LIVE WEBINAR · 90 MIN + 30 MIN Q&A

From initial access to persistence

Abusing passkeys in Azure

> A token is a time-bound ticket An authenticator is a new key to the building

Altered Security

- Trained more than 50000 security professionals from more than 130 countries!
- Our Red Team Labs Platform enables labs to be:
  - Affordable
  - Easy to Access
  - Stable and provide great user experience
  - Fun to Solve
  - Big enough to feel enterprise-like

ALTERED SECURITY

Red team labs	alteredsecurity.com/online-labs
Instructor-led bootcamps	alteredsecurity.com/bootcamps
GitHub	github.com/AlteredSecurity
Lab Platform	enterprisesecurity.io
Free Labs and Challenges	redlabs.enterprisesecurity.io

What is Hacker Summer

- Altered Security's month-long Red Teaming event.
  - Research blogs and webinars on Red Teaming for on-prem, Azure and AI
  - Call for Papers.
  - Discounts and giveaways.

Webinar Ground Rules

- You can ask questions on general channel (#general) on our Discord server (https://discord.com/invite/vcEwaRMwJe). We won't be able to monitor Zoom Chat.
- To ensure a smooth experience, participants are not allowed to use their microphone.
- Please maintain professional conduct and a respectful atmosphere throughout the webinar.

ABOUT ME

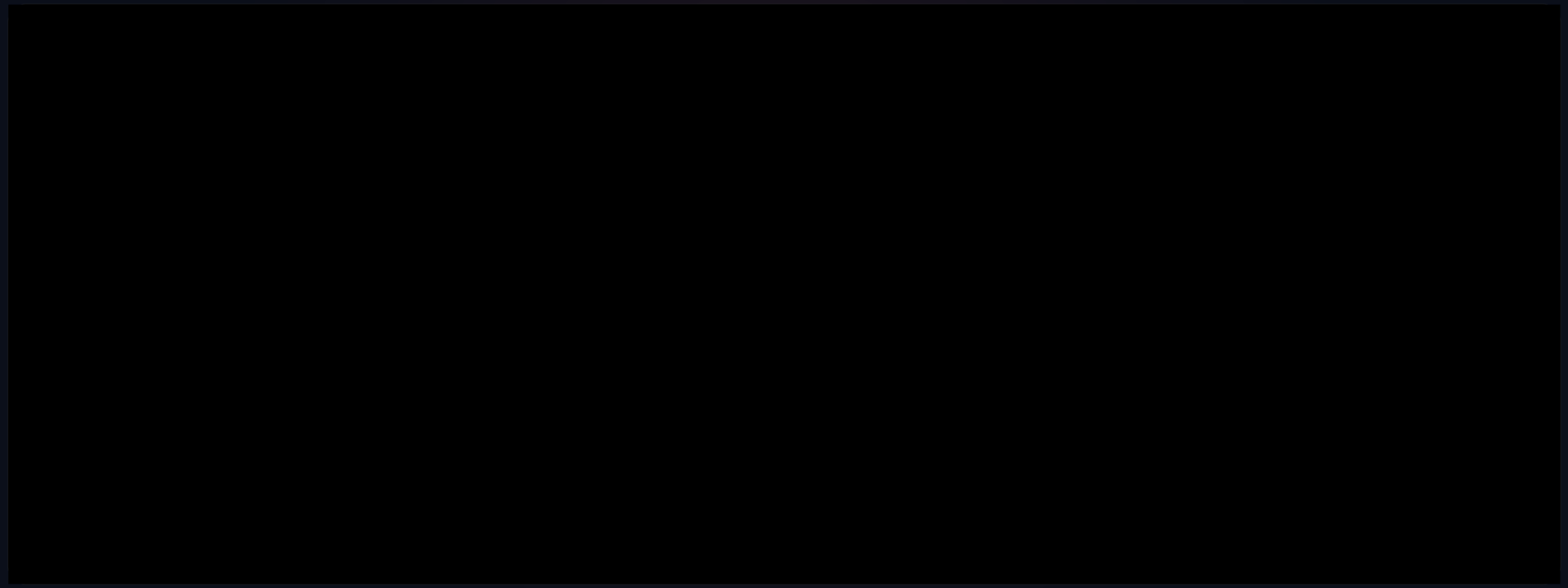
Nathan McNulty

Microsoft MVP in Security · Principal Security Solutions Architect, Patriot Consulting

With over 20 years of experience, his career started on Helpdesk and moved through SysAdmin, Client Architect, Cloud Architect, and Security Architect roles, spending nearly a decade administering and securing an environment with more than 50,000 users and 90,000 devices, building a security program primarily around Microsoft's E5 Security suite.

LIVE DEMO OPENING · 0:00–0:05 opening-demo.mp4

Password resets and revoked sessions are not enough



INCIDENT RESPONSE · TRADITIONAL PLAYBOOK

unauthorized key active

The traditional playbook worked, but the passkey survived

WHAT THE PLAYBOOK DID

- ✓ Password changed
  credential rotated
- ✓ Sessions revoked
  refresh tokens invalidated

→

THE MISS · A WAY BACK IN

login.microsoftonline.com

Sign-in successful

New authentication → new tokens

not a replayed cookie

Password reset: successful

Incident response: incomplete

unauthorized key active

CORRECT THE MENTAL MODEL

The attack did not steal a passkey, it registered one

AUTHORIZATION CONTEXT

Captured session

ESTSAUTH

Session material played against the registration surface

→

WebAuthn ceremony

NEW AUTHENTICATOR

Fresh key pair

private key

public key → Entra

Generated during the ceremony, not derived from the cookie

A passkey is a public-key credential, not a password

RELYING PARTY

Holds the public key

`public key + credential ID`

verifies assertions, stores no secrets

challenge →

← signature

no shared secrets

AUTHENTICATOR

Holds the private key

`private key · never leaves`

device or passkey provider

CEREMONY 1 · CREATE()

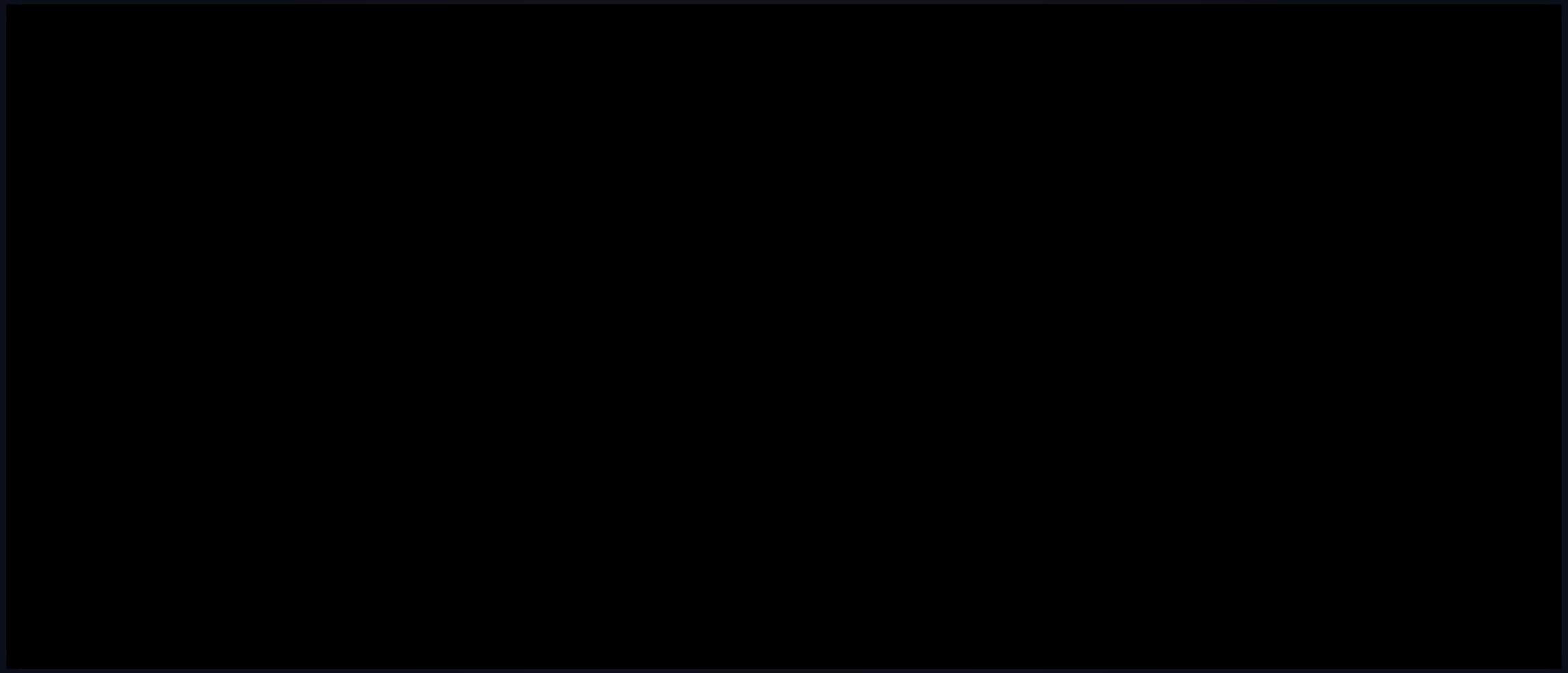
Registration creates the credential

1. RP generates challenge & creation options
2. Browser validates origin, invokes authenticator
3. Authenticator creates the key pair
4. Returns public key, credential ID, flags, attestation
5. RP validates origin, RP ID, flags, policy
6. Entra stores public key + metadata

The stored credential record is the durable object - and the attacker's goal

passkey-registration.mp4

How passkey registration works



CEREMONY 2 · GET()

Authentication proves control of the credential

1

RP issues a fresh challenge

→

2

Authenticator authorizes locally (UP/UV)

→

3

Private key signs the challenge

`key stays in the authenticator`

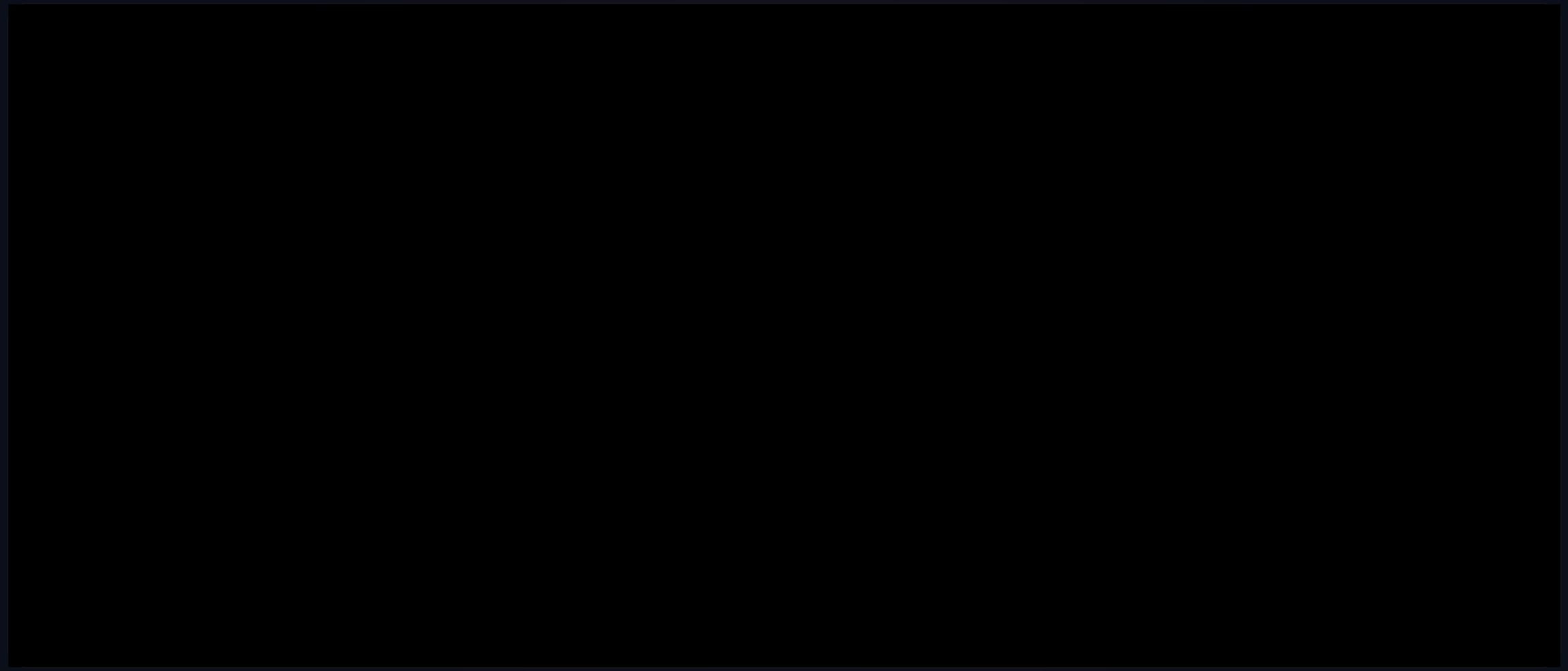
→

4

Entra verifies & evaluates counter, policy

passkey-use.mp4

How passkey authentication works



WHY PASSKEYS RESIST PHISHING

The browser binds the ceremony to the origin

`login.microsoftonline.com`	requests the credential	✓ assertion succeeds
`login.micros0ft-sso.com`	asks for the same credential	✗ never offered

> But our attack occurs after a valid session reaches the registration plane - downstream of origin binding, not against it

TWO CEREMONY FLAGS

User presence and user verification are different signals

UP User presence

Evidence that an interaction occurred - a touch or tap on the authenticator.

`a flag · not identity`

UV User verification

Local verification - PIN, biometric, or device unlock - per the authenticator.

`verification occurred · not who`

UV says local verification happened according to the authenticator, but it does not send gestures to the RP or IdP

PROVENANCE

Attestation can establish authenticator provenance

Connects claimed properties to a trust in the implementation stack or model, not unique per device or person.

THE LIMITS

Attestation does not answer every security question

ESTABLISHES

- Authenticator provenance
- Model & declared metadata
- Policy eligibility

DOES NOT ESTABLISH

- Who is operating it now
- Live device health
- Actual transport in use
- Current backup location
- Future UV events

Some signals exist in authenticator data - but without a trusted chain they are not strong, independently verified claims

CREDENTIAL MODELS

Synced passkeys trade provenance for recoverability

DEVICE-BOUND

One authenticator

Stays on a single device. Strongest device provenance; no portability.

`strong provenance`

SYNCED

Across devices

Backup-eligible; syncs through a provider. More recoverable, weaker device provenance.

`still phishing-resistant` `portable`

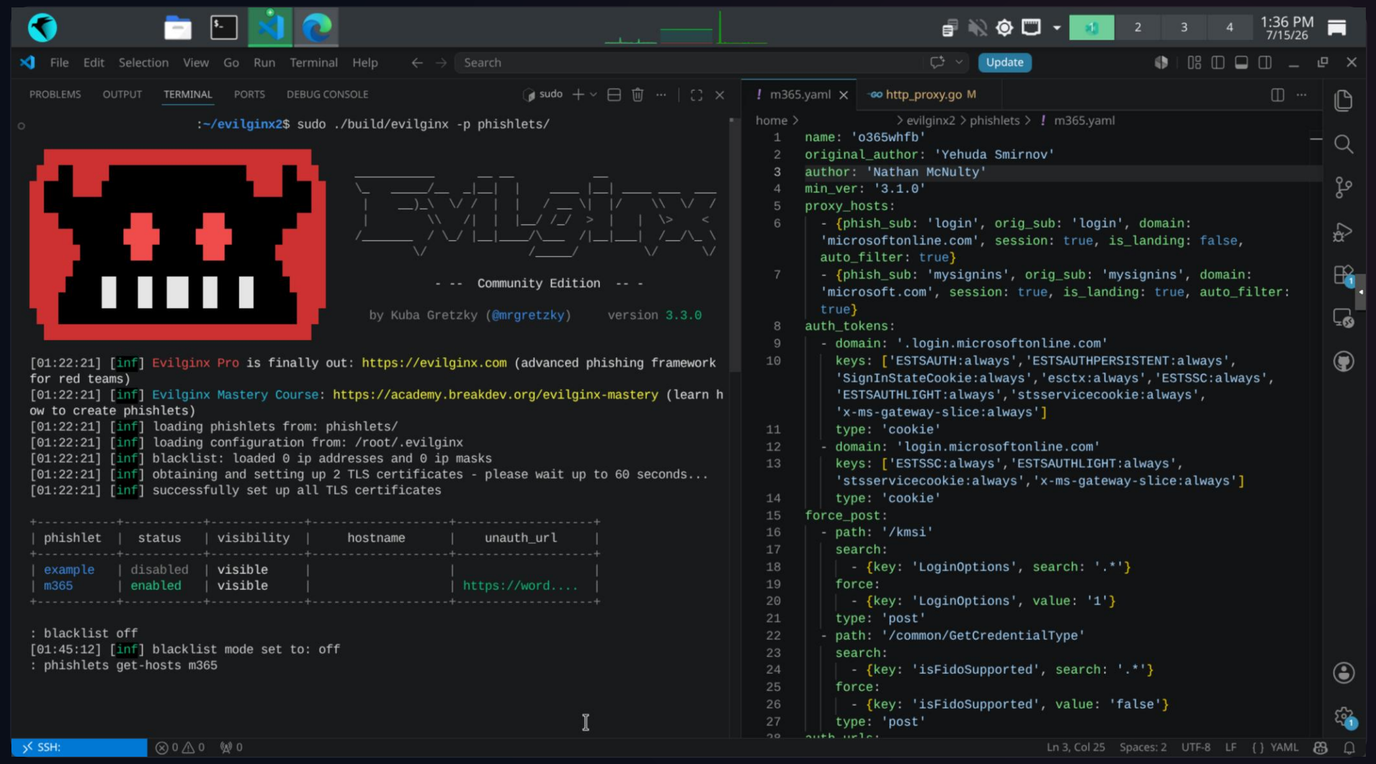
Read AAGUID, backup flags, provider behavior, attestation, and Entra policy together

LIVE DEMO ATTACKER · INFRASTRUCTURE

fallback: attacker-infra.png

RECONNAISSANCE · ATTACKER SETUP

The infrastructure behind the phish



SHARED EGRESS IP

A **NAT gateway** gives the VM and the Function App a single shared outbound IP - both halves egress from one address

REGION & ROUTING

Deploy in **any region**, with the potential for **special routing** to shape where traffic appears to originate

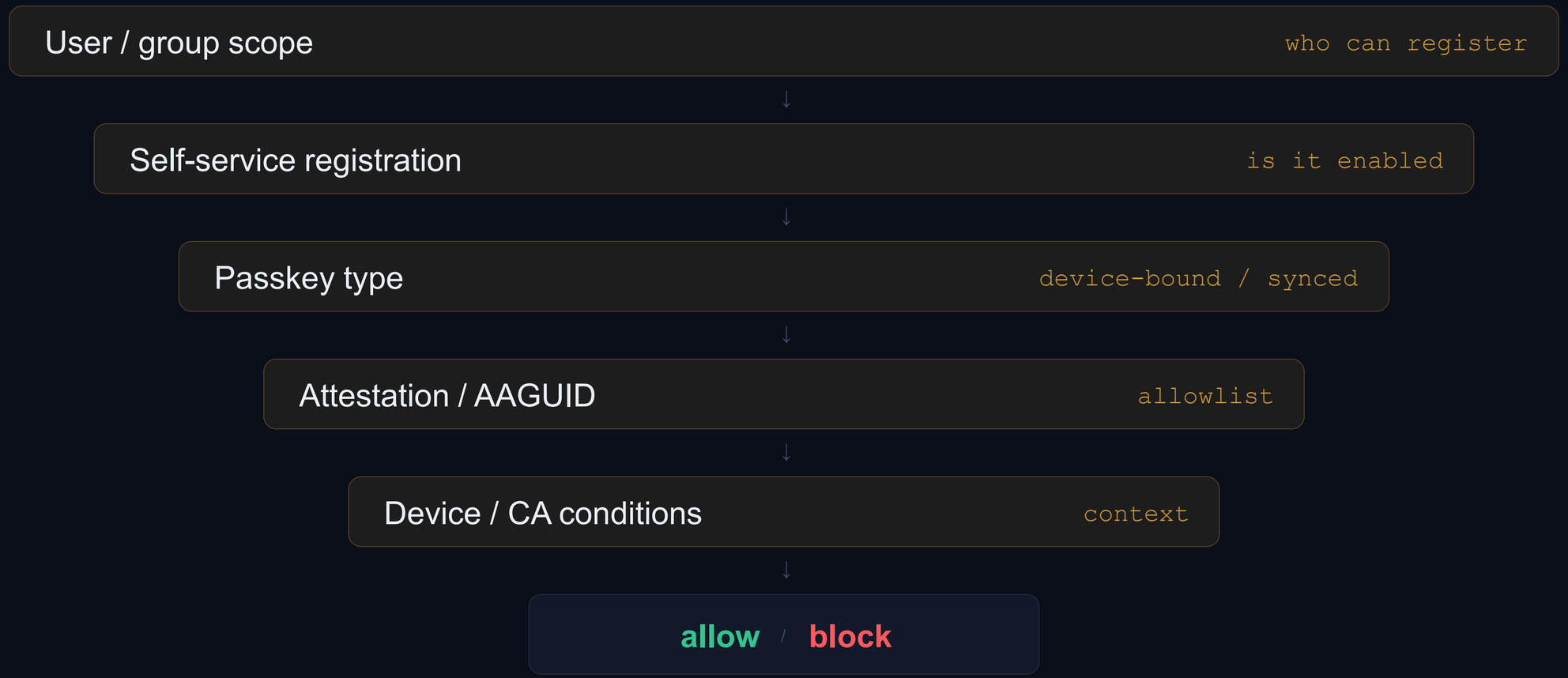
GUARDRAILS CUT BOTH WAYS

Mature cloud **security controls** can prevent the operator's own mistakes - tradecraft slips get caught too

ENTRA CONFIGURATION

Entra policy decides whether registration is practical

Authentication methods
› Passkey profiles

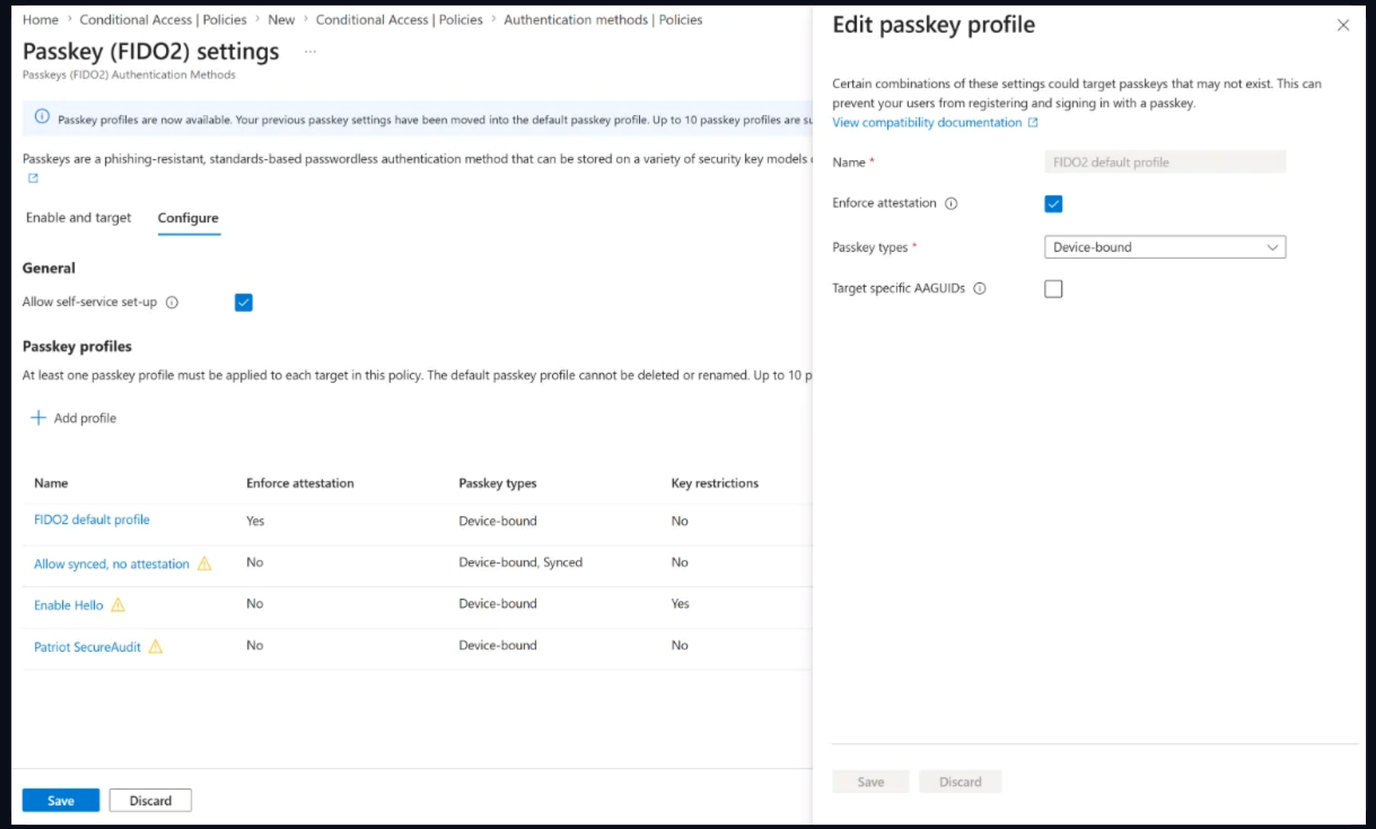


LIVE DEMO ENTRA PORTAL · PASSKEY PROFILES

fallback: passkey-profiles.png

ENTRA · PASSKEY (FIDO2) SETTINGS

Profiles decide what a passkey must prove



ENFORCE ATTESTATION

Requires a verifiable attestation chain, establishing authenticator provenance. Turn it **off** and any key can register.

PASSKEY TYPES

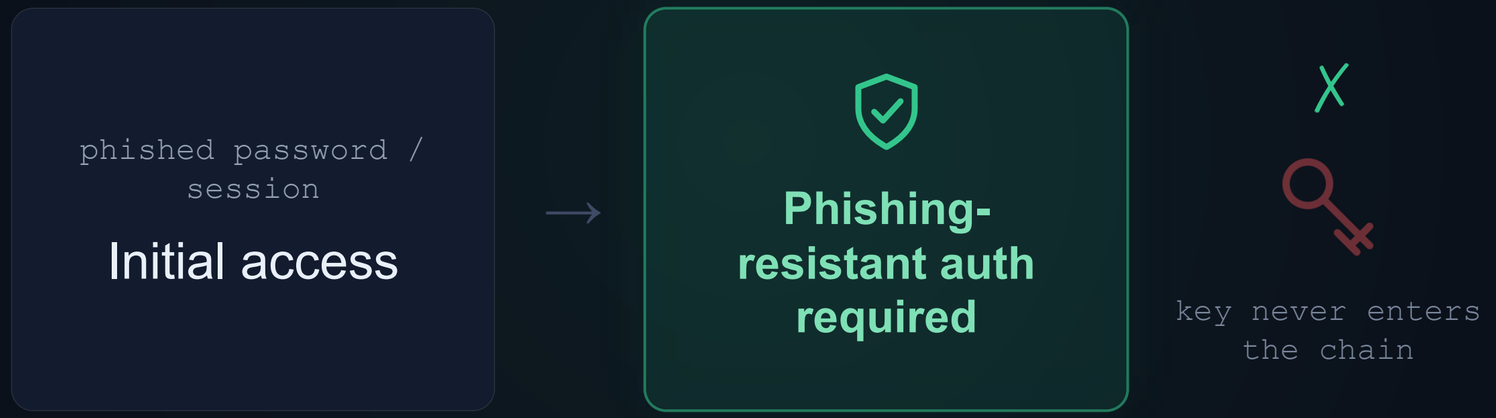
Device-bound stays on one authenticator; **synced** roams a cloud account — recoverable, but no hardware provenance.

KEY RESTRICTIONS · AAGUID

Allowlist authenticator models by AAGUID so only trusted hardware can enroll a credential.

CONTROL HIERARCHY · THE STRONGEST

The strongest control prevents the phish



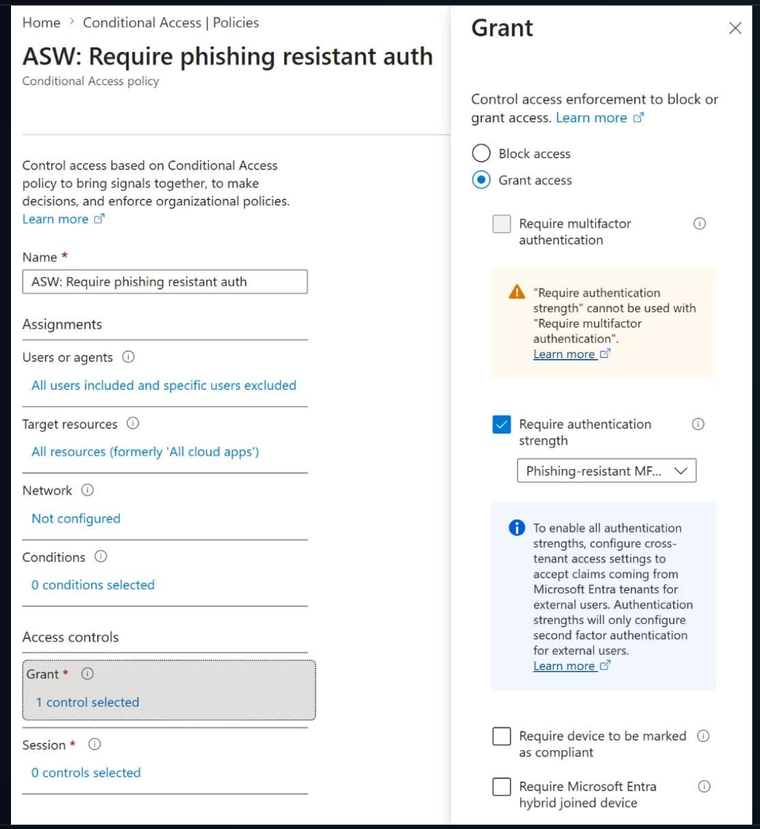
Registration controls are important - but they do not replace preventing the initial phish

LIVE DEMO ENTRA PORTAL · CONDITIONAL ACCESS

fallback: auth-strength.png

CONDITIONAL ACCESS · AUTHENTICATION STRENGTH

Require phishing-resistant auth on the resource



GRANT CONTROL

An **authentication strength** grant requires phishing-resistant MFA - passkey / FIDO2, Windows Hello, or certificate-based auth

TARGET ALL RESOURCES

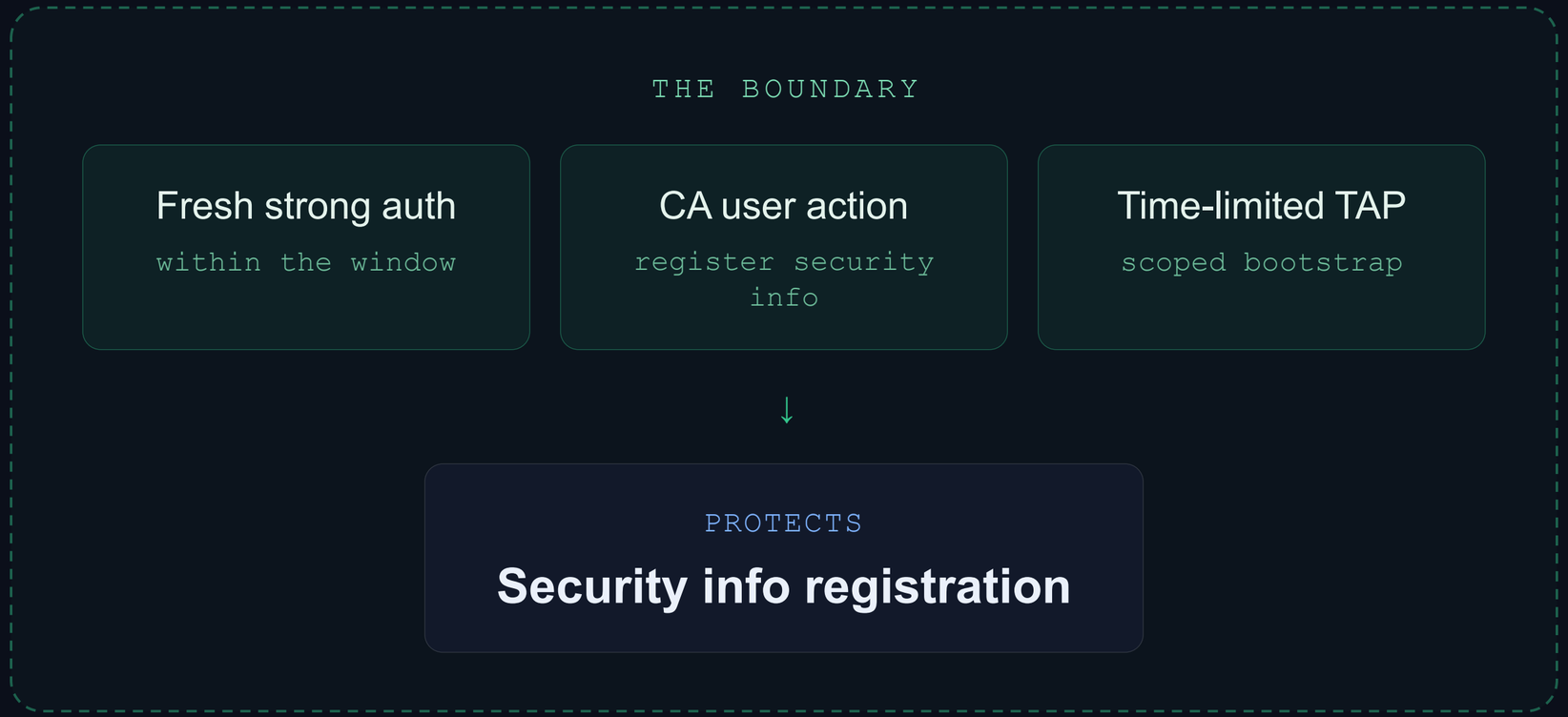
Scoping policy only to **sensitive apps** can leave gaps, especially for FOCI and BroCI resources

RESULT

A phished password or stolen session **never satisfies** the policy, so the attack is limited

CONTROL HIERARCHY · CONSTRAIN REGISTRATION

Registration needs its own security boundary



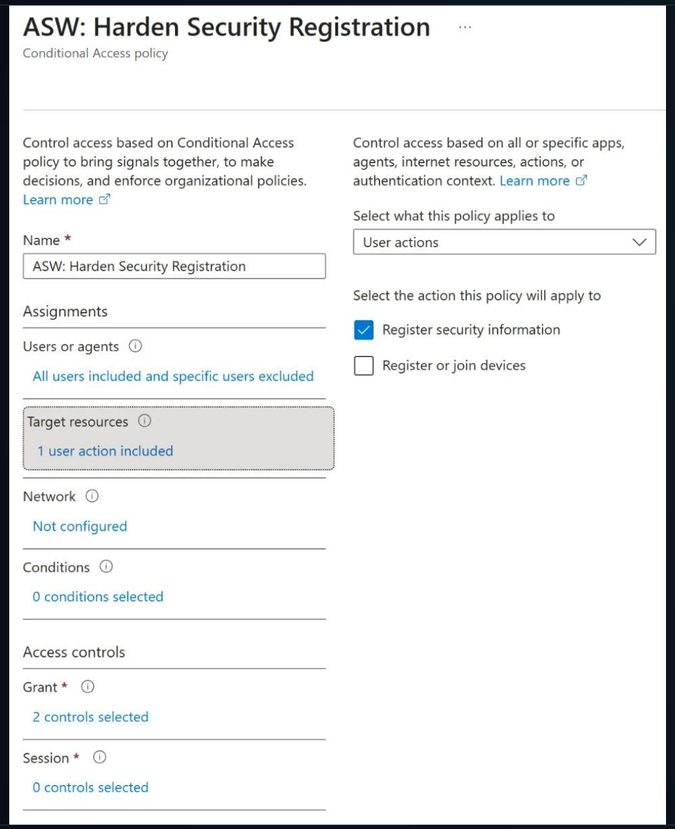
TAP is a controlled bootstrap for registration, not a universal replacement for Conditional Access

LIVE DEMO ENTRA PORTAL · CONDITIONAL ACCESS

fallback: register-securityinfo.png

CONDITIONAL ACCESS · USER ACTION

Protect the "Register security information" action



TARGET

The **user action** fires when a user registers or changes security info, separate from app sign-in

GRANT

Require **authentication strength** - phishing-resistant auth or TAP to add or change an authenticator

SESSION

Set **sign-in frequency** to "every time" so a stolen session alone can't silently enroll a key

DEVICE CONTEXT

Device state adds context to the identity

Registered

A relationship with Entra. Not the same trust as a managed corporate device.

Joined / hybrid

Stronger organizational device context for the identity.

Compliant

Satisfies your current policy evaluation . Not synonymous with joined.

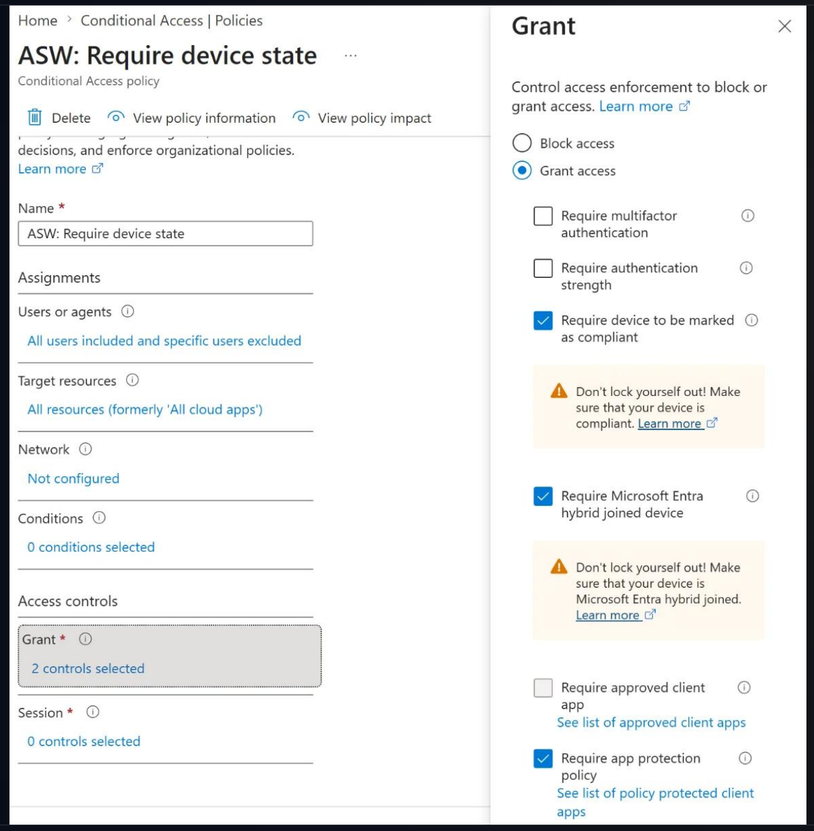
Strongest when combined with authentication strength, not relied on alone

LIVE DEMO ENTRA PORTAL · CONDITIONAL ACCESS

fallback: device-state.png

CONDITIONAL ACCESS · DEVICE CONDITIONS

Filter on device state in the policy



DEVICE FILTER

Target the policy on **device attributes** - join type, compliance, or **immutable**properties

GRANT CONTROL

Require a **compliant or hybrid-joined** device to reach the resource.

CAVEAT

Device state is **context, not a guarantee-** strongest combined with authentication strength

key survives

SESSION CONTROLS

Session controls do not remove authenticators

CAE

Reacts to events mid-session, not only at token expiry

Global Secure Access

Adds a broader network-enforcement path

Token Protection

Binds tokens to device context; cuts replay

CA App Control

Proxies sessions to monitor and block in-session actions

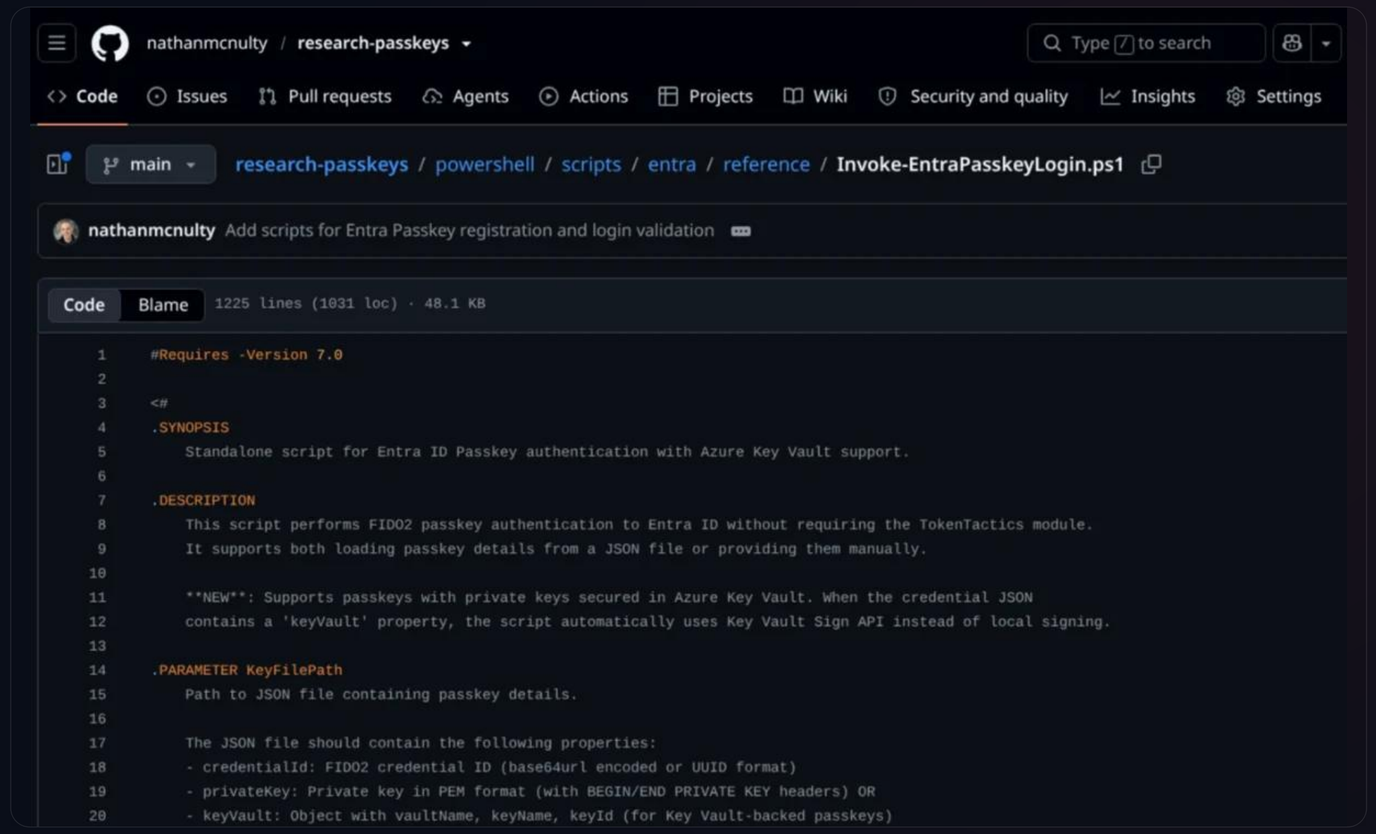
These constrain the ticket. They do not remove the key.

LIVE DEMO KEY VAULT PASSKEY · USING THE KEYS

fallback: keyvault-passkey-use.mkv

WHAT THE ATTACKER CAN DO

Using the Key Vault passkey



AZURE RESOURCES

Sign into the **portal** via extension, use Azure CLI or Azure PowerShell, or use CI/CD to run from code repositories

SCRIPTS

Manually register and use to access Graph or portals **headlessly** from a shell or CI/CD

AGENTS

Hand the identity to an **autonomous agent** that chooses its own next actions toward a goal

delegated reach

One delegated identity opens several doors

Microsoft Graph

Directory read - users, groups, roles, and lots of other resources

- sign-in + Graph audit

Microsoft 365

Read and send email, access Teams and send chats, access SharePoint sites and related resources

- Unified audit log

Azure RM

Subscriptions, resource groups, metadata.

- ARM activity log

Portal apps

Delegated portal-application access.

- app sign-in event

Actions still leave an audit trail that a responder can follow

EXTRACTION VS INVOCATION

Non-exportable does not mean low-impact

AZURE KEY VAULT

Private key cannot be exported

Can the key leave? No.

→

AUTHORIZED CALLER

Invokes sign operations

Can it still be used? Yes.

The question is **who can invoke the key**, under what conditions, and what that identity can do next

DEFENSIVE DESIGN

Human passkeys and workload identities solve different problems

HUMAN-OPERATED

Human passkey

Strong interactive authentication

poor fit for unattended use

DELEGATED · MOST FLEXIBLE

Key Vault-backed passkey

Delegated portal + app access, controlled signing

better access, stronger controls, more attack surface

CLEANEST FOR AUTOMATION

Workload identity

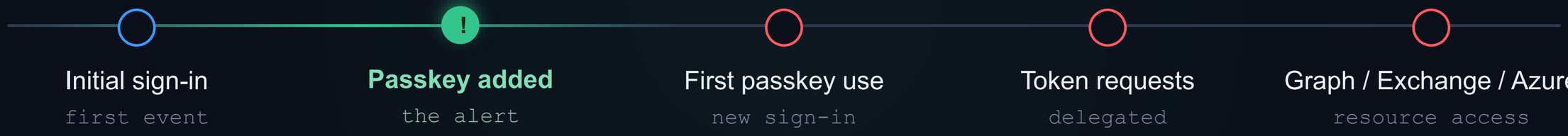
Managed identity, cert, or federated credential

separation + least privilege

Always start with a workload identity, but for tasks they can't do, consider Key Vault-backed passkeys

DETECTION

The authentication-method change may be the best alert



Catch the method change before resource access expands. Correlate identity, time, AAGUID, device, and location.

RESPONSE

Update the playbook, respond to the key

01

Reset password

Rotate the credential

removes one path

02

Revoke sessions

Invalidate refresh tokens

voids the tickets

03

Delete / disable the method

Remove unauthorized authenticators

removes the key

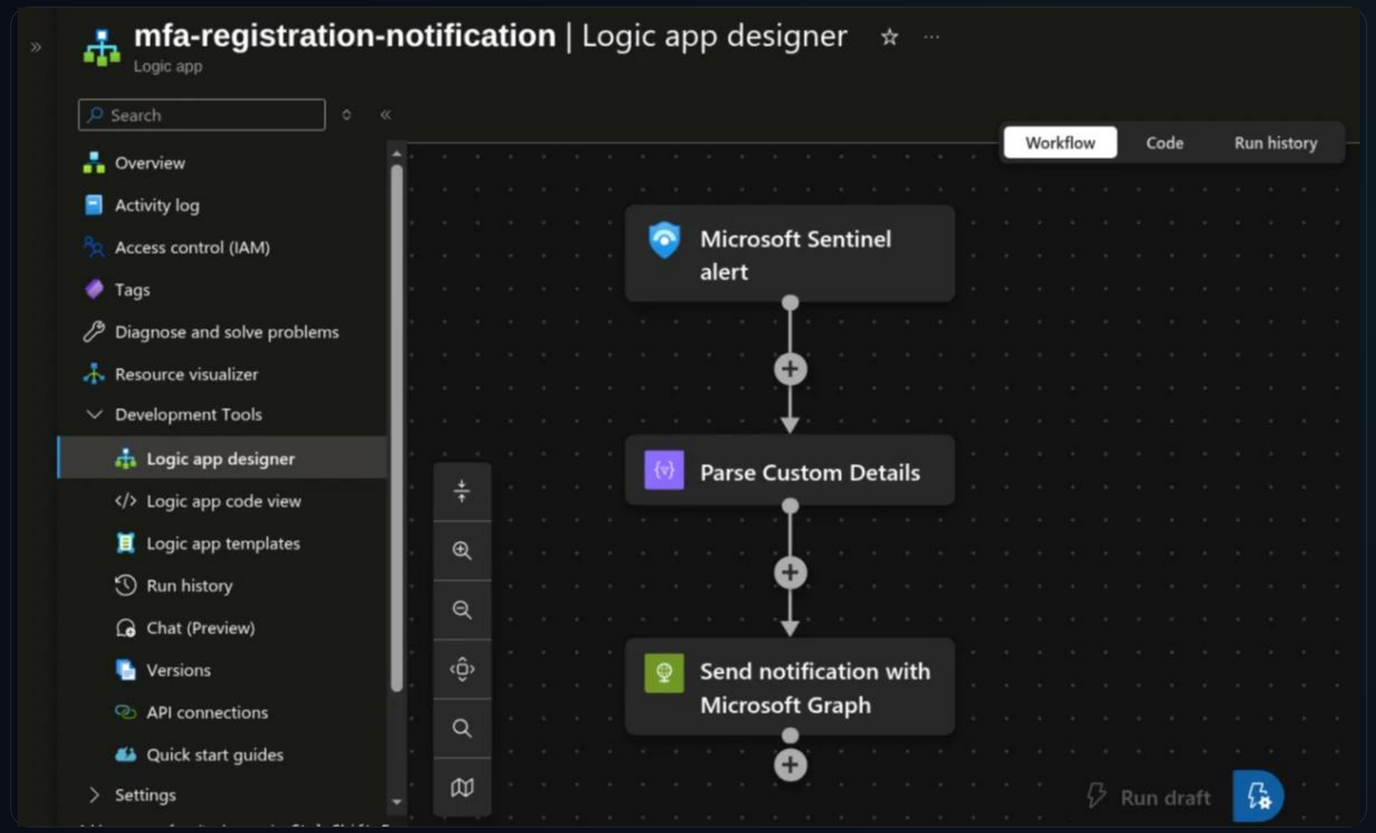
Then inventory every authenticator, review roles, consents, and service principals - hunt the pattern tenant-wide

LIVE DEMO DEFENSE · NOTIFY · DETECT · RESPOND

fallback: notify-detect-respond.mkv

THE DEFENDER'S ANSWER

Notify, detect, and respond to the key



PROACTIVE NOTIFICATIONS

Alert the moment a **passkey is added** - the authentication-method change fires before resource access expands

DETECTIONS

Correlate **identity, time, AAGUID, device, and location** with Function App and Key Vault operations to confirm the abuse

RESPONSE

Reset the password and revoke sessions, then **delete or disable the method** to remove the key - inventory every authenticator tenant-wide

CLOSING SYNTHESIS

The passkey is strong; the lifecycle still matters

- Phishing-resistant access
- Protected registration
- Trusted authenticator & device
- Session protection
- Method-change detection & removal

the chain, broken at five points

> Passkeys are not the weakness in this story. Unprotected enrollment and incomplete response are.

Q&A · 30 MINUTES

Questions and discussion

From initial access to persistence

unauthorized key removed