

Device-Code Phishing in Practice

Anatomy of a Microsoft 365 Account Takeover, from an Approved Code to Rogue Devices and a Concealment Rule

Chris Jordan, Fluency Security · October 22, 2026 · Sanitized case study

ABSTRACT. A device-code phishing lure persuaded one employee to type an attacker's code into Microsoft's real device-login page, click past Microsoft's own warning and complete MFA. That single approval issued a Microsoft Authentication Broker token to the attacker, who used it within seconds to register two rogue devices in Entra ID, signed in from Nigeria 98 seconds later in the same session the user had approved, and planted a hidden inbox rule to suppress mail from a trusted supplier. This paper walks through that intrusion end to end, using sanitized telemetry from a real customer environment ("Customer B"). It is a companion to Fluency's October 2026 paper on adversary-in-the-middle (AiTM) phishing, and the contrast is the point: there was no fake site, no proxy and no stolen password, so phishing-resistant MFA would not have helped. Automated containment cut the attacker off 3 minutes 47 seconds after the rule was created. The case was then closed as benign, and the raw logs showed otherwise.

Key findings

1. **The user completed MFA, for the attacker.** The approval happened on the user's own laptop, on the genuine Microsoft page. No password was stolen and nothing was relayed. Microsoft issued the tokens to whoever requested the code.
2. **The warning fired and was clicked through.** Microsoft displayed *"Are you trying to sign in to Microsoft Authentication Broker?"* (error 50199, `CmsiInterrupt`). The user clicked Continue eight seconds later.
3. **The token became persistence in under a minute.** The attacker registered a first rogue device 5 seconds after the approval and a second, Entra-joined device at 54 seconds, then signed in as that device from a Nigerian mobile carrier at 98 seconds, carrying the user's own Entra session ID.
4. **The attacker knew whom to silence.** The concealment rule targeted a specific supplier whose name appeared in nothing the attacker had opened. That implies prior knowledge, consistent with a lure sent from the supplier's compromised mailbox.
5. **Containment was fast; the closure was wrong.** Tenant automation disabled the account 3 minutes 47 seconds after the rule. Fluency's AI Assistant recommended escalation, but an analyst closed the case as "Benign Positive" six minutes later. The rogue devices were disabled, not deleted.
6. **One policy would have stopped it.** In the 14 days around the attack, the victim was the only non-administrator on the tenant to use the device-code flow. Blocking it with Conditional Access, excluding a short admin list, breaks nothing and removes the attack.

1 How device-code phishing works

Device-code phishing abuses a legitimate OAuth sign-in flow, so the attacker never needs to steal a password or defeat MFA. The OAuth 2.0 device authorization grant was designed for devices that cannot show a browser, such as smart TVs, conference-room equipment and command-line tools. The device asks Microsoft for a short user code; the user enters that code at `microsoft.com/devicelogs` on a second device and signs in; Microsoft then issues tokens to the device that asked. Nothing in the flow ties the person who types the code to the device that requested it.

The five stages are shown in Figure 1:

1. **Request.** The attacker's machine asks Microsoft for a user code for a chosen client application, here Microsoft Authentication Broker.
2. **Lure.** An email, chat or text tells the victim to enter the code, typically to release held mail, join a meeting or view a shared document. The code is valid for about 15 minutes, so the lure must be acted on quickly.
3. **Approval.** The victim enters the code on the real Microsoft page, clicks past the confirmation prompt and completes password and MFA.
4. **Token issued.** Microsoft returns an access token and a refresh token to the attacker's polling session.
5. **Persistence and abuse.** The attacker uses the tokens directly, or converts them into something longer-lived, such as a registered device.



Figure 1. The five stages of a device-code phishing attack: request, lure, approval, token issued, persistence and abuse. Unlike AiTM, every page the victim sees is genuine.

Why the Authentication Broker. The attacker did not request a code for an arbitrary app. Microsoft Authentication Broker (client ID `29d9ed98-a469-4536-ade2-f981bc1d605e`) is a first-party client whose tokens can be exchanged at the Device Registration Service to register or join a de-

vice to Entra ID. A registered device can then obtain a Primary Refresh Token (PRT), a device-bound credential that signs in to Microsoft 365 apps without asking for MFA again. Microsoft Threat Intelligence documented this shift to the Broker client in device-code campaigns in February 2025.

Why MFA did not help. In AiTM phishing the victim completes MFA through a proxy, and phishing-resistant methods break the attack because the credential is bound to the real domain. In device-code phishing the victim completes MFA on the real domain, so passkeys, Windows Hello and number matching all succeed as designed (Table 1). What breaks the attack is blocking the flow itself.

Table 1. Device-code phishing compared with AiTM phishing.

	AiTM phishing	Device-code phishing
Fake site or reverse proxy	Yes	No: the real Microsoft device-login page
Password stolen	Yes	No
Who completes MFA	Victim, through the proxy	Victim, directly with Microsoft
What the attacker gets	Session cookie and tokens	Access and refresh tokens for the client the attacker chose
Where the authentication comes from	The proxy's IP	The victim's own device and IP
Stopped by passkeys / phishing-resistant MFA	Yes	No
Stopped by blocking the device-code flow	No	Yes

ATT&CK classifies the approval as T1528 (Steal Application Access Token), the device registration as T1098.005 (Account Manipulation: Device Registration), and the later use of the token as T1550.001 (Use Alternate Authentication Material: Application Access Token).

2 Case study: Customer B

One non-privileged user was compromised; the attacker held access for about seven minutes and was active inside the mailbox for under two. Customer B is a mid-size organization outside the US whose security operations are run by a managed security partner on the Fluency platform. Fluency ingested the Office 365 unified audit log (Azure Active Directory, Exchange and general workloads) and Entra user records. Entra sign-in logs were not streamed, so every sign-in in this paper comes from the unified audit log's `UserLoggedIn` and `UserLoginFailed` records.

Table 2. Case summary.

Item	Detail
Victim	"User B", approver and recipient on supplier invoice and purchase-order threads; no directory roles; account about ten weeks old; no job title or department in Entra
Victim device	The user's own Windows laptop, Entra-registered, Edge browser, not marked compliant; on a mobile ISP at the time
Lure	Day 0, about 11:54 UTC; opened 12:06:46
Code approved	Day 0, 12:12:06 UTC, MFA completed

Item	Detail
Attacker active	12:13:44 – 12:15:11 UTC, 13 events, one IP
Containment	Automated disable and token revocation at 12:18:58 UTC, 6 min 52 s after approval
First alert	Fluency incident, risk score 7,200, at 12:24:14 UTC
Disposition	Closed as "Benign Positive" at 12:48:22 UTC; Fluency's follow-up investigation found it a true positive

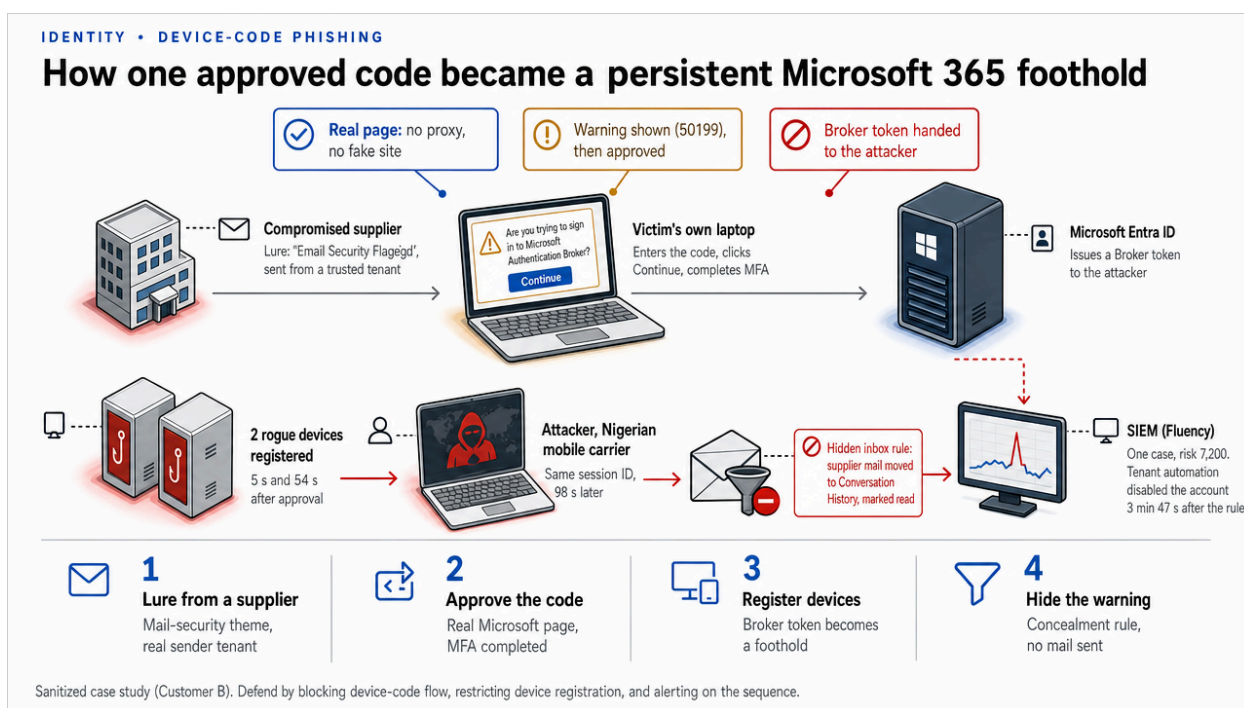


Figure 2. The Customer B attack chain: a lure from a trusted supplier tenant, an approval on the victim's own laptop, two rogue devices in 54 seconds, a foreign sign-in in the user's own session, and a rule that hides the supplier's mail. Everything after the approval took under three minutes.

Sanitization. The customer, its managed security partner, its supplier, user and administrator names, domains, tenant and object identifiers, device names, session, message and rule IDs, the victim's IP addresses and internet providers, and the subjects of business email are removed or replaced with labels. Times are UTC, on a single day labelled Day 0. The attacker's network is described by country and carrier type only. Microsoft application IDs, error codes, audit operation names, request types and the inbox-rule parameters are reproduced exactly: they describe the technique and the attacker's tooling, not the victim, and they are the most reusable detection material in this paper.

3 Phase 1: Setup and lure

The lure was a mail-security notice from another Microsoft 365 tenant in the customer's own country, most likely a compromised mailbox at one of Customer B's suppliers, opened 5 minutes before the code was entered.

Target selection. User B was not picked by title: the Entra profile had no job title, department or company. What the mailbox did show was a steady flow of supplier invoices, purchase-order approvals and payment confirmations. The most likely explanation is that the attacker had already

compromised a supplier mailbox and worked through its contacts, which puts anyone who corresponds with suppliers about money on the list (T1199 Trusted Relationship).

Table 3. Day 0 activity leading to the approval.

Time (UTC)	Event	Basis
05:33	User B uses the Microsoft quarantine portal and deletes about 24 held messages	Confirmed
~11:54	"Email Security Flagegd – [Customer B]" (misspelling original) arrives from another Microsoft 365 tenant in the same country	Arrival confirmed; sender inferred
11:54–12:10	Normal work: invoice approvals, purchase orders, replies	Confirmed
12:06:46	User B opens the lure in Outlook on the web	Confirmed
12:10:22	User B opens a "Health and Safety Audit" message, the last message read before the approval	Confirmed; role unknown

Tells in the lure

- **A familiar theme.** The user had spent part of that morning in the quarantine portal. A notice that mail was flagged and needed a code to release would have looked like more of the same.
- **Nothing to scan.** A device-code lure needs no malicious link and no attachment, only instructions and a code. The only URL it has to mention is Microsoft's own device-login page. URL detonation and attachment sandboxing have nothing to inspect.
- **A trusted sender tenant.** The message came from a real Microsoft 365 tenant, not a newly registered domain, so sender-reputation and domain-age checks passed. The misspelled subject was the only visible flaw.
- **A time limit.** A user code expires after about 15 minutes. The five minutes from opening the lure to entering the code fits that window.

Caveat. Message trace data was not in the audit stream, so the sender address, the message body and any other recipients have not been confirmed from logs. Section 7 lists the trace as the most decisive open check.

4 Phase 2: The handover (Day 0, 62 seconds)

User B entered the attacker's code on their own laptop at 12:11:58, saw Microsoft's warning, approved it with MFA eight seconds later, and the attacker registered two devices within the next 54 seconds.

Table 4. The handover. Every row is from the unified audit log.

Time (UTC)	Source	Event	Result
12:11:58	User B's laptop, mobile ISP	Code entered at the device-login page for Microsoft Authentication Broker. Microsoft shows "Are you trying to sign in to Microsoft Authentication Broker?"	UserLoginFailed, 50199 CmsiInterrupt
12:12:06	User B's laptop	User clicks Continue and completes MFA. The attacker's polling session receives a Broker token	Success, RequestType Cmsi:Cmsi
12:12:11	Device Registration Service	Rogue device 1 registered, User B added as registered owner	Success

Time (UTC)	Source	Event	Result
12:13:00	Device Registration Service	Rogue device 2 registered as Entra-joined; it can now obtain a PRT	Success (PRT inferred)
12:13:44	Nigerian mobile carrier	Sign-in as rogue device 2, Chrome, same Entra session ID as the 12:12:06 approval	Success, RequestType Login:reprocess

Tells in the handover

- **The 50199 interrupt.** `CmsiInterrupt` is Microsoft asking the user to confirm the app they are signing in to. On its own it is common, because legitimate apps and admin tools trigger it. A 50199 followed within seconds by a successful device-code sign-in to the Broker, by a user who is not an administrator, is the signature of this attack.
- **Device registration seconds after sign-in.** Five seconds between approval and the first device registration is not human. A real user enrolling a laptop goes through Windows setup or a company portal and does not finish it in five seconds from a device-code prompt.
- **Rogue device names.** Both rogue devices were named `DESKTOP-` followed by six hexadecimal characters. Windows-generated names use seven characters, and neither matched the tenant's naming convention. Baseline before alerting, but a newly registered device whose name matches nothing in your fleet deserves a look.
- **One session, two countries.** The attacker's sign-in carried the same Entra session ID as the user's approval, from a different country 98 seconds later. That is the direct technical link between the user's click and the attacker's access.

Why this was not AiTM. Customer B's first triage considered impossible travel and credential stuffing. Table 5 tests the AiTM explanation against the logs; every indicator points to device-code phishing.

Table 5. AiTM or device code: what the logs show.

Indicator	Expected for AiTM	Observed here
Where authentication happens	The phishing proxy's IP, with the victim's credentials relayed	The user's own registered laptop and normal ISP
Sign-in interrupt	None specific	50199 <code>CmsiInterrupt</code> , the device-code confirmation prompt
Client application	Usually OfficeHome or Outlook on the web	Microsoft Authentication Broker, a known device-code abuse client
Post-authentication action	Stolen session cookie replayed	Two devices registered within 5 and 54 s, then a sign-in as a registered device
Session linkage	Proxy session differs from any victim session	Attacker sign-in carries the same session ID as the approval
Password guessing	None	None; no failed-password attempts preceded the approval

Why device registration matters. Revoking sessions invalidates the refresh tokens from the sign-in. It does not remove a registered device. Until the rogue devices are deleted they are part of the account, and an Entra-joined device holding a PRT is the reason the attacker chose the Broker in the first place.

5 Phase 3: Inside the mailbox

The attacker spent 87 seconds in the mailbox: one default inbox load, then one concealment rule. No mail was sent, forwarded or exported.

Table 6. Attacker activity, all from one Nigerian mobile-carrier IP.

Time (UTC)	Operation	Detail
12:13:44	UserLoggedIn	Sign-in as rogue device 2, Login:reprocess , user's session ID
12:14:04	UserLoggedIn	Exchange Online, Outlook on the web
12:14:31	MailItemsAccessed x8 (Bind)	Default view loads about 60 recent Inbox and Sent Items messages, including invoice and purchase-order approvals and the lure itself. Nothing mentioning the supplier.
12:15:11	New-InboxRule	Concealment rule (below)

The concealment rule

Operation:	New-InboxRule	(Outlook on the web)
Name:	;	(no visible name)
SubjectOrBodyContainsWords:	admin@<supplier-domain> ; @<supplier-domain>	
MoveToFolder:	Conversation History	
MarkAsRead:	True	
StopProcessingRules:	True	

Any message mentioning the supplier's admin address or domain is moved to Conversation History, a folder Outlook creates for Teams and Skype history and that almost nobody opens, marked as read, and no further rules run. Three details mark it as deliberate:

- **No visible name.** A rule named `;` renders as blank or near-blank in the Outlook rules list. Punctuation-only names are a well-known BEC habit and are easy to alert on.
- **An obscure destination.** Deleting mail leaves a gap the user may notice; moving it to an existing folder they never open hides it without a trace in the Inbox.
- **Prior knowledge.** The supplier's domain did not appear in any message the attacker loaded. The attacker brought it with them, which fits a lure sent from that supplier's compromised mailbox.

Intent. The most likely purpose was to suppress the supplier's own warning, such as "our mailbox was hacked, ignore our last email", so the phish would not be exposed. With that in place, the inbox already held live invoice and payment threads, an ideal setting for invoice or bank-detail fraud. MITRE: T1564.008 (Hide Artifacts: Email Hiding Rules) and T1114.002 (Remote Email Collection).

Telemetry gap. Exchange search auditing (`SearchQueryInitiatedExchange`) was not producing events for any user on the tenant. "No search seen" is therefore not proof that the attacker did not search the mailbox.

6 Phase 4: Containment and the closure

Tenant automation cut the attacker off within four minutes of the rule. The response that followed left persistence partly open, and the case was closed as benign.

Table 7. Containment and response, Day 0.

Time (UTC)	Actor	Event
12:18:58	Tenant automation (service principal)	Account disabled and refresh tokens revoked, 3 min 47 s after the rule. No further attacker activity
12:22–12:27	User B	Six <code>UserDisabled</code> sign-in failures from the user's own laptop
12:24:14	Fluency	Incident raised, risk score 7,200
12:35:38	SOC analyst	Case acknowledged
12:42:59	Fluency AI Assistant	Recommends ESCALATE : strong indicators of account compromise
12:48:22	SOC analyst	Case closed as Benign Positive , "No risk was found"
12:48–12:50	Administrator	Sessions revoked, account re-enabled, sessions revoked again
12:55	Administrator	Both rogue devices disabled, not deleted
12:58:09	Service principal	Password reset and tokens revoked
13:01–13:02	Service principal	Four "Update user" changes, most likely MFA method re-registration
13:28–14:20	User B	Signs back in from usual locations. No further activity from Nigeria

Why the case was closed wrongly

- **The lockouts looked like an attack.** The AI Assistant's summary described "five failed logins followed by a successful sign-in" and suggested credential stuffing. The failures were the 50199 interrupt and the real user being refused with `UserDisabled` after containment. The assistant still reached the right verdict, but the wrong explanation made the case look like noise to a reader in a hurry.
- **The summary rounded the timeline.** Bucketed counts hid the order of events. Read in sequence, the raw records show approval, registration, foreign sign-in and rule inside three minutes, which no benign scenario explains.
- **Containment felt like closure.** With the account disabled and the password reset, the case looked finished. It was not: two registered devices remained, the lure had not been traced, and the supplier had not been warned.

What revocation did and did not do. Disabling the account and revoking refresh tokens stopped the attacker immediately. A disabled device cannot obtain tokens, but it is still a device object owned by the user and can be re-enabled by anyone with the right role, accidentally or otherwise. Deletion is the only clean end state.

7 Detection tells and hunting queries

Fluency's behavior rules caught the foreign sign-in and the first-ever inbox rule and raised one incident; the earlier and more specific signals were in the device-code approval and the device registrations. Table 8 ranks the tells by how early they fire and how specific they are. The queries are written for Microsoft Sentinel / Log Analytics and should be validated against the reader's own workspace schema before deployment.

Table 8. What fired, and what should have.

#	Tell	Source	Would have fired	Specificity
—	<i>Fired:</i> first sign-in from a new country and new ISP (sub-score 2,600); first inbox rule by this user (1,000); combined case risk 7,200	Fluency behavior rules	12:24 (incident)	Medium
T1	50199 followed by a successful device-code sign-in to the Broker, non-admin user	SigninLogs / UAL	12:12:06	Very high
T2	Device registered to a user within 30 minutes of a device-code sign-in	AuditLogs + SigninLogs	12:12:11	Very high
T3	Sign-in from a device registered less than an hour earlier, from a new country	SigninLogs + AuditLogs	12:13:44	High
T4	One session ID seen from two countries or ASNs within 10 minutes	Sign-in logs (interactive + non-interactive)	12:13:44	High
T5	Inbox rule with a punctuation-only name, or MarkAsRead + StopProcessingRules + move/delete on an outside address	OfficeActivity	12:15:11	High
T6	Case closed as benign while the AI verdict is ESCALATE	SOC case platform	12:48:22	Process control

Baseline first: who uses the device-code flow at all? In most tenants almost nobody legitimately does. On Customer B, User B was the only Broker device-code event in 14 days; the other ten device-code sign-ins were the managing partner's administrators using Graph and command-line tools from known networks. This list is also the exclusion group for the block policy in Section 8.

```
SigninLogs
| where TimeGenerated > ago(30d)
| where AuthenticationProtocol == "deviceCode"
| summarize SignIns = count(), Apps = make_set(AppDisplayName),
              IPs = make_set(IPAddress) by UserPrincipalName
| order by SignIns desc
```

T1: Warning shown, then approved, for the Broker. The device-code filter is what turns a noisy error code into a strong signal.

```
let warned = SigninLogs
| where ResultType == "50199"
| project WarnedAt = TimeGenerated, UserPrincipalName, SessionId;
SigninLogs
| where ResultType == "0" and AuthenticationProtocol == "deviceCode"
| where AppId == "29d9ed98-a469-4536-ade2-f981bc1d605e"
| join kind=inner warned on UserPrincipalName, SessionId
| where TimeGenerated between (WarnedAt .. WarnedAt + 10m)
| project WarnedAt, TimeGenerated, UserPrincipalName, IPAddress, SessionId, UserAgent
```

T2: Device-code sign-in, then a device registration. Legitimate device-code use almost never registers a device.

```

let dc = SigninLogs
| where AuthenticationProtocol == "deviceCode" and ResultType == "0"
| project SignIn = TimeGenerated, UPN = tolower(UserPrincipalName), IPAddress, AppDisplayName;
AuditLogs
| where OperationName in ("Register device", "Add registered owner to device")
| mv-expand T = TargetResources
| extend UPN = tolower(tostring(T.userPrincipalName))
| join kind=inner dc on UPN
| where TimeGenerated between (SignIn .. SignIn + 30m)
| project SignIn, TimeGenerated, OperationName, UPN, IPAddress, AppDisplayName, TargetResources

```

T4: One session, two countries.

```

union SigninLogs, AADNonInteractiveUserSignInLogs
| where ResultType == "0" and isnotempty(SessionId)
| summarize Countries = make_set(Location), IPs = make_set(IPAddress),
    First = min(TimeGenerated), Last = max(TimeGenerated)
    by SessionId, UserPrincipalName
| where array_length(Countries) > 1 and Last - First < 10m

```

T5: Concealment inbox rules.

```

OfficeActivity
| where Operation in ("New-InboxRule", "Set-InboxRule", "UpdateInboxRules")
| mv-apply p = todynamic(Parameters) on (
    summarize P = make_bag(bag_pack(tostring(p.Name), tostring(p.Value)))
| extend RuleName = tostring(P.Name)
| where RuleName matches regex @"^[W_]*$"
    or (Parameters has "MarkAsRead" and Parameters has "StopProcessingRules"
        and Parameters has_any ("MoveToFolder", "DeleteMessage") and Parameters has "@")
| project TimeGenerated, UserId, ClientIP, RuleName, Parameters

```

No Entra sign-in logs? Customer B had none, and the case was still fully reconstructed from the unified audit log. There, look for a `UserLoginFailed` record with `LogonError = CmsiInterrupt` and `ApplicationId = 29d9ed98-a469-4536-ade2-f981bc1d605e`, followed by a `UserLoggedIn` whose `ExtendedProperties RequestType` is `Cmsi:Cmsi`; then `Add registered owner to device`. by the Device Registration Service; then a `UserLoggedIn` with `RequestType Login:reprocess` and a `DeviceProperties ID` registered minutes earlier.

What correlation added. No single event looked malicious on its own: people travel, people create rules, and Microsoft shows the 50199 prompt for legitimate apps. The detection is the sequence of approval, registration, foreign sign-in and concealment rule, all within four minutes and joined by one session ID. Fluency correlated the foreign sign-in and the rule into one case within minutes; adding T1 and T2 would have moved the alert to the approval itself, 12 minutes before the incident was raised and before the attacker had signed in.

8 Defensive lessons

The controls that would have changed this outcome are, in order of impact: blocking the device-code flow, restricting device registration, a response playbook that deletes devices, and a second review whenever a human overrides an escalation.

- 1. Block the device-code flow.** In Conditional Access, use *Conditions* → *Authentication flows* → *Device code flow* with *Block*, applied to all users and excluding only the short list from the base-

line query. This single policy would have stopped the attack outright, and for most organizations it breaks nothing. Phishing-resistant MFA does not substitute for it.

2. **Restrict device registration and join.** Limit who can register or join devices in Entra, and require MFA or a compliant device for the "Register or join devices" user action. The Broker token was valuable only because it could become a device.
3. **Write a token-and-device playbook, not a password playbook.** The order is: disable or revoke sessions; **delete** every device registered to the user since the approval; remove the inbox rule and check for forwarding, delegates and OAuth consents; verify which MFA methods are registered; reset the password; and review Sent Items for the attacker window. Disabling a device is not deletion.
4. **Trace the lure and warn the supplier.** Run a message trace on the lure to confirm the sender, find every other recipient and purge the message. Warn the supplier by phone, not by email, since their mailbox is the likely source. Require call-back verification for any bank-detail change.
5. **Teach the warning.** Users should know that *"Are you trying to sign in to...?"* after entering a code they were sent means stop. No legitimate email asks a user to type a code at `microsoft.com/devicelogin` to release mail.
6. **Require a second analyst when a human overrides an escalation.** Flag any case closed as benign while the AI verdict is ESCALATE, and require analysts to read raw events in order rather than a bucketed summary before closing an identity case.
7. **Ingest the logs that carry the evidence.** Stream Entra sign-in logs (interactive and non-interactive) and audit logs, and enable Exchange search auditing (`SearchQueryInitiatedExchange`) so a mailbox search by an attacker is visible.

Appendix A Sanitized timeline

Time (UTC)	Phase	Event	Source
Before D0	Setup	Supplier mailbox compromised and contacts harvested (inferred)	—
D0 05:33	Context	User deletes about 24 messages in the quarantine portal	UAL (Exchange)
D0 ~11:54	Lure	"Email Security Flagegd" arrives from another Microsoft 365 tenant	UAL (Exchange)
D0 12:06:46	Lure	Lure opened in Outlook on the web	UAL (Exchange)
D0 12:11:58	Approval	Code entered; 50199 CmsiInterrupt for Microsoft Authentication Broker	UAL (Azure AD)
D0 12:12:06	Approval	Continue clicked, MFA completed (Cmsi:Cmsi); Broker token issued to attacker	UAL (Azure AD)
D0 12:12:11	Persistence	Rogue device 1 registered, user as owner	UAL (Azure AD)
D0 12:13:00	Persistence	Rogue device 2 registered, Entra-joined	UAL (Azure AD)
D0 12:13:44	Access	Sign-in from Nigerian mobile carrier as rogue device 2, same session ID	UAL (Azure AD)
D0 12:14:04	Access	Outlook on the web sign-in	UAL (Azure AD)
D0 12:14:31	Collection	8 MailItemsAccessed (Bind), about 60 messages	UAL (Exchange)
D0 12:15:11	Evasion	New-InboxRule named ; hiding supplier mail	UAL (Exchange)

Time (UTC)	Phase	Event	Source
D0 12:18:58	Response	Automated account disable and token revocation	UAL (Azure AD)
D0 12:22–12:27	Context	Six UserDisabled failures from the user's laptop	UAL (Azure AD)
D0 12:24:14	Detection	Fluency incident, risk 7,200	Fluency
D0 12:42:59	Detection	AI Assistant: ESCALATE	Fluency
D0 12:48:22	Response	Case closed as Benign Positive	Fluency
D0 12:48–12:50	Response	Sessions revoked; account re-enabled	UAL (Azure AD)
D0 12:55	Response	Both rogue devices disabled (not deleted)	UAL (Azure AD)
D0 12:58:09	Response	Password reset, tokens revoked	UAL (Azure AD)
D0 13:01–13:02	Response	Four user updates, likely MFA re-registration	UAL (Azure AD)
D0 13:28–14:20	Recovery	User signs back in; no further attacker activity	UAL (Azure AD)

Appendix B Indicator patterns

These describe the technique and the operator, not the victim. The single attacker IP was a mobile-carrier address used once; blocking it has little value.

Type	Value	Notes
Client application	Microsoft Authentication Broker, 29d9ed98-a469-4536-ade2-f981bc1d605e	Chosen for device registration
Sign-in sequence	UserLoginFailed 50199 CmsiInterrupt → UserLoggedIn RequestType Cmsi:Cmsi	Device-code approval after warning; 8 s apart
Device registration	Device Registration Service adds the user as registered owner 5 s and 54 s after approval	One registered, one Entra-joined
Rogue device names	DESKTOP- + 6 hexadecimal characters	Not the 7-character Windows default; not the tenant convention
Attacker sign-in	RequestType Login:reprocess, Chrome, as the newly joined device, same session ID as the approval	Consistent with a PRT
Inbox rule	Name ; ; SubjectOrBodyContainsWords on a supplier address and domain; MoveToFolder Conversation History; MarkAsRead; StopProcessingRules	Created 40 s after mailbox load
Lure	Subject "Email Security Flagegd – [company]", from another Microsoft 365 tenant	Misspelling in the original
Infrastructure	Nigerian mobile-carrier IP, first use on the tenant in 30 days	13 events, 87 seconds
Timing	Approval → device 1 in 5 s → device 2 in 54 s → foreign sign-in in 98 s → rule in 3 min 5 s	Scripted operation

Appendix C MITRE ATT&CK mapping

Tactic	Technique	Evidence in this case
Reconnaissance	T1589.002 Gather Victim Identity Information: Email Addresses	Target chosen through a supplier relationship; supplier domain known before access (inferred)
Initial Access	T1566.002 Phishing: Spearphishing Link · T1199 Trusted Relationship	Mail-security lure from another Microsoft 365 tenant, suspected compromised supplier (inferred)
Credential Access	T1528 Steal Application Access Token	Device-code approval for Microsoft Authentication Broker
Persistence	T1098.005 Account Manipulation: Device Registration	Two rogue devices registered to the user
Defense Evasion	T1550.001 Use Alternate Authentication Material: Application Access Token	Login:reprocess sign-in from the rogue device in the user's session
Initial Access / Persistence	T1078.004 Valid Accounts: Cloud Accounts	Sign-ins from Nigeria as the user
Collection	T1114.002 Email Collection: Remote Email Collection	8 MailItemsAccessed , about 60 messages
Defense Evasion	T1564.008 Hide Artifacts: Email Hiding Rules	Blank-named rule hiding supplier mail

References

1. Fluency Security, *Adversary-in-the-Middle in Practice: Anatomy of a Microsoft 365 Session Theft*, October 2026 (companion paper).
2. Microsoft Threat Intelligence, *Storm-2372 conducts device code phishing campaign*, February 2025, including the update on the move to the Microsoft Authentication Broker client and device registration.
3. Microsoft Learn, *Block authentication flows with Conditional Access policy* (device code flow).
4. Customer B incident investigation, Fluency data lake: Office 365 unified audit log (Azure Active Directory, Exchange, general), Entra user records, Fluency case record and AI Assistant triage.

Figures 1 and 2 were generated with ElevenLabs image generation from the case data described in this paper.