

Phish, Click, Breach: Hunting for a Sophisticated Cyber Attack

 techcommunity.microsoft.com/blog/microsoftsecurityexperts/phish-click-breach-hunting-for-a-sophisticated-cyber-attack/4267916



Authors:

-Gourav Khandelwal (@Gourav_Khandelwal)

-Akash Chaudhuri (@AkashChaudhuri)

-Matthew Mesa (@matthewmesa)

-Sagar Patil (@Sagar256)

-Uri Oren (@orenuri)

-Krithika Ramakrishnan (@krithikar)

Introduction

Since April 2024, we have observed a significant increase in Teams phishing attacks, which have led to endpoint-related incidents, particularly through the abuse of Remote Monitoring and Management (RMM) tools such as Quick Assist (Ref : [Threat actors misusing Quick Assist in social engineering attacks leading to ransomware | Microsoft Security Blog](#)), and other tools such as Any Desk, and Team Viewer.

Initially, the attack began with a spam flood, followed by the attacker impersonating the Help Desk on Teams. The attacker would contact the user via Teams, send a malicious link to start the RMM session, and deliver the harmful payload during the session. This would lead to hands-on keyboard activity, data exfiltration, and ultimately result in ransomware attacks.

Over time, the attack method evolved. The attackers now directly reach out to users on Teams, impersonating the service desk. Once the user accepts the Teams invite, the attacker provides a SharePoint link containing malicious payloads, which could lead to critical security breaches. Recent trends in social engineering attacks highlight this adaptability, with attackers varying their tactics based on the target. For instance, they might use a SharePoint link for one victim while opting for a different hosting platform for another on the same day. Moreover, attackers are moving beyond traditional link-based strategies by persuading users to install remote access software like AnyDesk and TeamViewer or convincing them to initiate connections via Microsoft's Quick Assist, which is installed by default in the Windows Operating System.

Microsoft continues to aggressively combat threats, such as halting notorious DarkGate, which is a very capable malware. Since December 2023, Microsoft Threat Intelligence has been tracking Storm-1674 attacker group misusing App Installers with Teams Phishing as the initial access vector (Ref : [Intel Article - Microsoft Defender](#)). In this scenario, the attacker convinces the user that they are interacting with the service desk, allowing the attacker to perform malicious activities on the device through Remote Monitoring and Management (RMM) tools. What makes this attack unique is that each attack kill chain is different, as every payload varies.

The activity is attributed to Storm-1811 and Storm-1674 by Microsoft Threat Intelligence.

In this blog, we will walk through one of the observed scenarios and discuss hunting approaches for detecting such attacks.

Attack Flow

Teams Phishing

In the majority of the attacks observed, impersonation of the IT desk in a one-on-one Teams conversation from attacker owned tenants. Attackers also call the users on Teams, create meetings and send chat messages that contain malicious URLs or attachments to through the meeting's chat feature.

The tenants were usually newly created in a span of less than 7 days. In a few scenarios, the Teams Phishing was preceded by a spam flood with more than 1000+ emails every hour. This was used to set the context for the attacker to call the user impersonating the help desk under the pretext of fixing the spam flood.

The attacks were highly targeted, with attackers focusing on at least three users per tenant through Teams phishing. By aggregating the number of users targeted by an external user from a tenant every hour, we can identify these attacks more effectively.

Hunting for Compromises

Hunt for spam flood attack

EmailEvents

| where Timestamp > ago(1d)

| where EmailDirection == "Inbound"

| make-series Emailcount = count()

on Timestamp step 1h by RecipientObjectId

| extend (Anomalies, AnomalyScore, ExpectedEmails) =
series_decompose_anomalies(Emailcount)

| mv-expand Emailcount, Anomalies, AnomalyScore, ExpectedEmails to typeof(double),
Timestamp

| where Anomalies != 0

| where AnomalyScore >= 10

Hunt for Suspicious External Teams messages

CloudAppEvents

where Timestamp > ago(1d)

```

|where ApplicationId == 28375

// This action type is recorded when a new chat is created with the user

| where ActionType == "ChatCreated"

// This field records the sender's Account Object ID, since the sender is a third party, the field
is expected to be empty

| where isempty(AccountObjectId)

// Validation for the message being sent from a Foreign tenant

| where tobool(RawEventData.ParticipantInfo.HasForeignTenantUsers) == true

| where RawEventData.CommunicationType == "OneOnOne"

// Validation that the conversation is not initiated from a guest tenant

| where tobool(RawEventData.ParticipantInfo.HasGuestUsers) == false

| where tobool(RawEventData.ParticipantInfo.HasOtherGuestUsers) == false

// Validation that the sender is not recognized. If the sender is not recognized, only the email
address is populated here

| where AccountId has "@"

```

This query can also be appended with aggregation by sender tenant to identify targeted attempts:

```

| extend TargetUserUPN = tolower(tostring(RawEventData.Members[1].UPN))

| extend TargetTenant = tostring(RawEventData.OrganizationId)

| extend AttackerTenant = tostring(RawEventData.Members[0].OrganizationId)

| extend AttackerUPN = tostring(RawEventData.Members[0].UPN)

| extend AttackerName = tostring(RawEventData.Members[0].DisplayName)

| summarize UsersTargeted =
dcount(TargetUserUPN ) by AttackerTenant, AttackerUPN, bin(Timestamp, 1h)

|where UsersTargeted >= 2

```

RMM Tools Abuse

In cases involving spam floods, the attacker will often call the user via Teams and persuade them to open the Quick Assist application (one of the most targeted RMM applications) and provide the access code. Once the user shares the code, the attacker gains access to the device. If the user also approves the "Request control" prompt, the attacker gains full control over the device.

Hunting for Compromises

Hunt for Teams Activity followed by suspicious RMM:

```
let interestingUsers = DeviceProcessEvents
| where Timestamp > ago(1h)
| where isnotempty(InitiatingProcessAccountObjectId)
|where FileName has_any ("quickassist.exe", "anydesk.exe", "teamviewer_service.exe") //
Multiple RMM tools can be abused here
| project InitiatingProcessAccountUpn;
CloudAppEvents
| where Timestamp > ago(1d)
| where Application == "Microsoft Teams"
| where ActionType == "ChatCreated"
| where isempty(AccountObjectId)
| where RawEventData.ParticipantInfo.HasForeignTenantUsers == true
| where RawEventData.CommunicationType == "OneOnOne"
| where RawEventData.ParticipantInfo.HasGuestUsers == false
| where RawEventData.ParticipantInfo.HasOtherGuestUsers == false
| where AccountId has "@"
```

```

| extend TargetUPN = tolower(tostring(RawEventData.Members[1].UPN))

| where TargetUPN in (interestingUsers )

| extend VictimTenant = tostring(RawEventData.OrganizationId)

| extend AttackerTenant = RawEventData.Members[0].OrganizationId

| extend AttackerUPN = RawEventData.Members[0].UPN

| extend AttackerName = RawEventData.Members[0].DisplayName

```

Initial Access

The Storm-1811 actor calls users on Teams, then abuses RMM tools to deploy payloads and initiate credential theft for initial access. And, the Storm-1674 actor either calls users or uses Teams chat to deliver malicious payloads via phishing links hosted on file-sharing services usually like SharePoint.

Hunting for Compromises

In addition to the Teams phishing activities recorded in CloudAppEvents telemetry, clicks on SharePoint URLs are logged in the UrlClickEvents table. Correlating suspicious signals on devices with UrlClickEvents table can help identify and highlight this activity.

Correlating URL click events on alerted devices

```

let alertedDevices = AlertEvidence

| where Timestamp > ago(1h)

| where isnotempty(DeviceId)

|distinct DeviceId;

let interestedUsers = DeviceProcessEvents

| where Timestamp > ago(1h)

| where DeviceId in (alertedDevices)

| where isnotempty(InitiatingProcessAccountUpn)

| distinct InitiatingProcessAccountUpn;

```

UrlClickEvents

| where Timestamp > ago(1d)

| where ActionType == "ClickAllowed" or IsClickedThrough != "0"

| where Workload has "Teams"

| where AccountUpn in (interestedUsers)

Credential Access

After taking control of the target user's device through RMM, the attacker executes a script under the pretext of fixing the spam flood activity. The name of the script also justifies the intent to convince the user for the next steps (Eg : Spam Filter Update). When the script is executed, it prompts the target user to provide the credentials, persuaded by the attacker.

In a few other scenarios, the attacker also redirects the user to an AiTM phishing page to complete the sign-in with MFA to compromise the session token.

Hunting for Compromises

These compromises can be identified by correlating risky sign-in attempts with Teams phishing from external tenants. The below query can be used to identify identity compromises (Adversary-in-the-middle attack) through Teams messages with malicious links/attachments as well:

let usersWithRiskySignIn = AADSignInEventsBeta

|where Timestamp > ago(1h)

|where RiskLevelDuringSignIn == 100

|project AccountUpn;

CloudAppEvents

| where Timestamp > ago(1d)

| where Application == "Microsoft Teams"

| where ActionType == "ChatCreated"

| where isempty(AccountObjectId)

```

| where RawEventData.ParticipantInfo.HasForeignTenantUsers == true
| where RawEventData.CommunicationType == "OneOnOne"
| where RawEventData.ParticipantInfo.HasGuestUsers == false
| where RawEventData.ParticipantInfo.HasOtherGuestUsers == false
| where AccountId has "@"
| extend TargetUPN = tolower(tostring(RawEventData.Members[1].UPN))
| where TargetUPN in (interestingUsers )
| extend TargetTenant = tostring(RawEventData.OrganizationId)
| extend AttackerTenant = RawEventData.Members[0].OrganizationId
|where TargetTenant != AttackerTenant
| extend AttackerUPN = RawEventData.Members[0].UPN
| extend AttackerName = RawEventData.Members[0].DisplayName
|project project-reorder Timestamp, AttackerTenant, AttackerUPN, AttackerName,
TargetUPN

```

Execution

Using a scripted cURL command, the attacker downloads additional payloads in an RMM session, or shares a SharePoint link on Microsoft Teams with payloads and tools (like NetSupport RAT). In a few scenarios, an SSH connection was also setup with the attacker's machine.

In conjunction with RMM tools, attackers use various command-line utilities to manipulate Active Directory (AD) environments. One such utility is Csvde, a command-line tool that imports and exports data from Active Directory Domain Services (AD DS). Csvde can be exploited by threat actors to extract sensitive AD information or to introduce malicious entries into the directory, further compromising the security of the environment.

Detect suspicious file downloads

DeviceNetworkEvents

```

| where InitiatingProcessFileName in~ ("curl.exe", "powershell.exe", "certutil.exe",
"bitsadmin.exe")

```

```

| where RemoteIPType == "Public"

| where RemoteUrl endswith ".exe" or RemoteUrl endswith ".dll" or RemoteUrl endswith ".zip"

| project Timestamp, DeviceId, InitiatingProcessFileName, RemoteIP, RemoteUrl

| extend AlertType = "Suspicious file download from unknown IP address"

```

Detection of csvde.exe Download and AD Enumeration

```

// Detect curl downloading csvde.exe

```

```

let csvde_download = DeviceProcessEvents

    | where InitiatingProcessFileName =~ "cmd.exe"

    | where ProcessCommandLine has_all ("curl","-o","csvde.exe","http:")

    | project DeviceId, Timestamp, CurlCommandLine=ProcessCommandLine, CurlProcessId
= ProcessId;

```

```

// Detect execution of csvde.exe with specific parameters

```

```

let csvde_execution =

    DeviceProcessEvents

    | where FileName =~ "csvde.exe"

    | where ProcessCommandLine has_all ("-r","objectClass=Computer")

        and ProcessCommandLine has_all ("-
I","samAccountName","description","info","operatingSystem")

        and ProcessCommandLine contains "-f"

    | project DeviceId, Timestamp, CsvdeCommandLine=
ProcessCommandLine, CsvdeProcessId = ProcessId;

```

Join the two events and look for them occurring within 5 minutes

```

csvde_download

```

```

| join kind=inner (

    csvde_execution

```

```

) on DeviceId

```

```

| where Timestamp between (Timestamp1 .. Timestamp1 + 5m)

| extend AlertType = "Potential Active Directory Enumeration", Details = strcat("curl.exe
was used to download csvde.exe, which was then executed to enumerate AD computers. ")

// Detect potential data compression and exfiltration

let compress_exfil = DeviceProcessEvents

| where FileName =~ "7z.exe"

| where ProcessCommandLine contains "x -p"

| project Timestamp, DeviceId, FileName, ProcessCommandLine, ProcessId

| join kind=inner (

    DeviceNetworkEvents

    | where InitiatingProcessId != 0

) on $left.ProcessId == $right.InitiatingProcessId

| project Timestamp, DeviceId, FileName, ProcessCommandLine, RemoteIP, RemotePort

| extend AlertType = "Potential data compression and exfiltration";

```

Reconnaissance

This attack specifically involved high number of reconnaissance commands including ipconfig, systeminfo, geo location scans, user recons, EDR protection status. In a typical attack, this information is exfiltrated to an external C2 server. However, in these attacks, the attacker could have probably taken a screenshot through the RMM.

Persistence

The payloads downloaded by the attacker was used to create persistence either using scheduled tasks or by being added to the startup folder.

```

// Persistence through Startup operations
let regKeys = pack_array(@"CurrentVersion\Run",
@"CurrentVersion\RunOnce",
@"CurrentVersion\RunOnceEx",
@"Programs\Startup",

```

```

@"CurrentVersion\RunServicesOnce",
@"CurrentVersion\RunServices",
@"CurrentVersion\Policies\Explorer\Run",
@"Windows NT\CurrentVersion\Windows",
@"System\CurrentControlSet\Control\Session Manager");
let startUpOperations = DeviceFileEvents
| where FolderPath has @"Start Menu\Programs\Startup"
| where ActionType in ("FileCreated", "FileModified", "FileRenamed");
// Persistence through Registry Tampering
let regOperations = DeviceRegistryEvents
| where hasAlertDevices
| where Timestamp between (startTime .. endTime)
| where DeviceId in (alertedDevices)
| where RegistryKey has_any (regKeys)
| where ActionType in ("SetValue", "CreateKey", "RenameKey");
// Persistence through Scheduled task creation
let scheduledOperations = DeviceProcessEvents
| where Timestamp between (startTime .. endTime)
| where DeviceId in (alertedDevices)
| where (InitiatingProcessCommandLine has "schtasks" and InitiatingProcessCommandLine
has_any ("run", "create", "change"));
union startUpOperations, regOperations, scheduledOperations
| summarize arg_min(Timestamp, *) by DeviceId

```

Detections

- Suspicious activity using Quick Assist
- Possible remote access tool activity
- Suspicious usage of remote management software
- Suspicious location of remote management software
- Possible NetSupport Manager activity

References

NOTE: The following references are available for Microsoft Defender customers.

- Qakbot distributor Storm-0464 shifts to DarkGate and IcedID : [Intel Article - Microsoft Defender - Shift to DarkGate and IcedID](#)
- Financially motivated threat actors misusing App Installer : [Intel Article - Microsoft Defender - App Installer misuse](#)
- Threat actors misusing Quick Assist in social engineering attacks leading to ransomware : [Intel Article - Microsoft Defender - Quick Assist misuse](#)

Recommendations

- Educate Microsoft Teams users to verify 'External' tagging on communication attempts from external entities, be cautious about what they share, and never share their account information or authorize sign-in requests over chat.
- Administrators have an option to manage [chats/Teams meetings with external users not managed by the Organization](#)
- Apply Microsoft's [security best practices for Microsoft Teams](#) to safeguard Teams users.
- Educate users about diligent use of RMM tools
- Implement [Conditional Access authentication strength](#) to require phishing-resistant authentication for employees and external users for critical apps.
- Enable [investigation and remediation](#) in full automated mode to allow Defender for Endpoint to take immediate action on alerts to resolve breaches, significantly reducing alert volume.

NOTE: The following is available for Microsoft Defender customers.

Refer to [Microsoft's human-operated ransomware overview](#) for general hardening recommendations against ransomware attacks.