Microsoft Security

Exploiting Token Based Authentication: Attacking and Defending Identities in the 2020s

Dr Nestori Syynimaa (MSTIC)

Microsoft Security

Who am I?

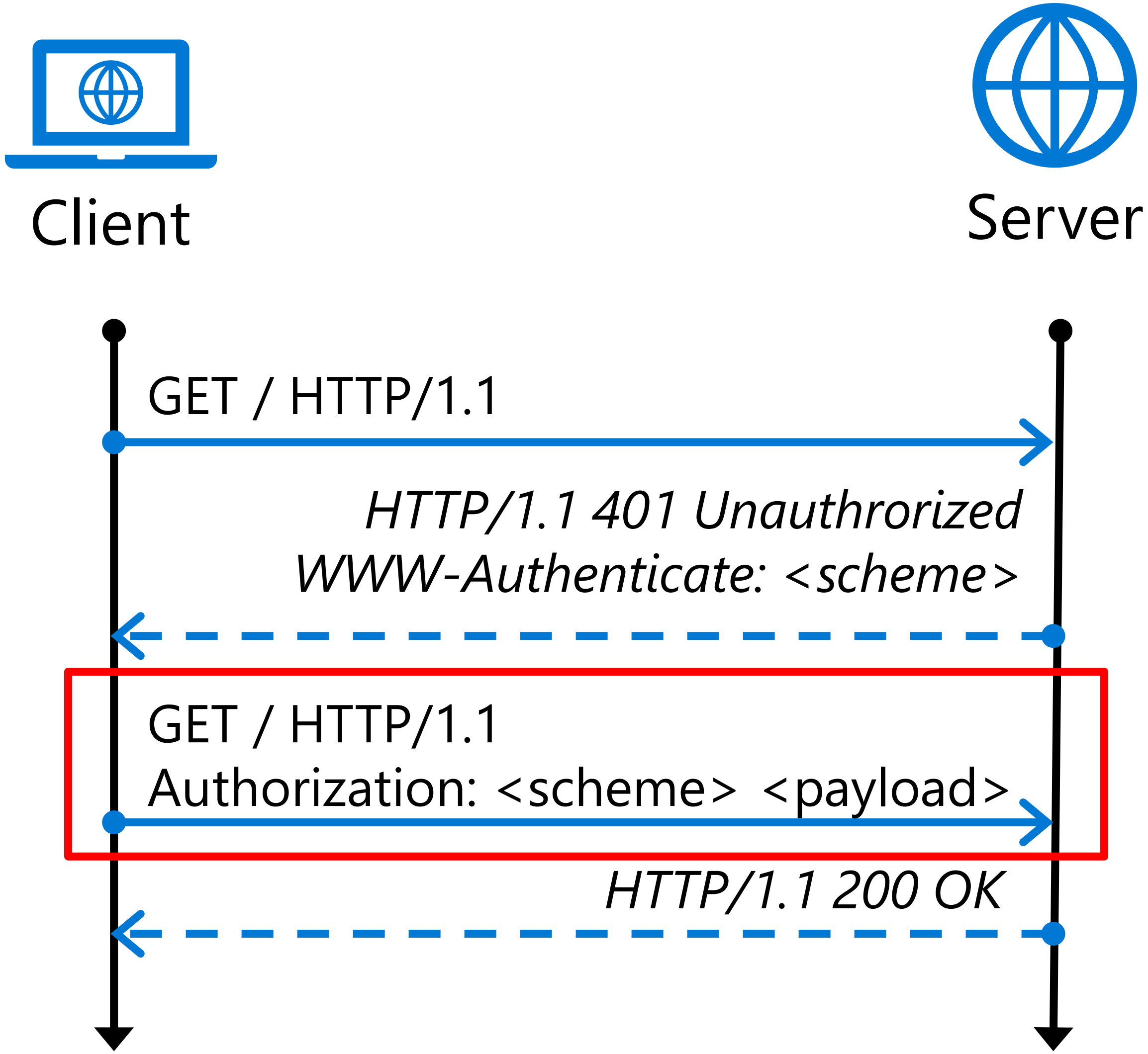
- Dr Nestori Syynimaa (@DrAzureAD)
- Principal Identity Security Researcher
- Microsoft Threat Intelligence Center (MSTIC)

Contents

- Introduction
- Federated authentication methods
- Token based authentication attacks
- Detecting & preventing

Introduction

General HTTP Authentication framework RFC 7235


Client
Server
GET / HTTP/1.1
HTTP/1.1 401 Unauthrorized
WWW-Authenticate: <scheme>
GET / HTTP/1.1
Authorization: <scheme> <payload>
HTTP/1.1 200 OK

- After the authentication, usually *session* cookies are used
- Some schemes:

Basic	RFC 7617
Bearer	RFC 6750
Negotiate / NTLM	RFC 4599

Key concepts

User

- Consumes services

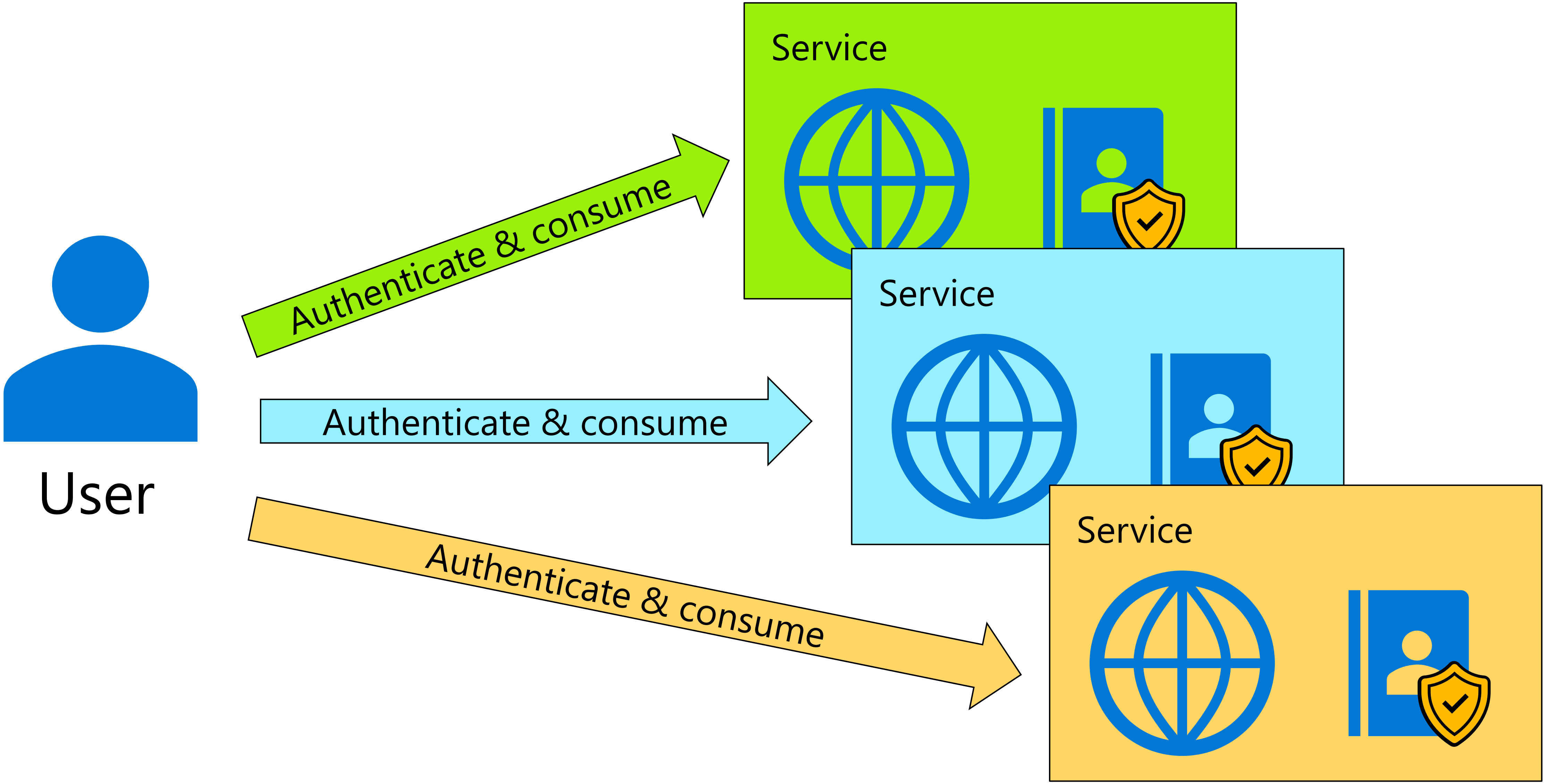
Service Provider (SP)

- Provides services

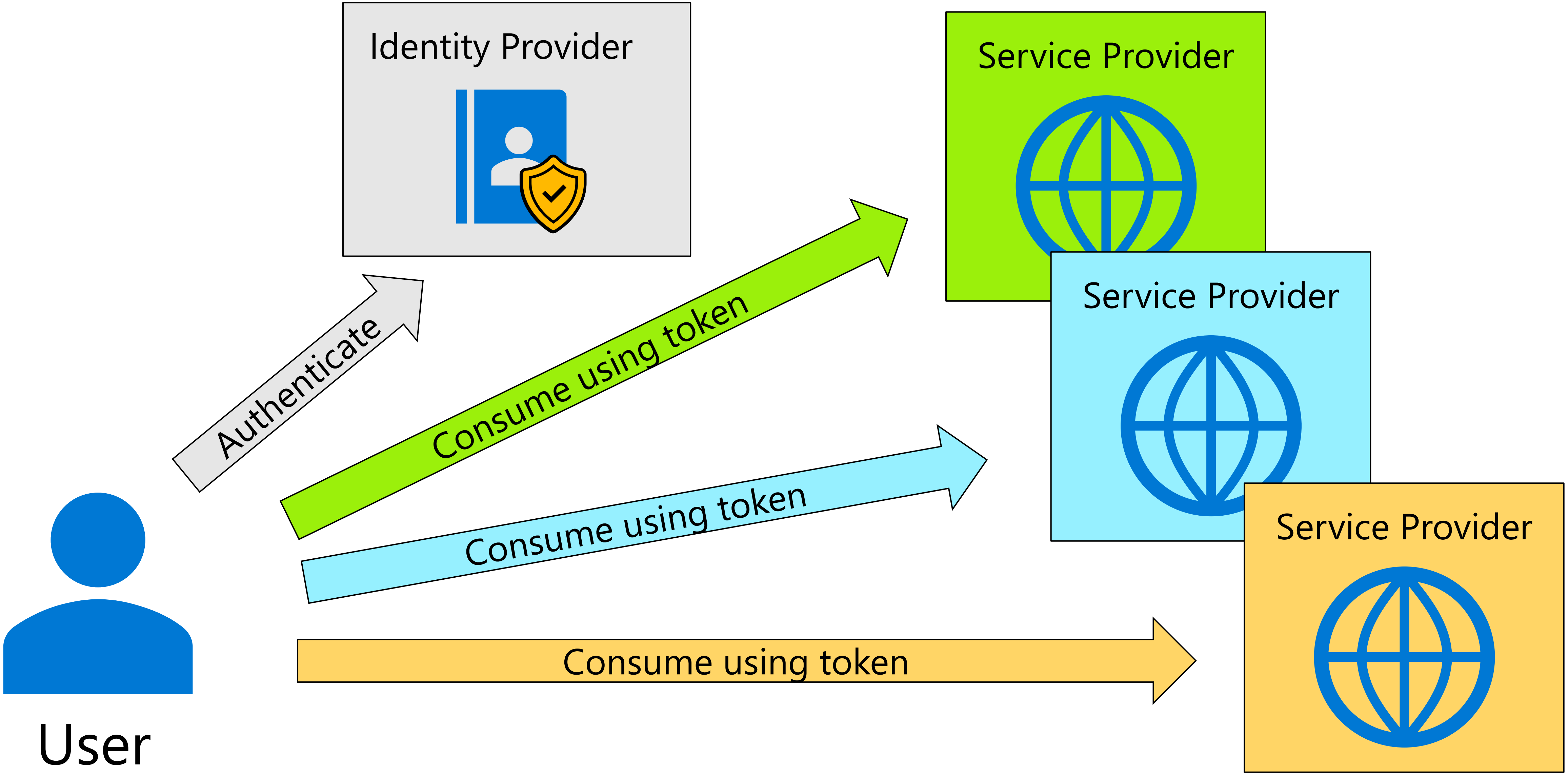
Identity Provider (IdP)

- Provides identity and access management

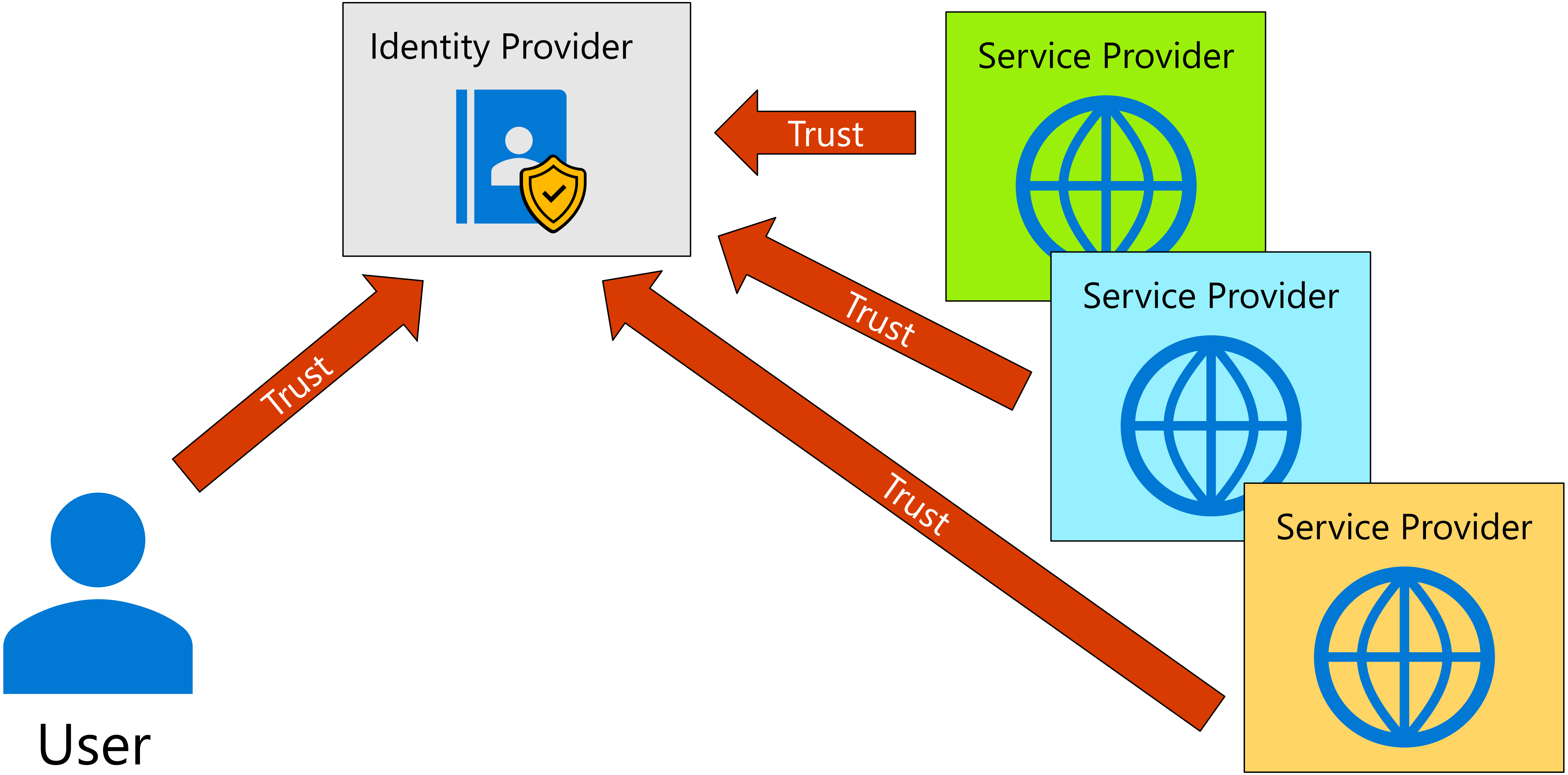
Brief history of authentication: Silo model



Brief history of authentication: Federated model (SSO)

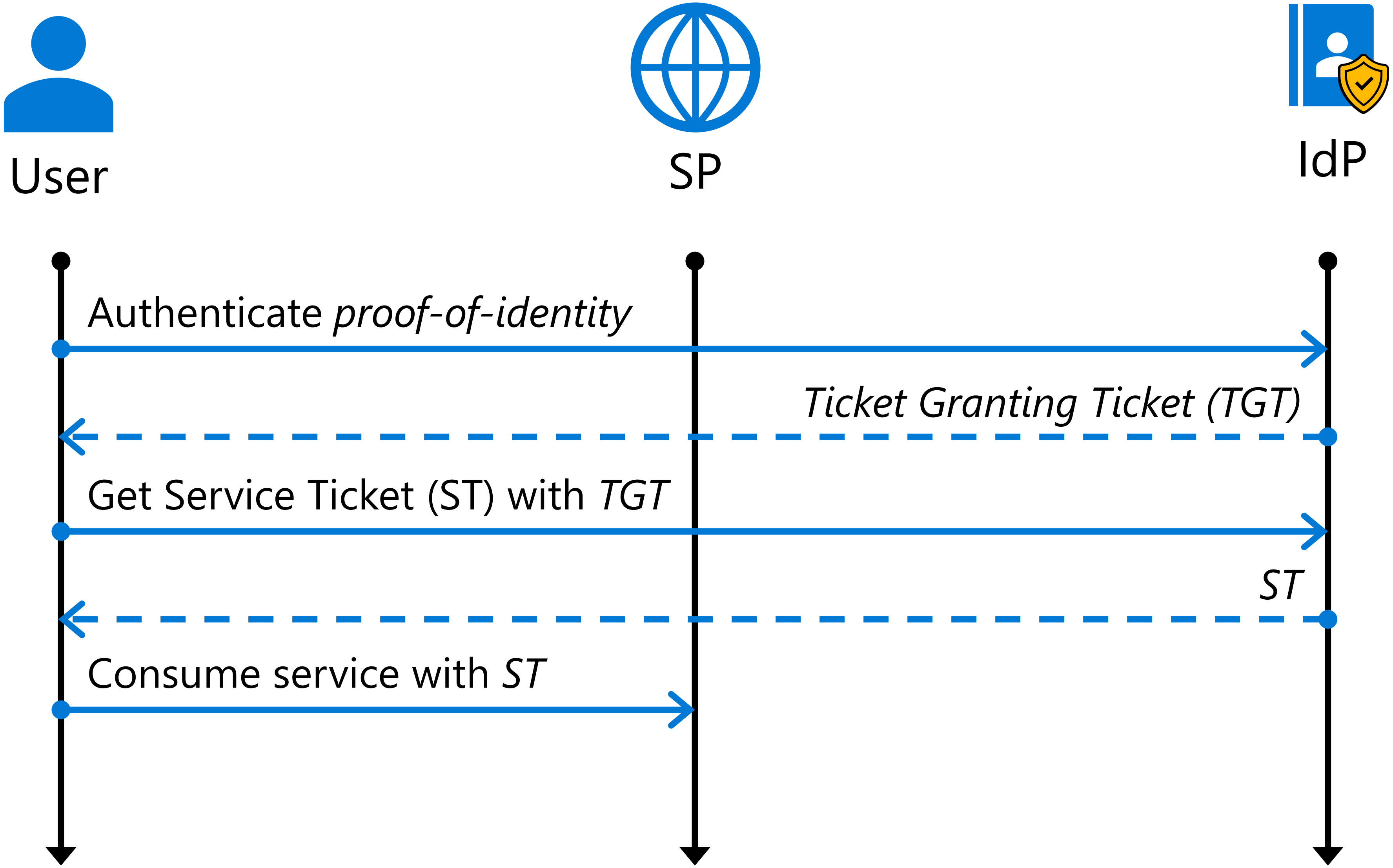


Brief history of authentication: Federated model (SSO)

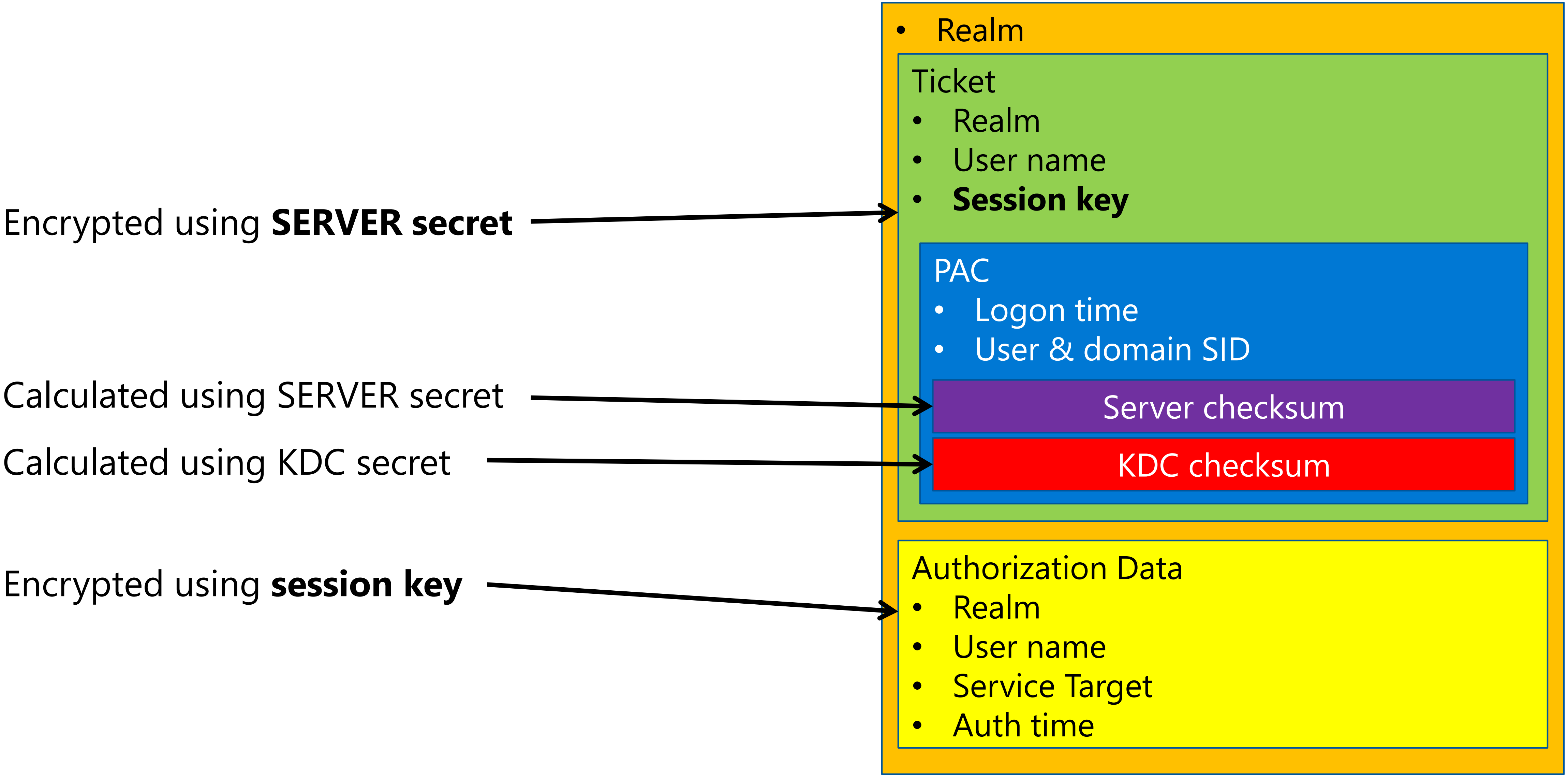


Federated authentication methods

Kerberos authentication flow


User
SP
IdP
Authenticate *proof-of-identity*
Ticket Granting Ticket (TGT)
Get Service Ticket (ST) with *TGT*
ST
Consume service with *ST*

Kerberos Application Request (KRB_AP_REQ) message



- Realm

Ticket

- Realm
- User name
- **Session key**

PAC

- Logon time
- User & domain SID

Server checksum

KDC checksum

Authorization Data

- Realm
- User name
- Service Target
- Auth time

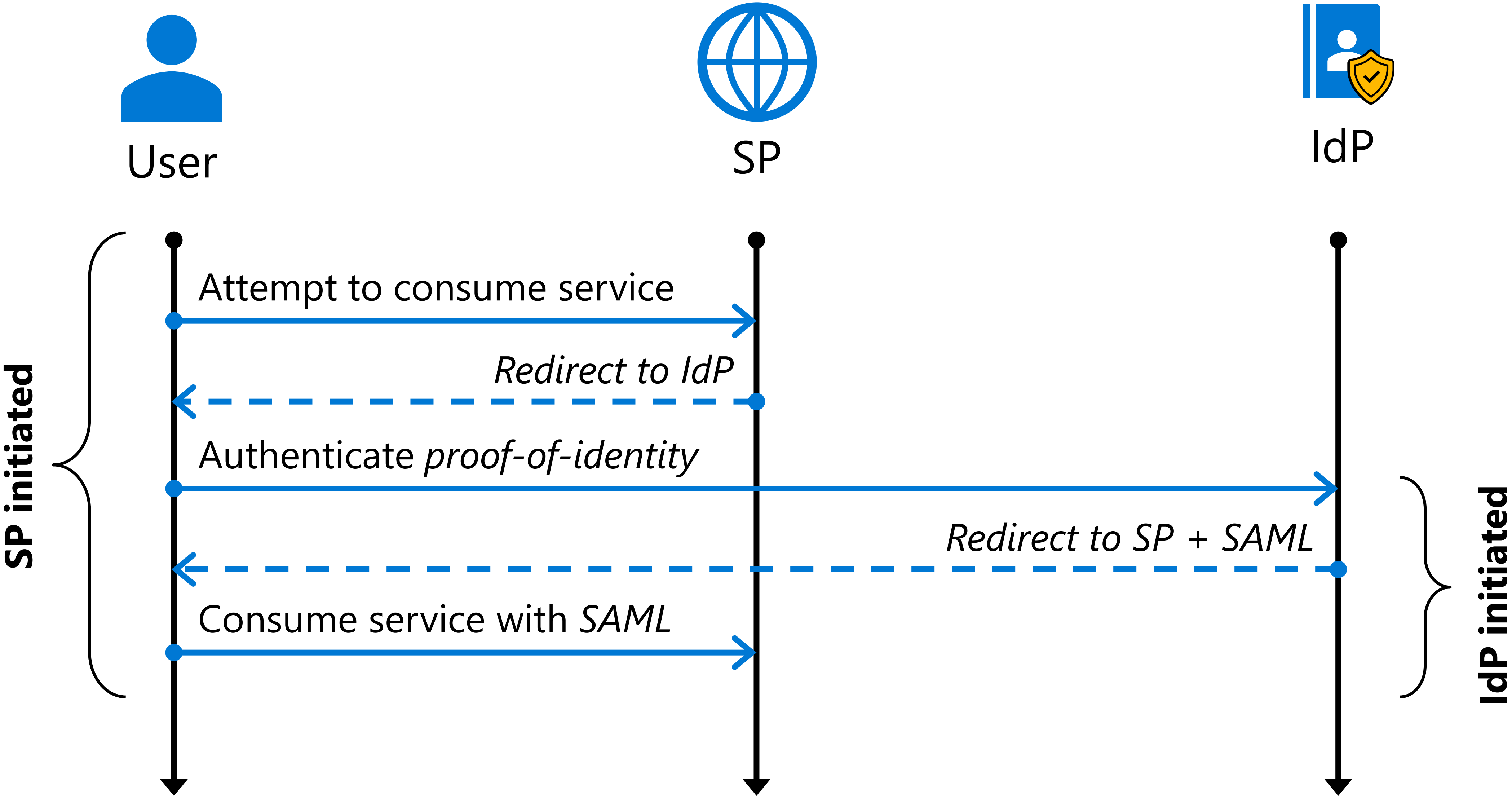
Encrypted using **SERVER secret**

Calculated using SERVER secret

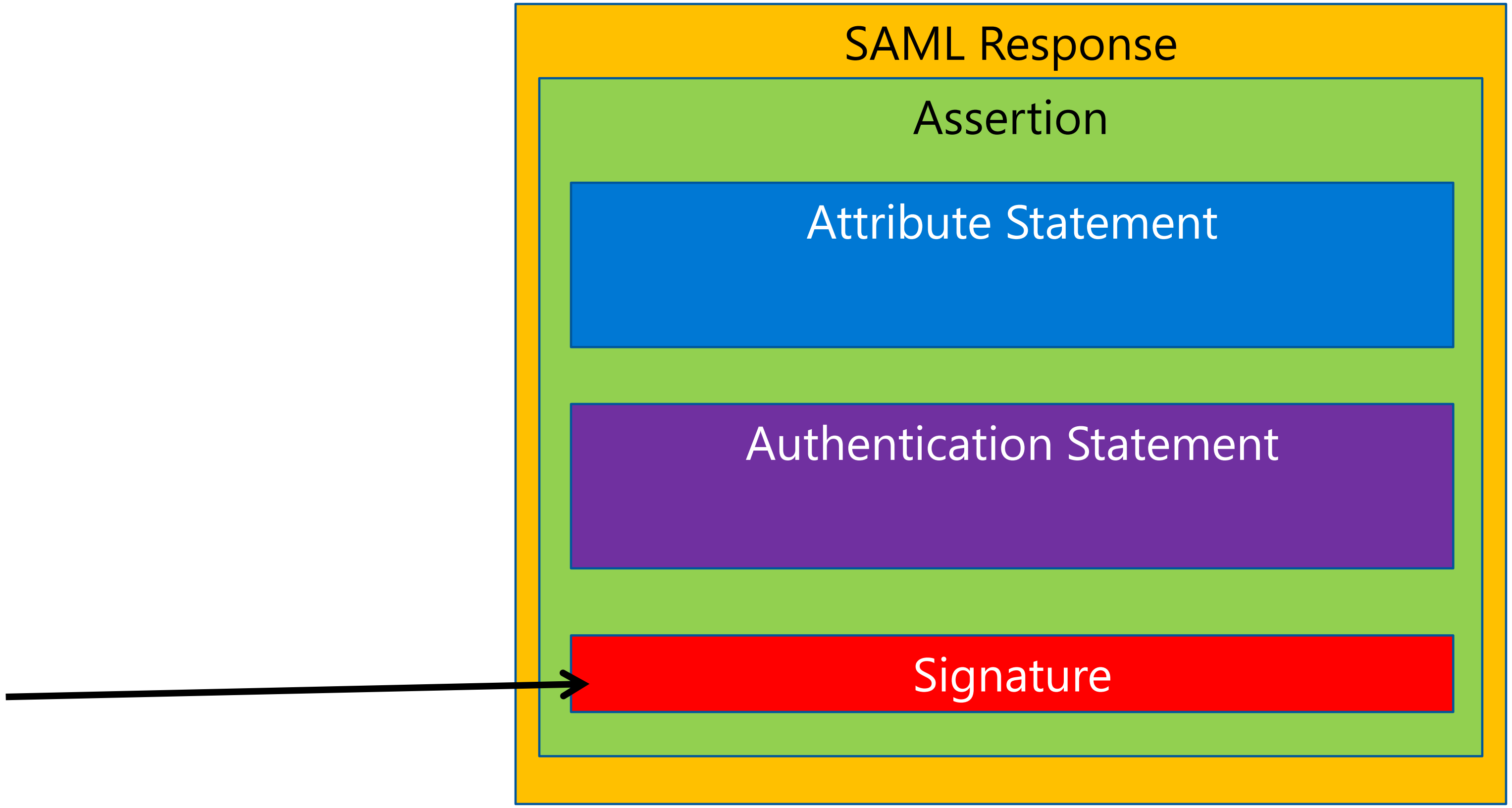
Calculated using KDC secret

Encrypted using **session key**

SAML authentication flows

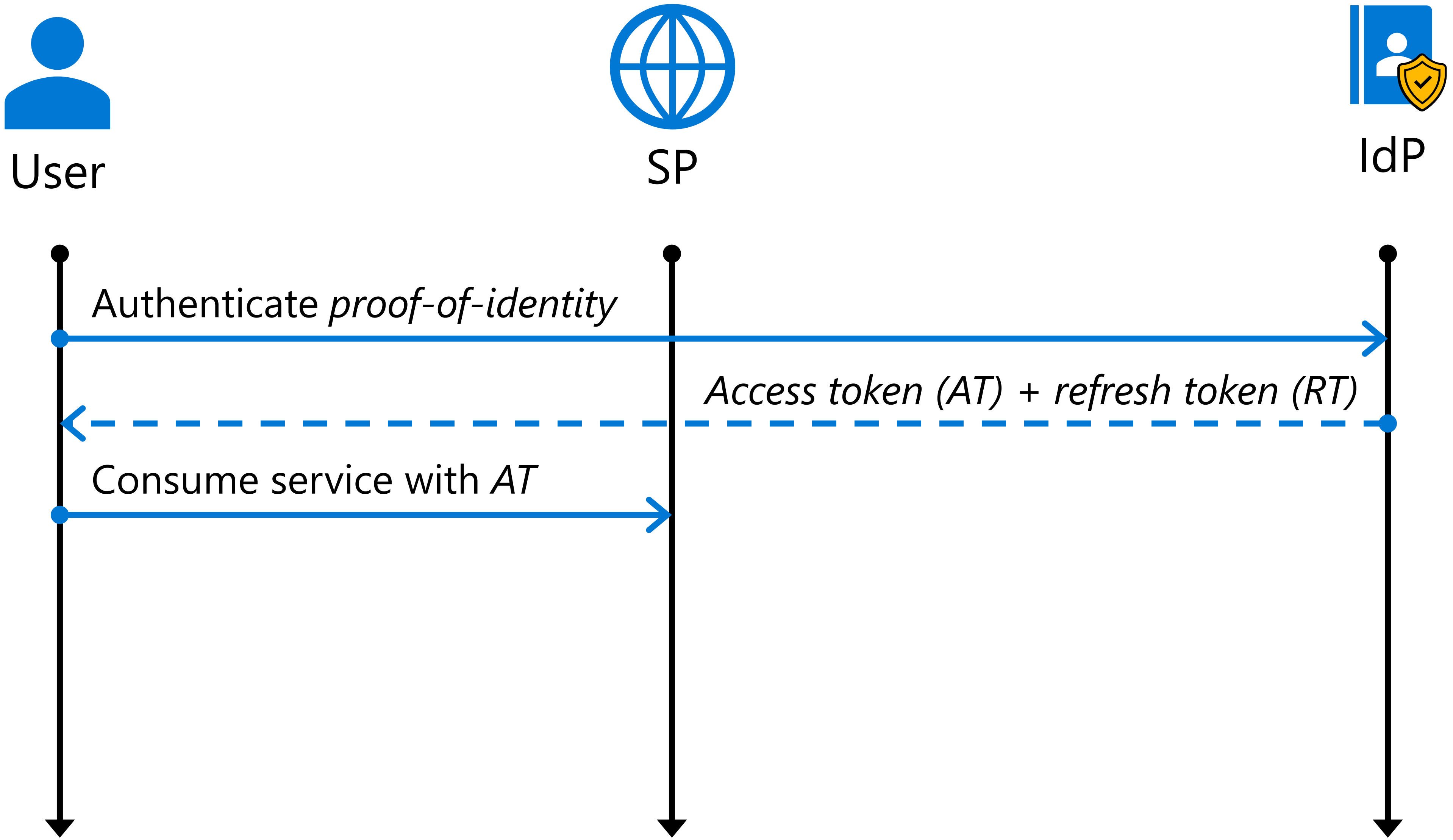

User
SP
IdP
Attempt to consume service
Redirect to IdP
Authenticate *proof-of-identity*
Redirect to SP + SAML
Consume service with *SAML*
SP initiated
IdP initiated

SAML response message



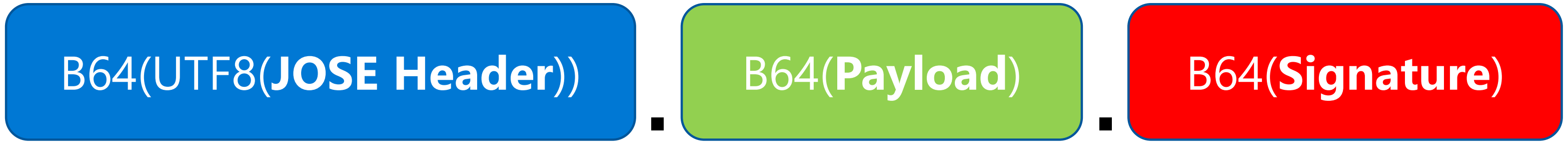
Signed using IdP private key

(simplified) OAuth authentication flow


User
SP
IdP
Authenticate *proof-of-identity*
Access token (AT) + refresh token (RT)
Consume service with *AT*

Entra ID: JSON Web Signature (JWS)

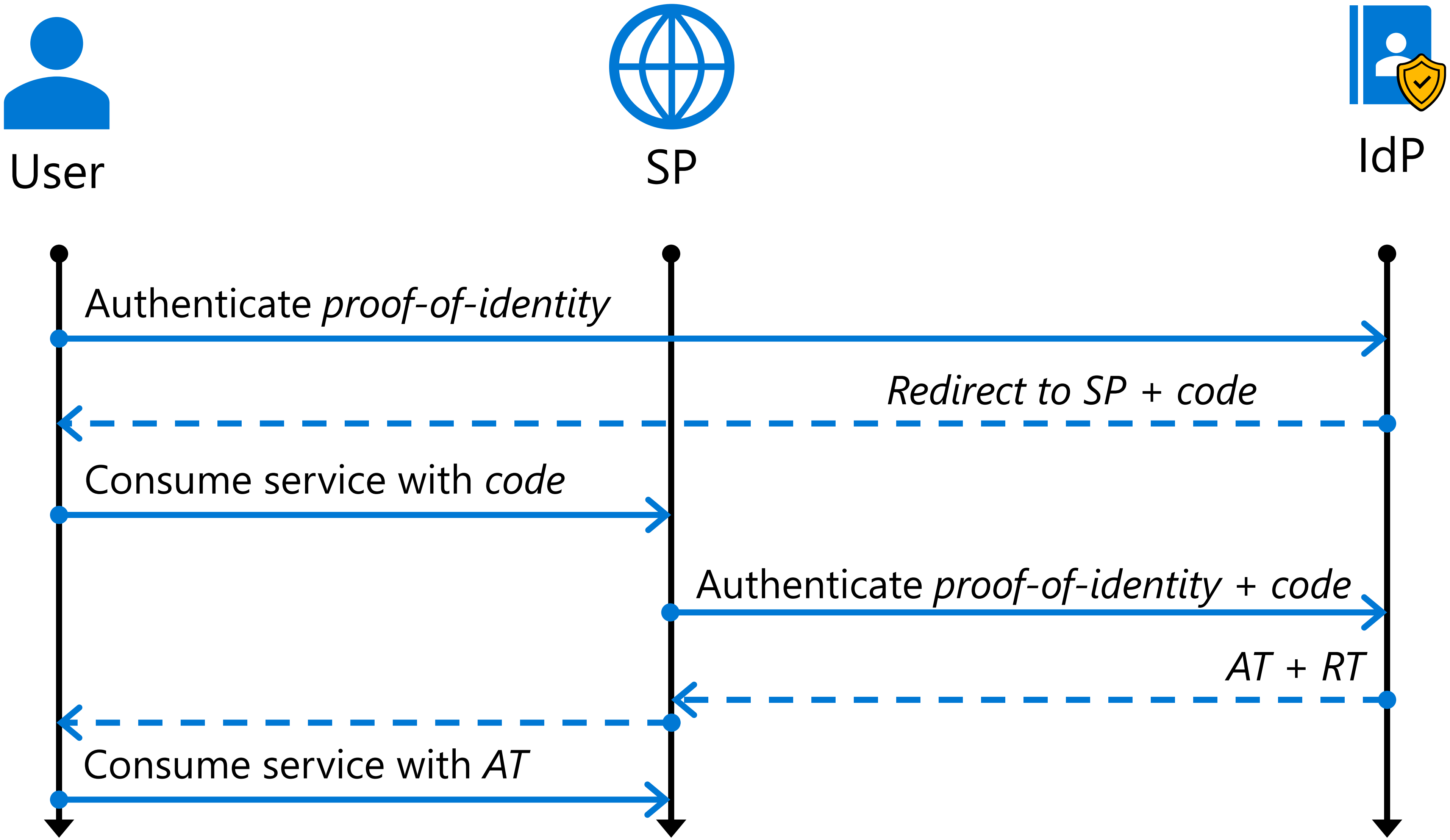
- Used in Entra ID for Access & Id tokens
- Three parts
  - JOSE (Javascript Object Signing and Encryption) Header
  - Payload (a claims set as JSON)
    - User information
    - Device information
    - Client
    - Resource
  - Signature (IdP secret key)



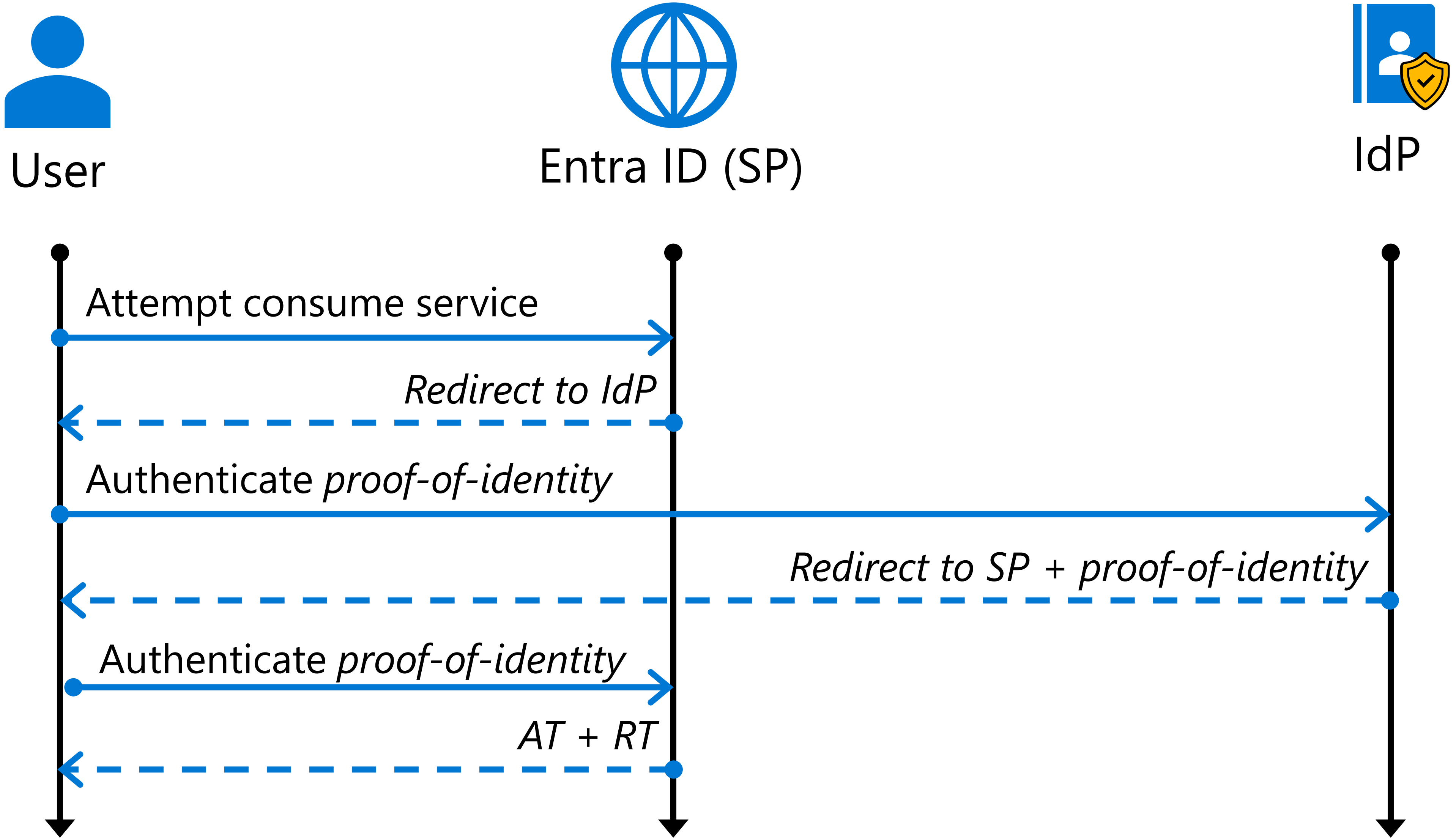
B64(UTF8(**JOSE Header**)) . B64(**Payload**) . B64(**Signature**)

https://www.rfc-editor.org/rfc/rfc7515.html

Entra ID authorization code flow


User
SP
IdP
Authenticate *proof-of-identity*
Redirect to SP + code
Consume service with *code*
Authenticate *proof-of-identity + code*
AT + RT
Consume service with *AT*

Entra ID Hybrid authentication flow


User
Entra ID (SP)
IdP
Attempt consume service
Redirect to IdP
Authenticate *proof-of-identity*
Redirect to SP + proof-of-identity
Authenticate *proof-of-identity*
AT + RT

Summary of federated methods

Protocol	Since	Format	Trust based on
Kerberos	1989	ASN.1	Passwords
SAML*	2002	XML	Certificates
OAuth	2007	JWT (JWS)	Certificates

* SAMLp or WS-FED

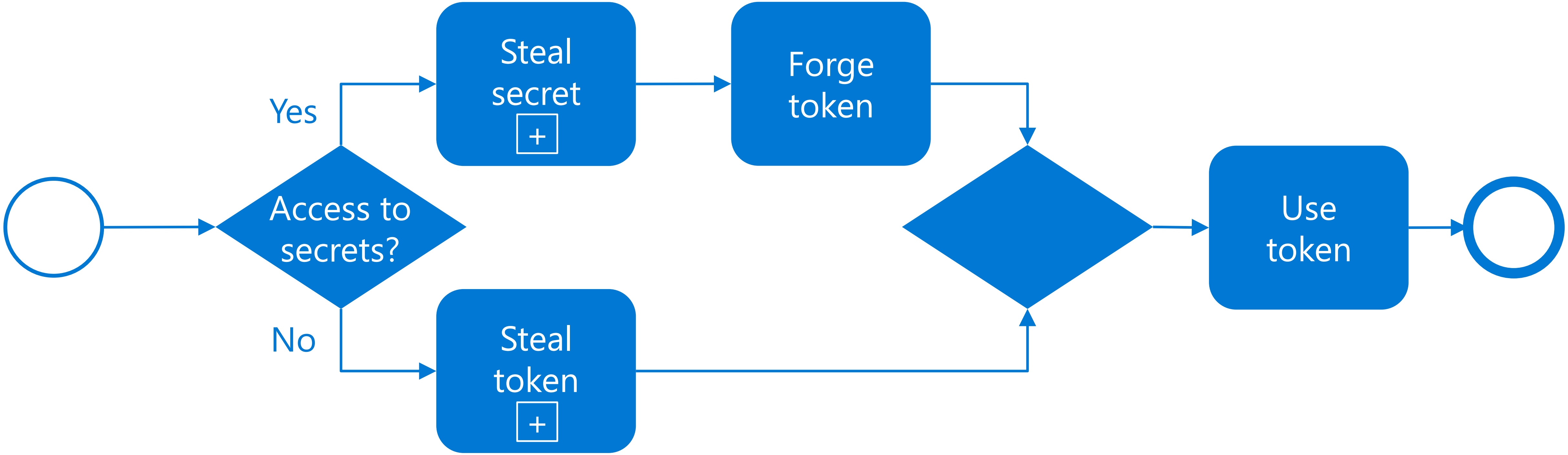
Token-based authentication attacks

Token-based authentication

- **Any** party in **possession of a bearer token** (a "bearer") can use it to get access to the associated resources (without demonstrating possession of a cryptographic key). To prevent misuse, **bearer tokens need to be protected** from disclosure in storage and in transport.

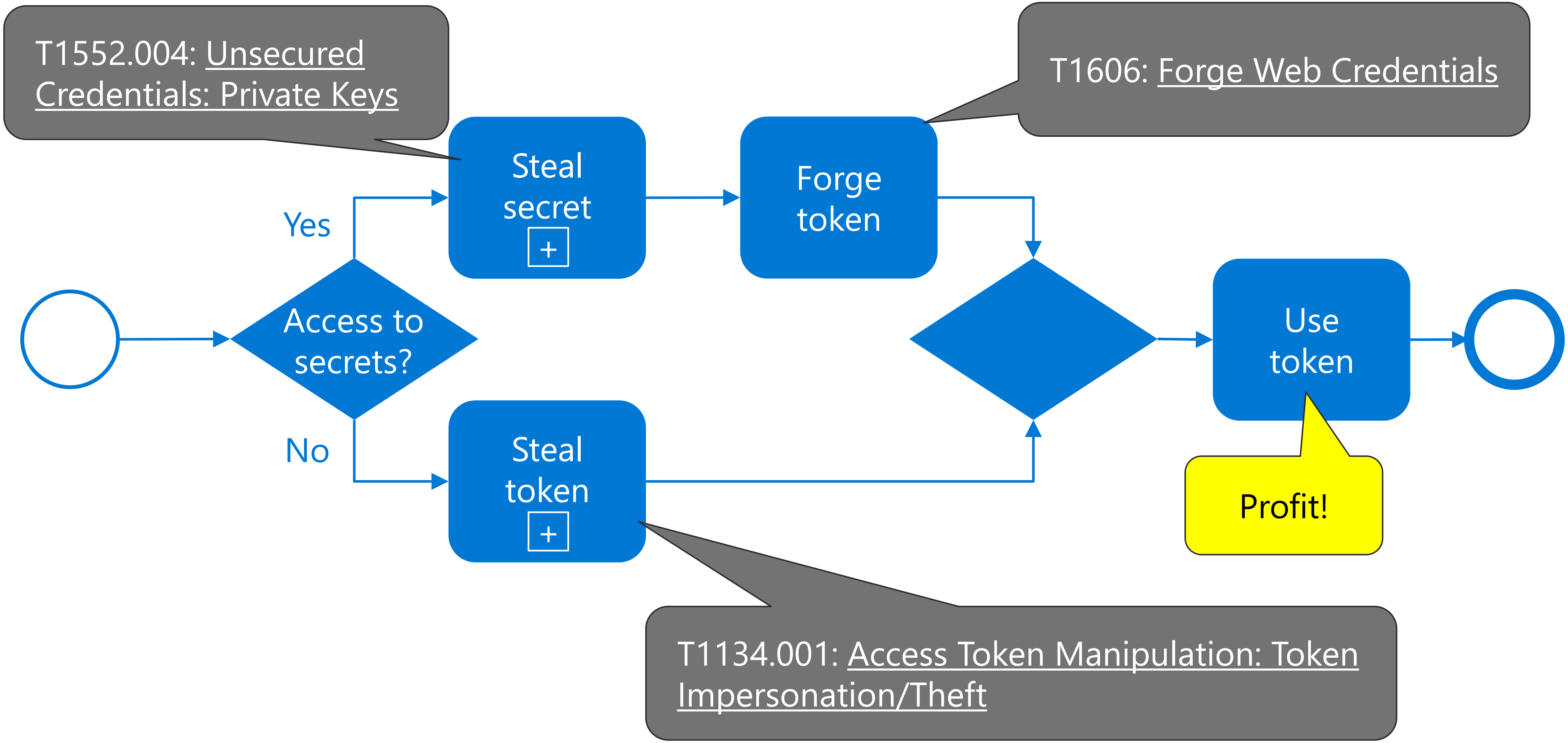
https://datatracker.ietf.org/doc/html/rfc6750

Token-based authentication attack graph



https://www.omg.org/spec/BPMN/2.0/

MITRE ATT&CK® techniques

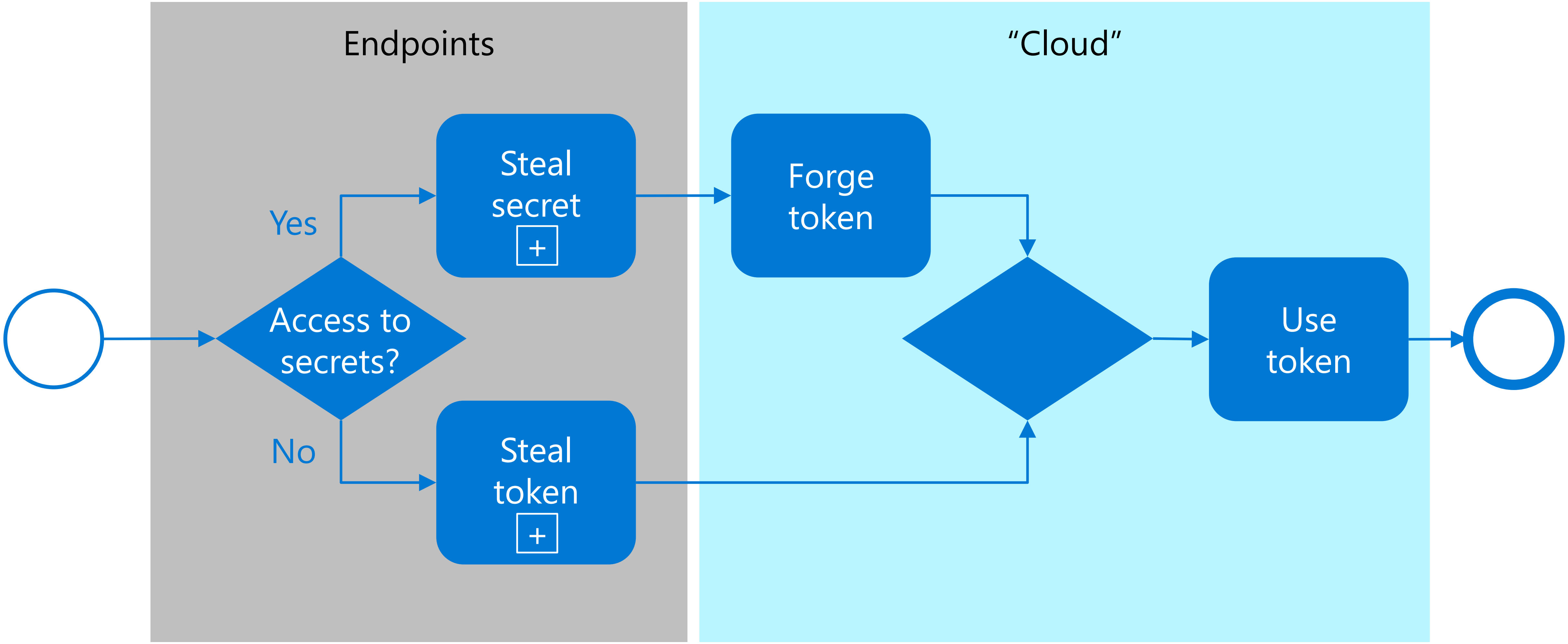


T1552.004: Unsecured Credentials: Private Keys

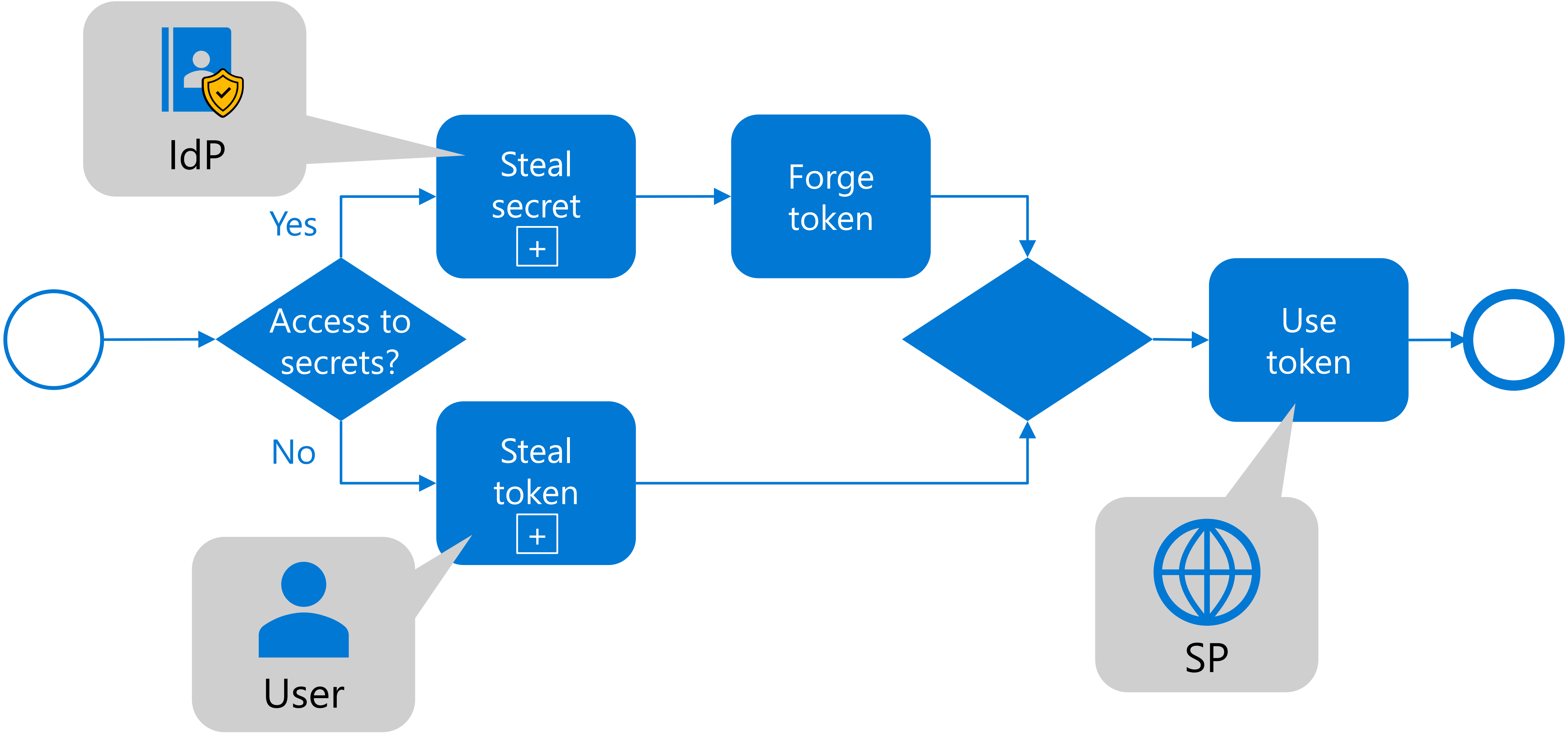
T1606: Forge Web Credentials

T1134.001: Access Token Manipulation: Token Impersonation/Theft

Realms

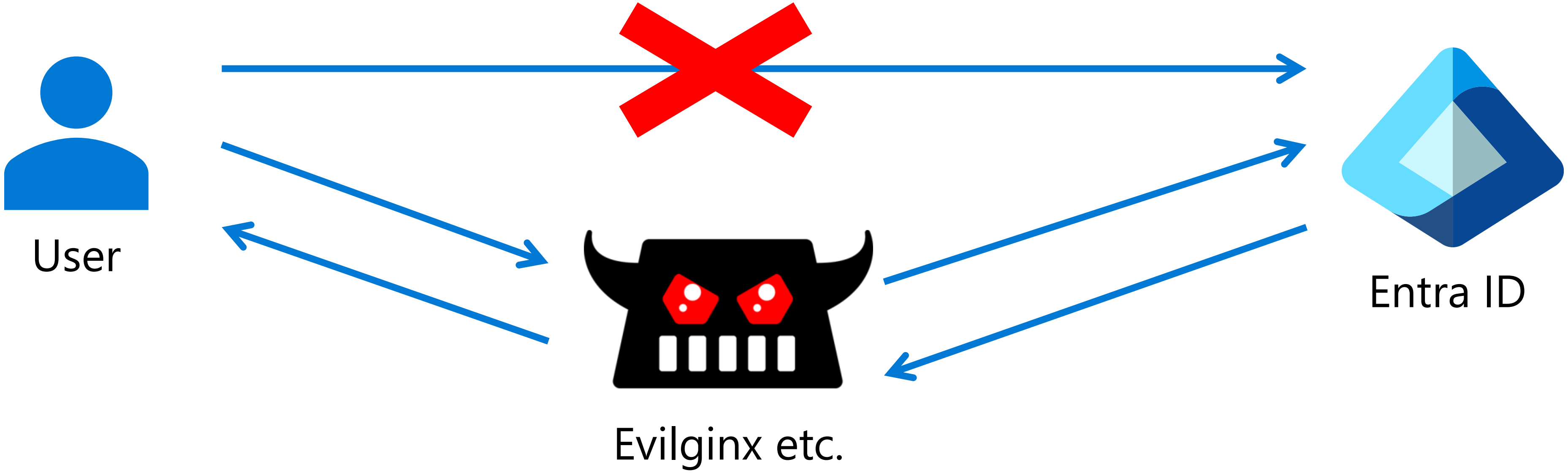

Endpoints
"Cloud"
Access to secrets?
Yes
No
Steal secret
+
Steal token
+
Forge token
Use token

Authentication roles



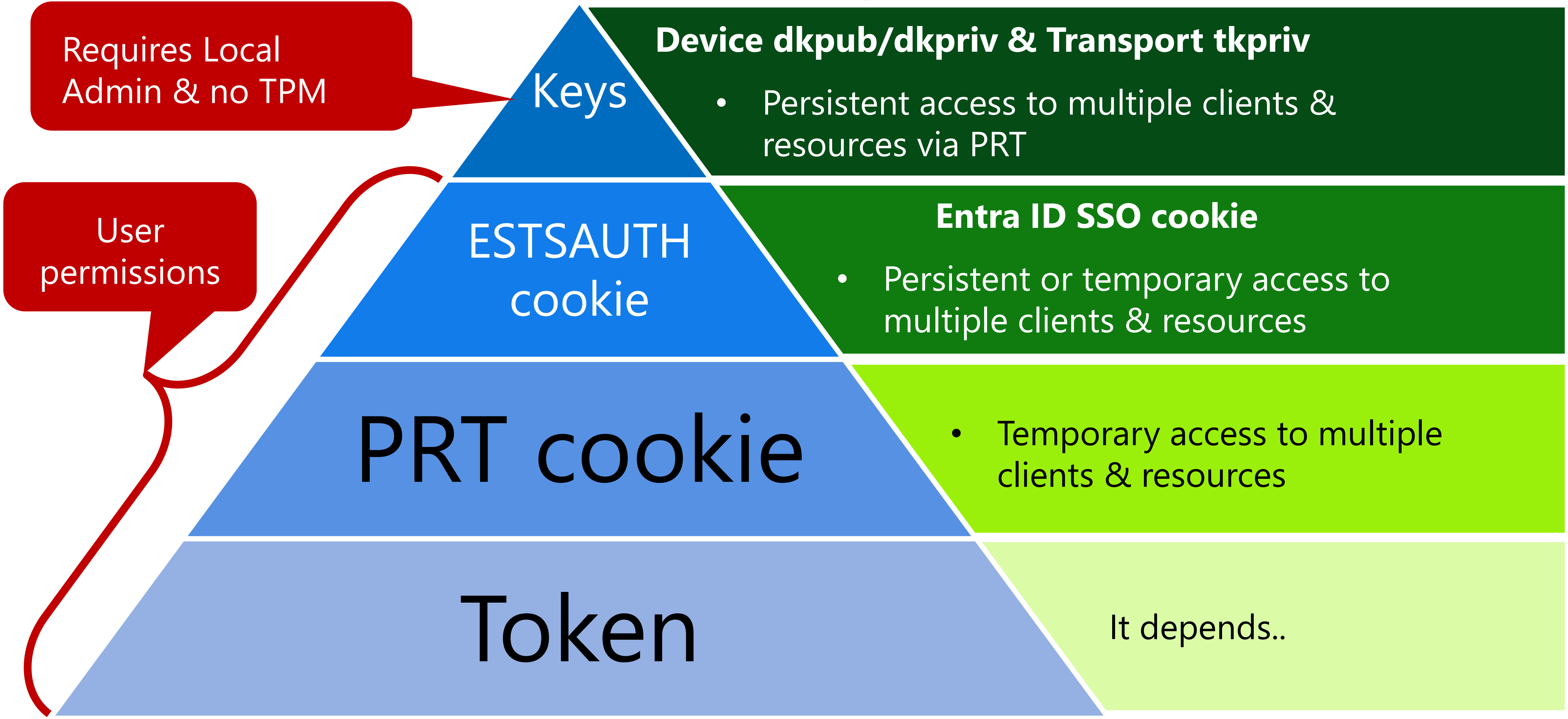
~~Man-in-the-Middle (MitM)~~
Adversary-in-the-Middle (AitM)

- *An attack where the **adversary positions** himself **in between** the **user and** the **system** so that he can intercept and alter data traveling between them.*[1]



[1] NIST Glossary

What to steal from user’s endpoint?



Requires Local Admin & no TPM

User permissions

Keys

Device dkpub/dkpriv & Transport tkpriv

- Persistent access to multiple clients & resources via PRT

ESTSAUTH cookie

Entra ID SSO cookie

- Persistent or temporary access to multiple clients & resources

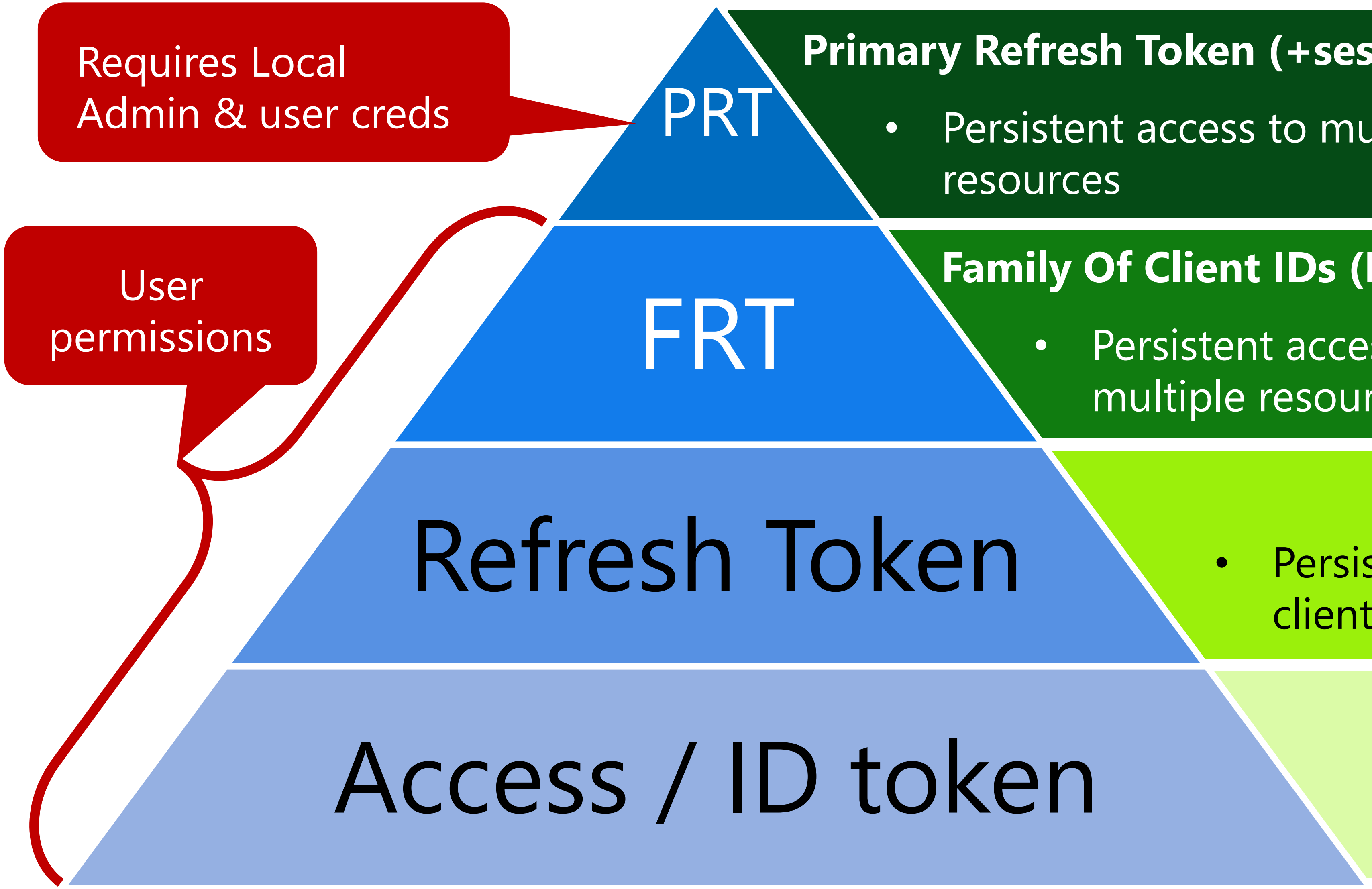
PRT cookie

- Temporary access to multiple clients & resources

Token

It depends..

What token to steal?



Requires Local Admin & user creds

User permissions

Primary Refresh Token (+session key) **90 d**

- Persistent access to multiple clients & resources

Family Of Client IDs (FOCI) Refresh Token **90 d**

- Persistent access to FOCI clients & multiple resources

90 d

- Persistent access to single client & multiple resources

1-28 h

- Temporary access to single client & resource

Meet our bad guys:

@L364CYB173

- Writes shellcode with HxD
- Can remotely spawn calc.exe on patched Windows
- Limitless resources

@CL0UD3N16M4

- Lives in parents' cellar
- Knows how the f*ck the cloud works
- 15€ weekly allowance

Demo: Stealing signing secrets and forging tokens

I got access to target organization AD FS server

Nice! Can you get me AD FS token signing certificate?

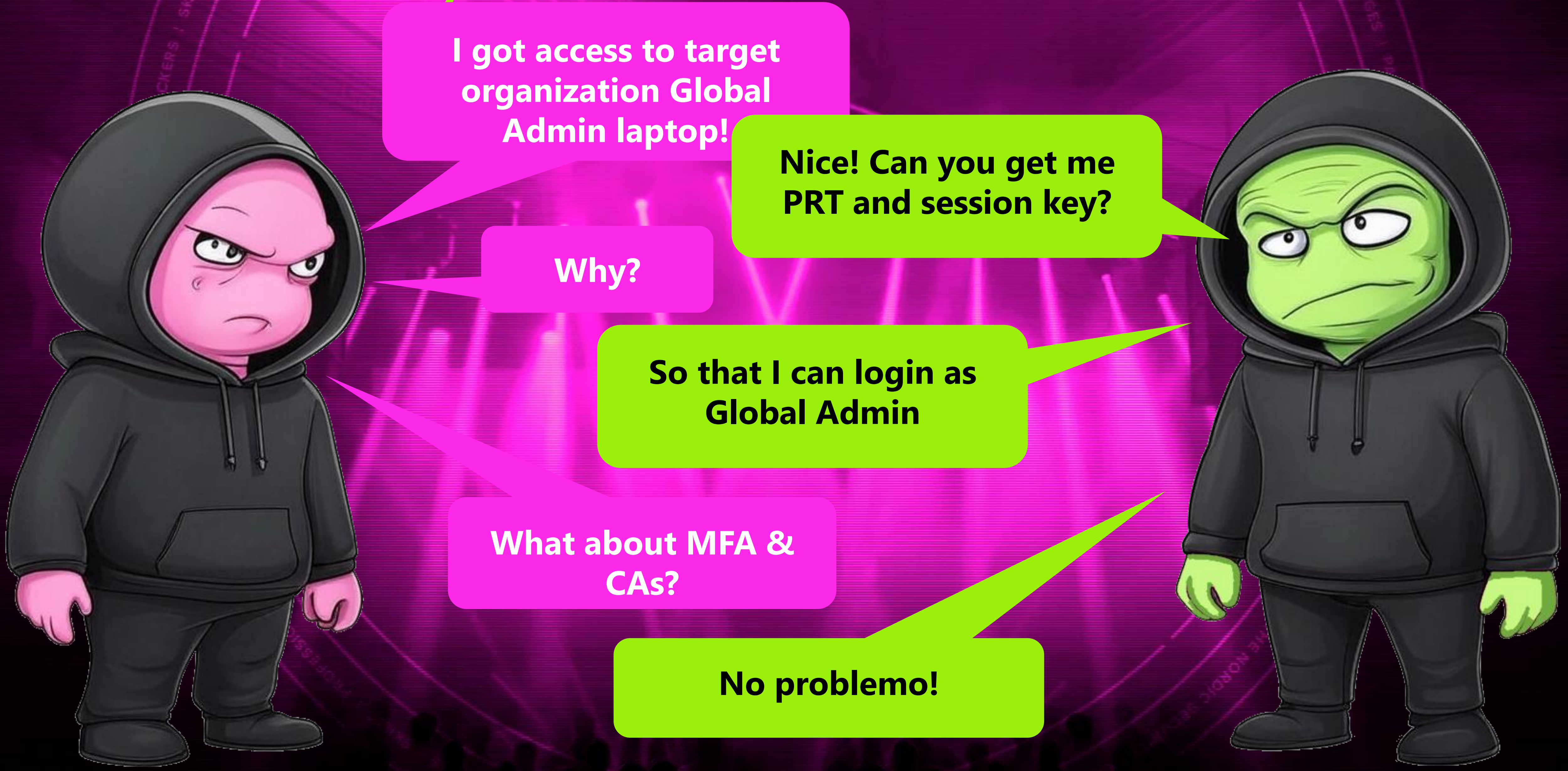
Why?

So that I can forge SAML tokens

Why?

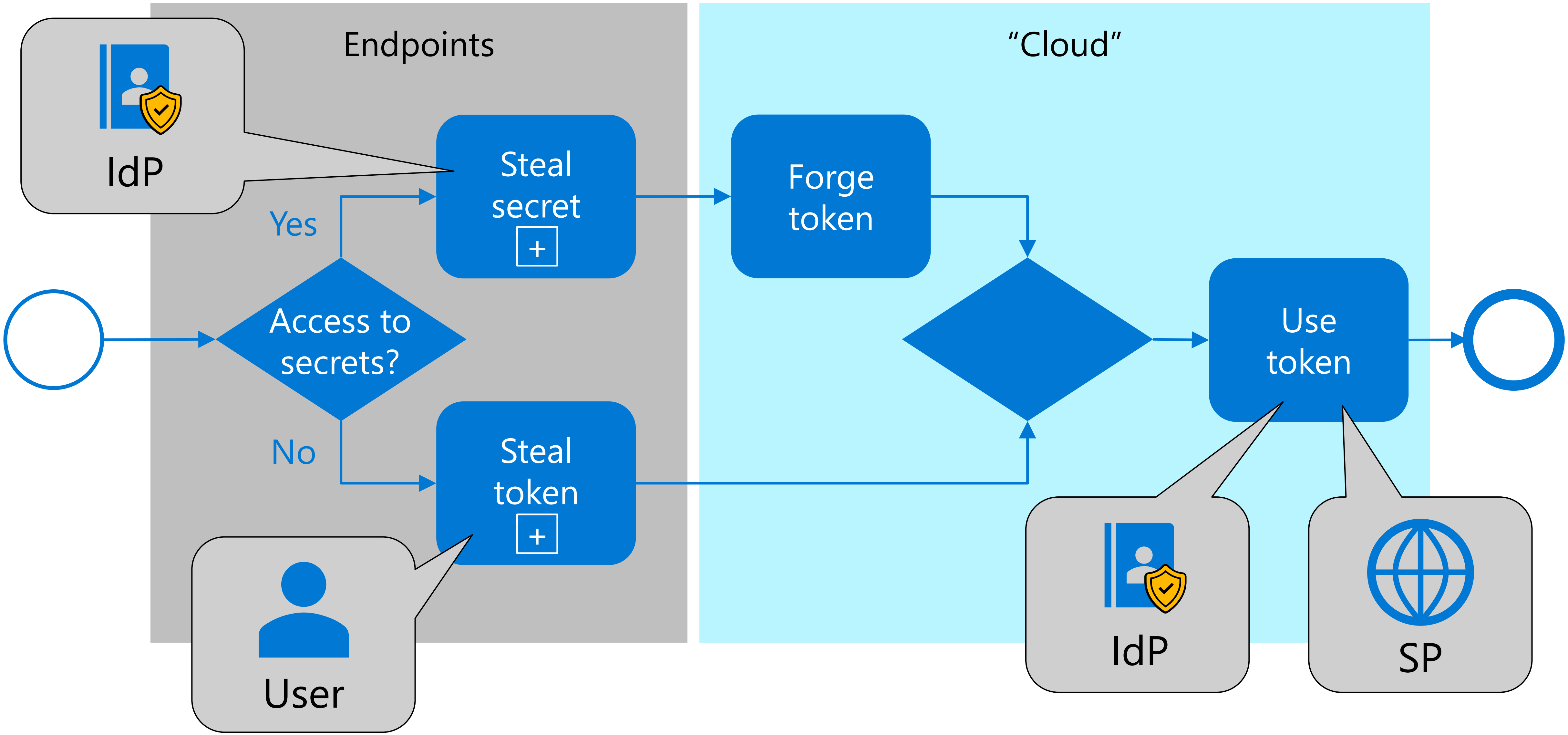
So that I can log in as any user of the tenant

Demo: Stealing tokens


I got access to target organization Global Admin laptop!
Nice! Can you get me PRT and session key?
Why?
So that I can login as Global Admin
What about MFA & CAs?
No problemo!

Detecting & preventing

Detection sources



Scenario 1: On-prem identity

On-prem
Active Directory

On-prem
web server

Logon events

Usage logs

Scenario 2: Hybrid identity

On-prem
Active Directory

On-prem
AD FS

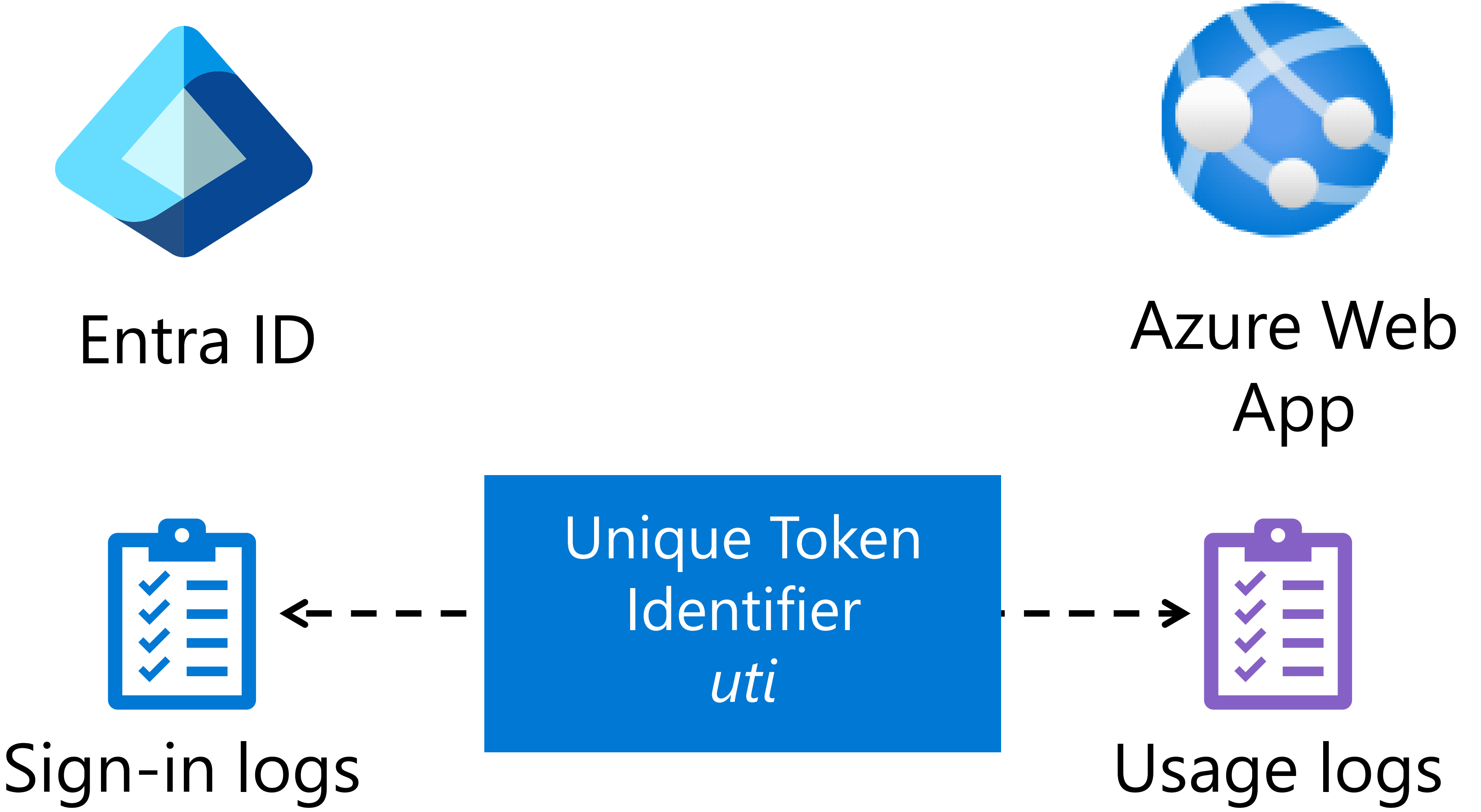
Entra ID

Logon events

AD FS audit events

Sign-in logs

Scenario 3: Cloud-only identity 1



Entra ID

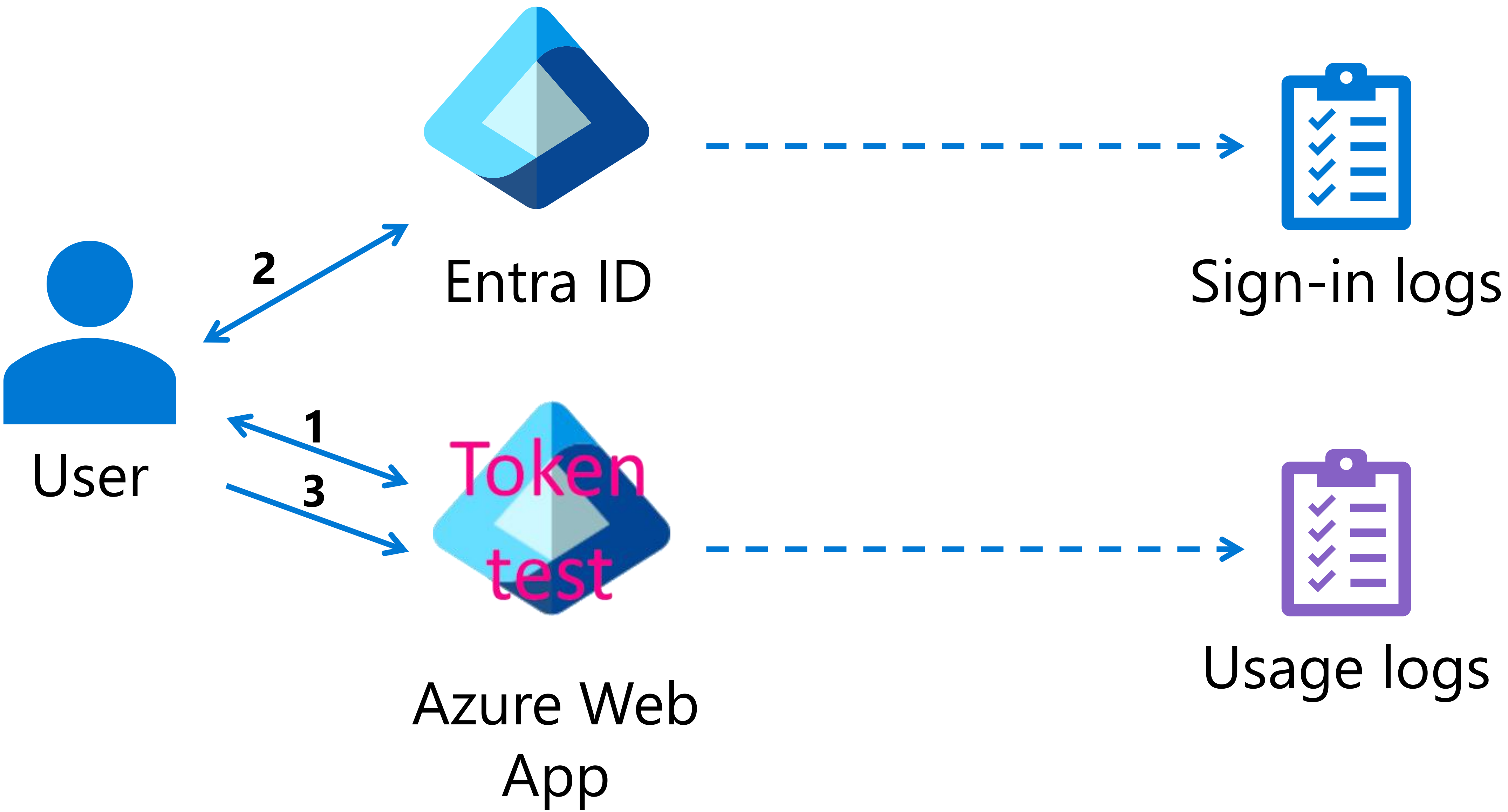
Azure Web App

Unique Token Identifier
uti

Sign-in logs

Usage logs

Demo



Entra ID

Sign-in logs

User

Azure Web App

Usage logs

Scenario 4: Cloud-only identity 2

Entra ID

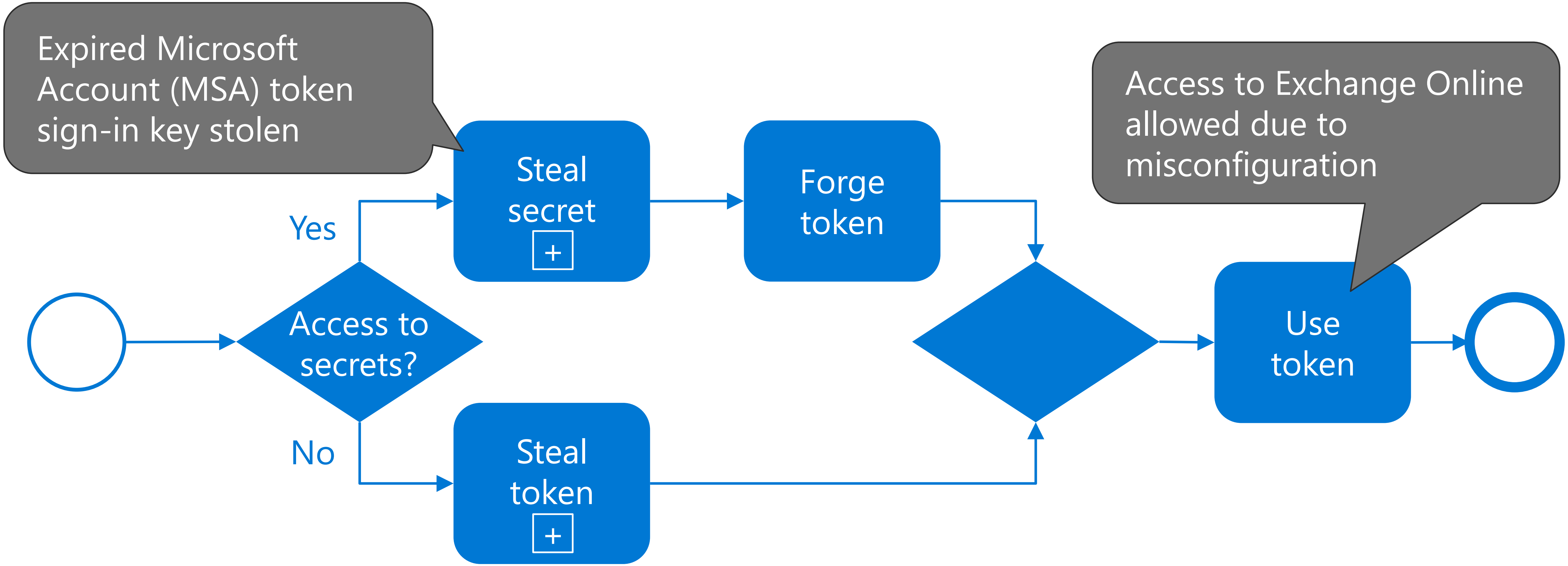
Microsoft 365

Sign-in logs

MS Graph Activity Logs

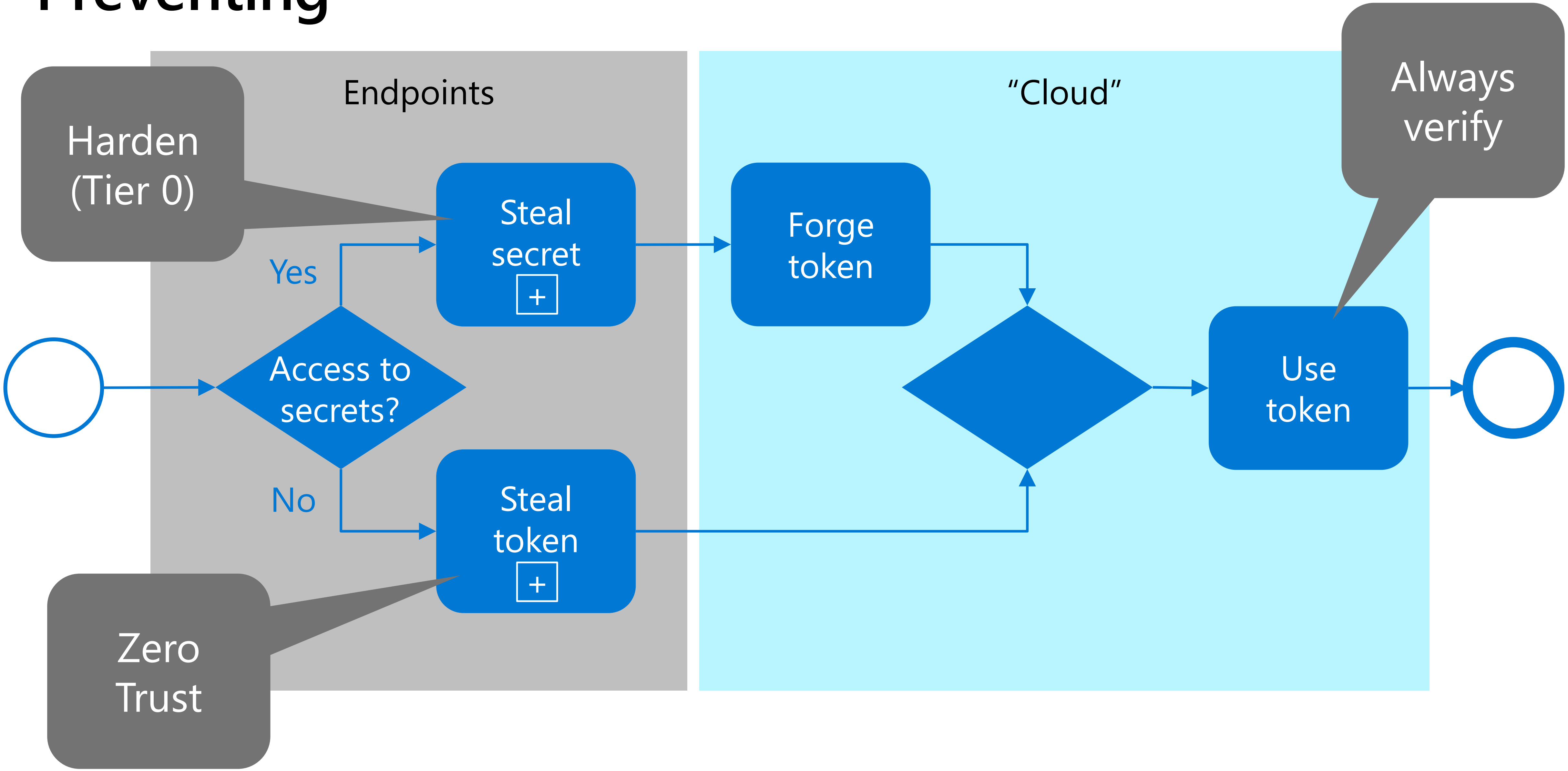
Unified audit log

Storm-0558 accessed emails of 25 organisations

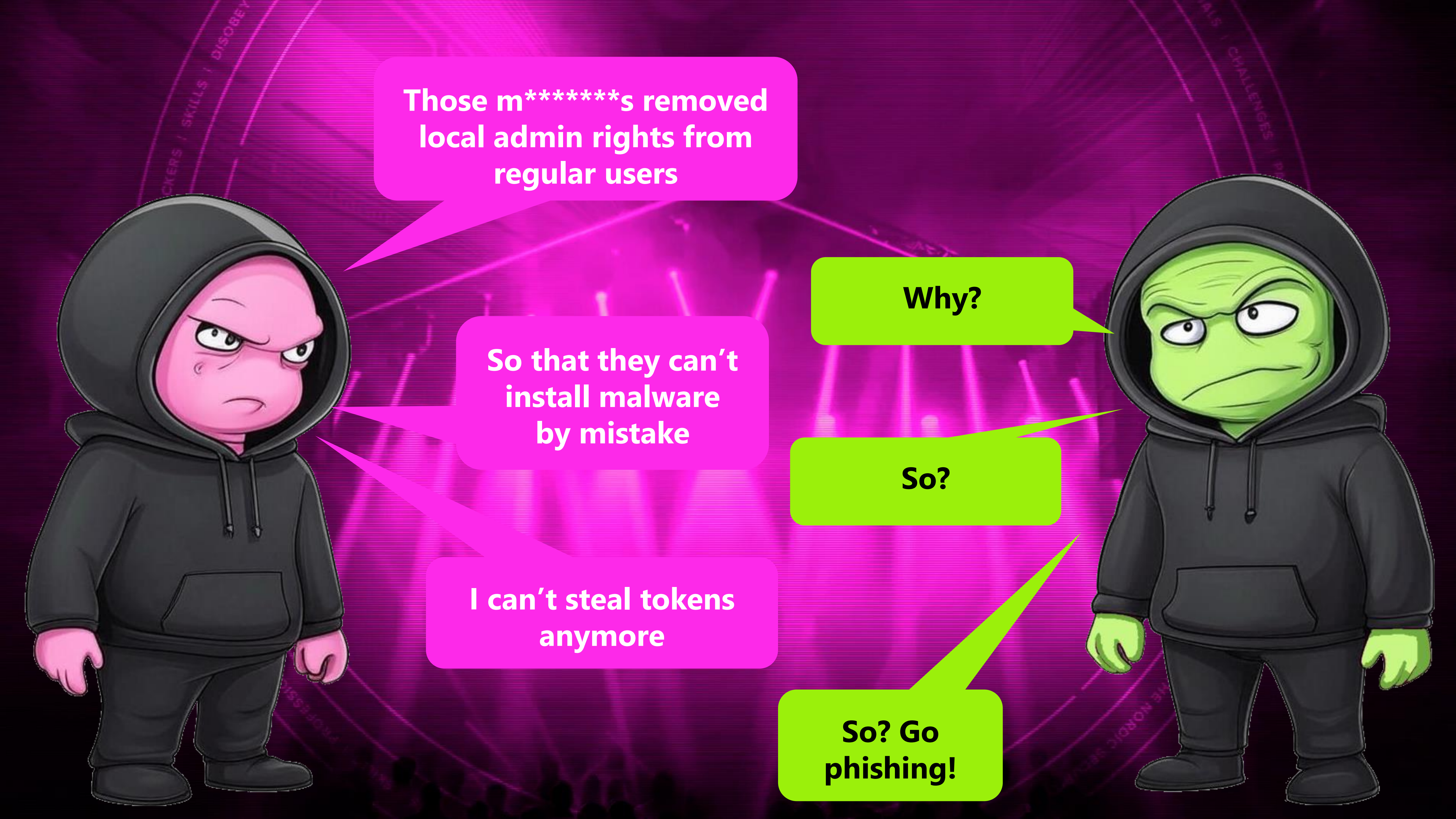

Expired Microsoft Account (MSA) token sign-in key stolen
Access to Exchange Online allowed due to misconfiguration
Yes
No
Access to secrets?
Steal secret
+
Forge token
Steal token
+
Use token

https://aka.ms/storm-0558
https://msrc.microsoft.com/blog/2023/07/microsoft-mitigates-china-based-threat-actor-storm-0558-targeting-of-customer-email/
https://msrc.microsoft.com/blog/2023/09/results-of-major-technical-investigations-for-storm-0558-key-acquisition/

Preventing


Endpoints
"Cloud"
Harden (Tier 0)
Always verify
Steal secret
Forge token
Yes
Access to secrets?
No
Use token
Steal token
Zero Trust

Sh*t! They have hardened their AD FS server farm!
Why?
Because it's "TIER 0" assett
So?
I can't steal token signing certificate anymore
So? Steal some tokens then!

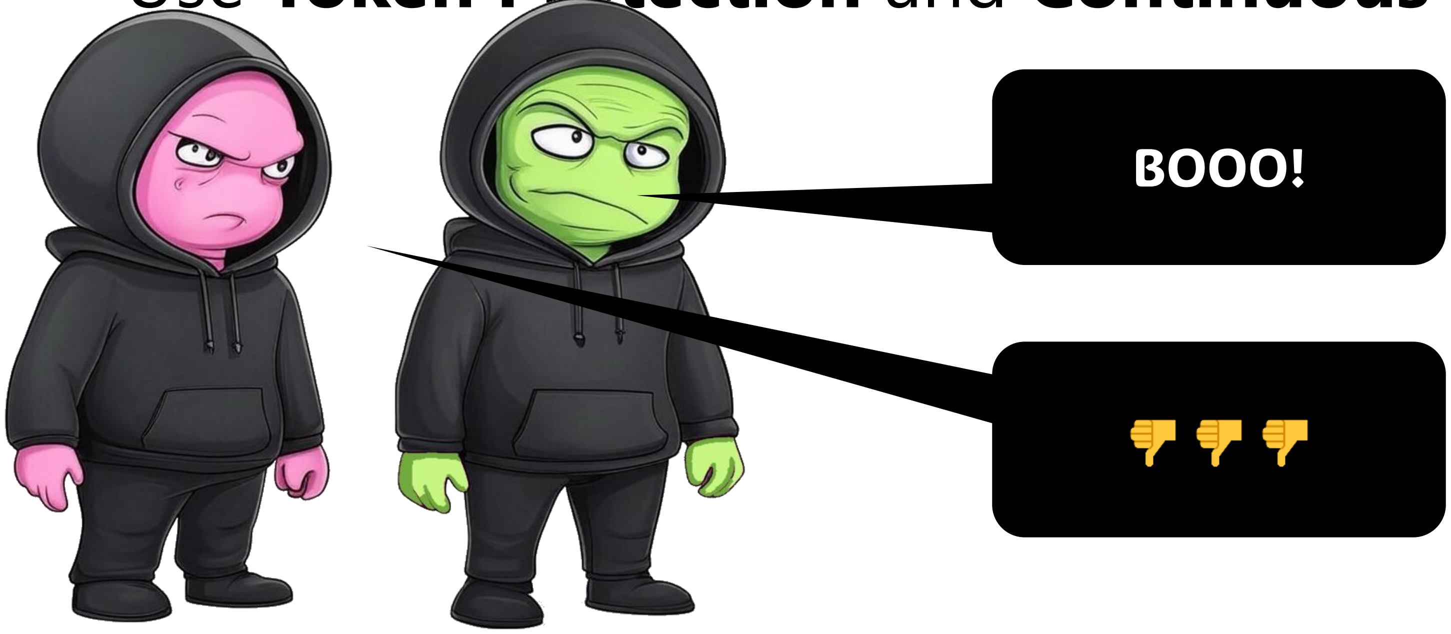

Those m*******s removed local admin rights from regular users
Why?
So that they can’t install malware by mistake
So?
I can’t steal tokens anymore
So? Go phishing!

Aaargh! Now they are using Token Protection!
Why?
To mitigate against AiTM attacks!
So?
I can’t even phish tokens anymore!
Oh sh*t!

Summary

Summary

- Stealing ***tokens*** gives temporary access as one person
- Stealing token sign-in ***secrets*** gives persistent access as any person
- Detecting and preventing token-theft is a team sport
- Detection requires access to **IdP *and* SP** logs
- Use **Token Protection** and **Continuous Access Evaluation**



Microsoft Security

THANK YOU!

F*CK OFF!

N00b!!