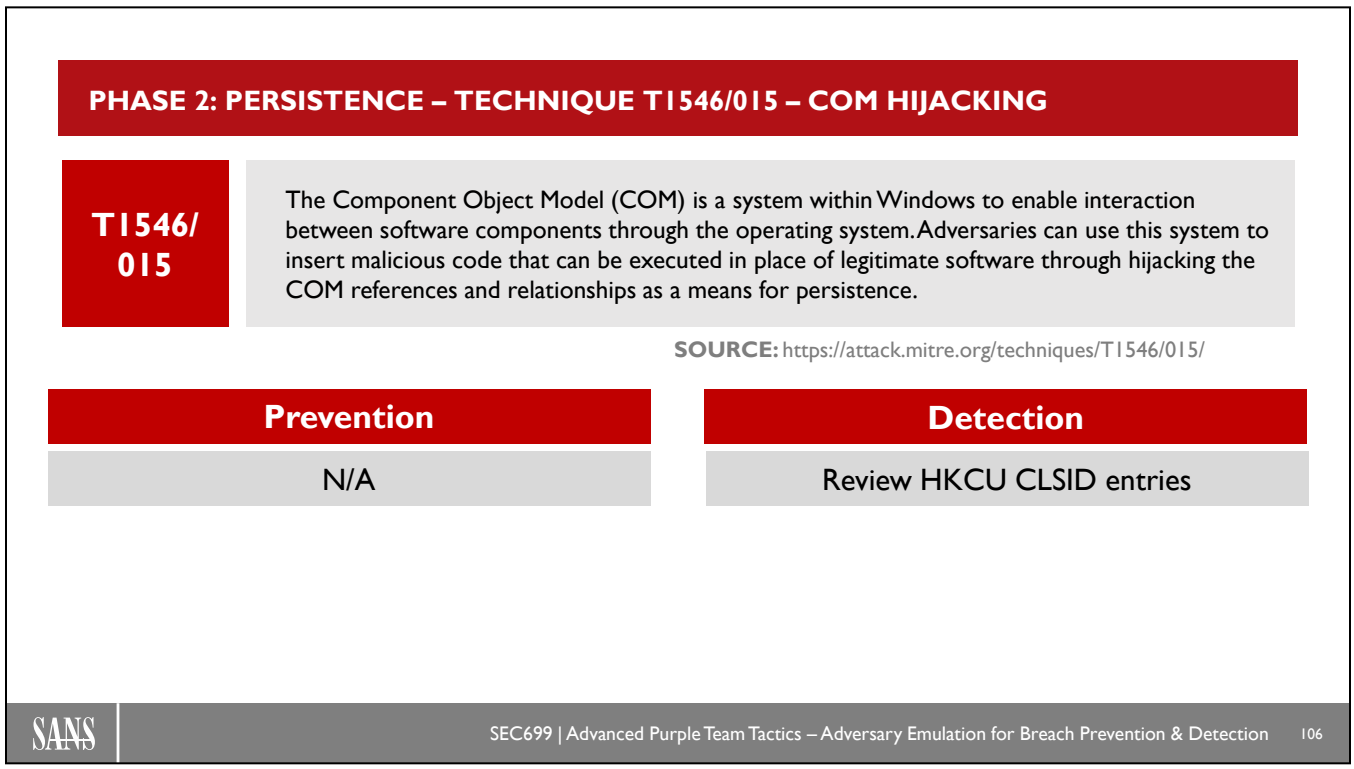
How could we prevent this?

- AppLocker: Application Whitelisting can be used to limit script execution
- AMSI: The Anti Malware Scanning Interface can help detect and block malicious scripts
- Windows Script Host restrictions: The WSH (Windows Script Host) can be disabled, or the host process can be blocked from creating outbound network connections (typical downloader behavior).

How could we detect this?

- Sysmon event ID 1 (Process Creation): Look for typical scripting keywords and parent-child relationships

The full ATT&CK entry can be found at https://attack.mitre.org/techniques/T1064/.



Phase 2: Persistence – Technique T1546/015 - COM Hijacking

From MITRE's ATT&CK framework:

"The Component Object Model (COM) is a system within Windows to enable interaction between software components through the operating system. Adversaries can use this system to insert malicious code that can be executed in place of legitimate software through hijacking the COM references and relationships as a means for persistence."

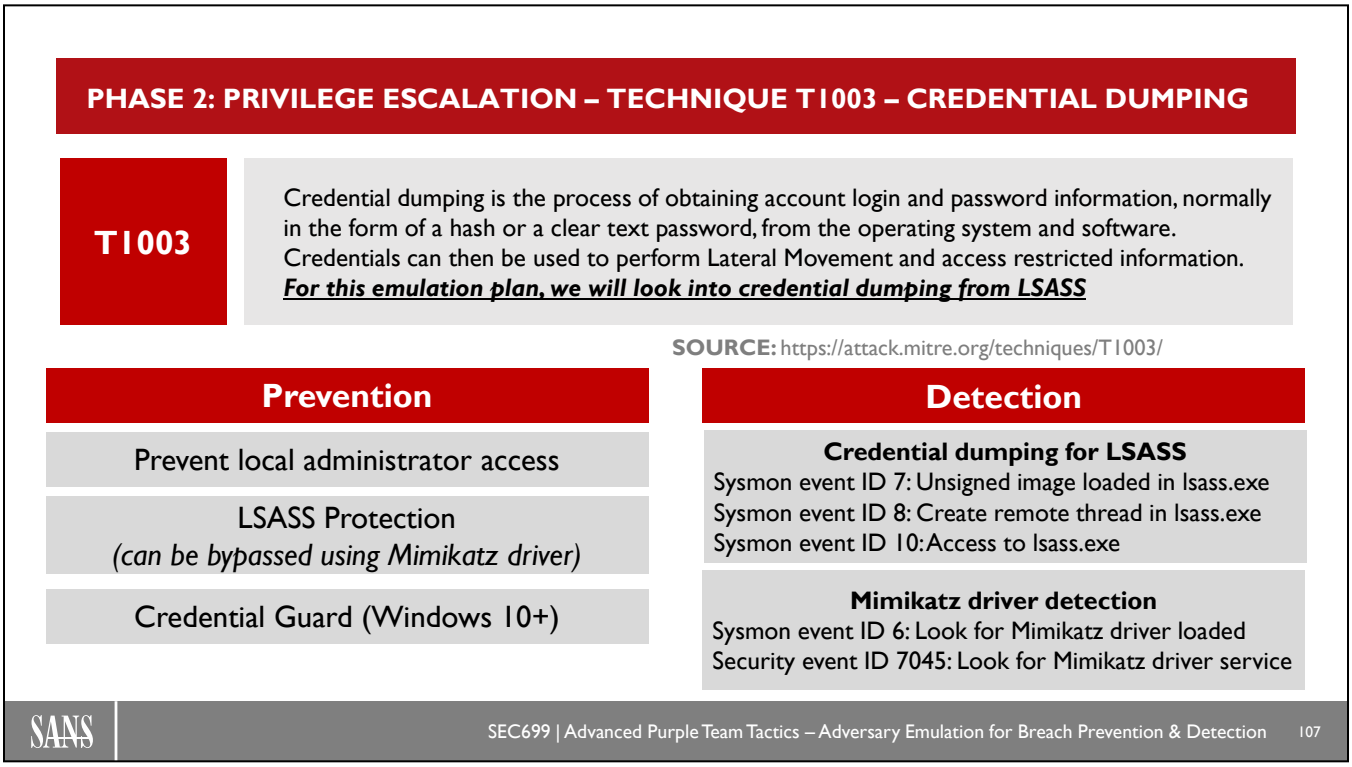
How could we prevent this?

- Not Applicable: COM objects are a default feature of Microsoft Windows

How could we detect this?

- Detecting COM object hijacking is not simple, as it very much relies on built-in Windows functions. That being said, HKCU CLSID entries are rather rare and could be considered exceptions

The full ATT&CK entry can be found at https://attack.mitre.org/techniques/T1122/.



Phase 2: Privilege Escalation – Technique T1003 – Credential Dumping

From MITRE's ATT&CK framework:

"Credential dumping is the process of obtaining account login and password information, normally in the form of a hash or a clear text password, from the operating system and software. Credentials can then be used to perform Lateral Movement and access restricted information."

For this emulation plan, we will look into credential dumping from LSASS. Other credential dumping techniques will be discussed later.

How could we prevent this?

- This technique requires the adversary to have local administrator privileges, so a simple technique to prevent this is to ensure adversaries do not obtain such level of access.
- LSASS protection (As of Windows 8). Note that this protection can be bypassed by installing a device driver (which Mimikatz implements).
- Credential Guard (As of Windows 10)

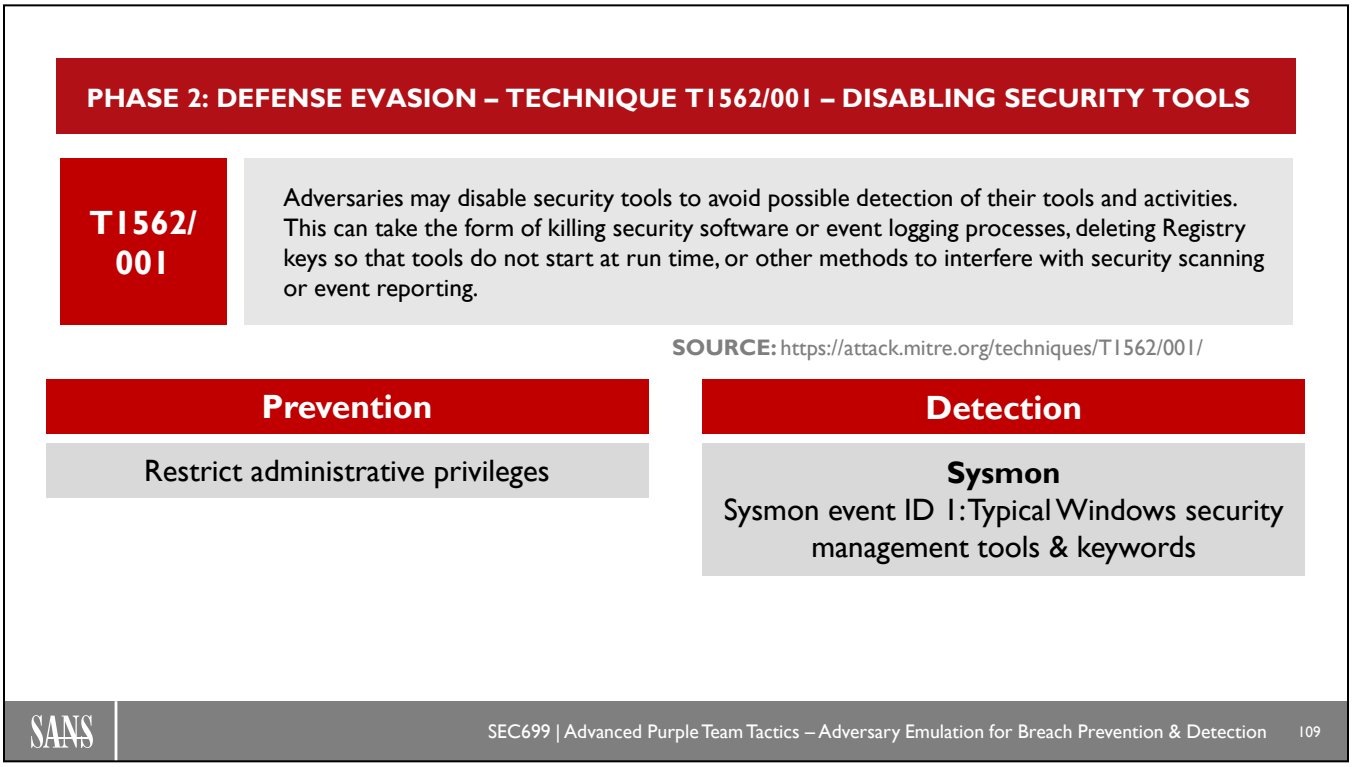
How could we detect this?

- To detect credential dumping for LSASS, we could use the following Sysmon events:
  - Event ID 7 (Image Loaded): Look for unsigned images that are loaded in lsass.exe
  - Event ID 8 (Create Remote Thread): Look for remote threads created in lsass.exe
  - Event ID 10 (Process Access): Look for access to lsass.exe

- Should LSASS protection be configured, the adversary could bypass it by deploying the Mimikatz driver. This can be detected by looking for:
  - Sysmon event ID 6 (Driver Loaded): Look for the Mimikatz driver (mimidrv.sys) being loaded
  - Windows security event ID 7045 (Service Installation): Look for the Mimikatz driver being installed

A very interesting presentation on detection of this technique can be found at https://2017.zeronights.org/wp-content/uploads/materials/ZN17_Kheirkhabarov_Hunting_for_Credentials_Dumping_in_Windows_Environment.pdf

The full ATT&CK entry can be found at https://attack.mitre.org/techniques/T1003/.



Phase 2: Defense Evasion – Technique T1562/001 – Disabling Security Tools

From MITRE's ATT&CK framework:

"Adversaries may disable security tools to avoid possible detection of their tools and activities. This can take the form of killing security software or event logging processes, deleting Registry keys so that tools do not start at run time, or other methods to interfere with security scanning or event reporting."

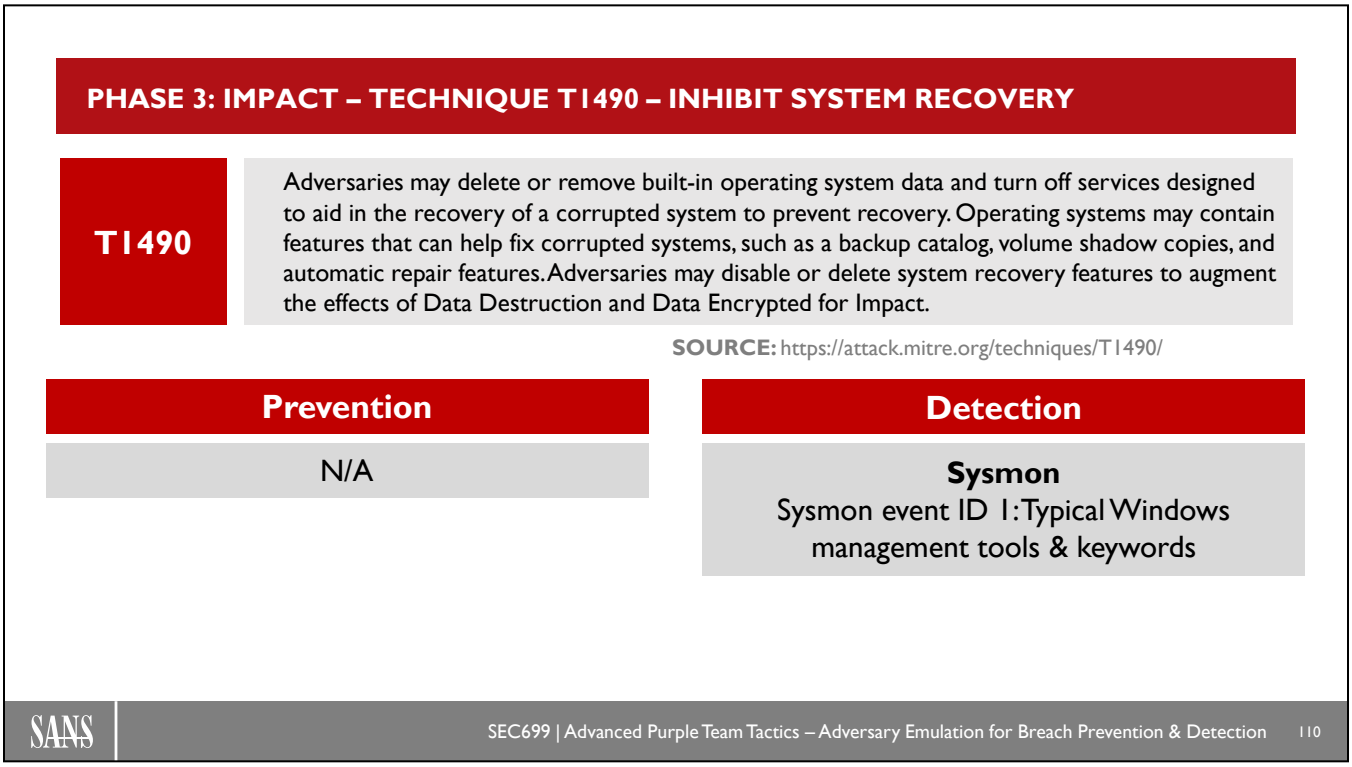
How could we prevent this?

- In order to disable / reconfigure security software such as Antivirus or firewall software, administrative privileges are typically required. A means of prevention is thus to restrict administrative privileges.

How could we detect this?

- To detect this behavior, we could use the following Sysmon events:
  - Event ID 1 (Process Creation): Look for typical Windows security tools & keywords

The full ATT&CK entry can be found at https://attack.mitre.org/techniques/T1089/.



Phase 3: Impact – Technique T1490 – Inhibit System Recovery

From MITRE's ATT&CK framework:

"Adversaries may delete or remove built-in operating system data and turn off services designed to aid in the recovery of a corrupted system to prevent recovery. Operating systems may contain features that can help fix corrupted systems, such as a backup catalog, volume shadow copies, and automatic repair features. Adversaries may disable or delete system recovery features to augment the effects of Data Destruction and Data Encrypted for Impact."

How could we prevent this?

- Not applicable: Adversaries use built-in tools and features to achieve this goal

How could we detect this?

- To detect this behavior, we could use the following Sysmon events:
  - Event ID 1 (Process Creation): Look for typical Windows management tools & keywords

The full ATT&CK entry can be found at https://attack.mitre.org/techniques/T1490/.

Course Roadmap

- Introduction & Key Tools
- Initial Access
- Lateral Movement
- Persistence
- **Azure AD & Emulation Plans**
- Adversary Emulation Capstone

SEC699.5

Azure AD
Azure AD Structure and Management
Azure AD Hybrid Authentication
Azure AD Authentication Methods
Azure AD Conditional Access
Introduction to Azure Identities
Azure AD Security Logging

Executing emulation plans
APT-28 Emulation Plan
Exercise: APT-28 Emulation Plan
APT-34 Emulation Plan
Exercise: APT-34 Emulation Plan
Turla Emulation Plan
Exercise: Turla Emulation plan

This page intentionally left blank.

EXERCISE: MANUAL EXECUTION OF TURLA EMULATION PLAN

Please refer to the workbook for further instructions on the exercise!

This page intentionally left blank.

COURSE RESOURCES AND CONTACT INFORMATION

AUTHOR CONTACT
Erik Van Buggenhout
evanbuggenhout@nviso.eu
James Shewmaker
jshewmaker@sans.org

SANS INSTITUTE
11200 Rockville Pike
Suite 200
North Bethesda, MD 20852
301.654.SANS (7267)

PENTEST CONTACT
Stephen Sims
ssims@sans.org

SANS EMAIL
GENERAL INQUIRIES: info@sans.org
REGISTRATION: registration@sans.org
TUITION: tuition@sans.org
PRESS/PR: press@sans.org

This page intentionally left blank.