bluemoontuesday - Post-Incident Report

Nicholas H. Coleman - Security Analyst

Executive Summary

Incident ID: `INC-2025-01-22`

Incident Severity: High

Incident Status: Resolved

Incident Overview: On January 22nd 2025, an incident occurred at the organization: *bluemoontuesday*. An employee, while searching for Google Authenticator online, downloaded a suspicious file. Our SOC team was notified that this incident may have been similar to these previous cases: Incident-1, Incident-2. Once notified of the incident, our team was provided a pcap file containing the associated network traffic.

The team was then able to locate the infected Windows host along with the name of the user who was involved in the download of the initial malicious file. The domain name for the malicious site as well as the C2 Servers associated were also found. We were then able to contain the threat by denylisting the C2 Servers and all domains associated.

Key Findings

Initial Infection Vector

The incident was initiated when a user at *bluemoontuesday* attempted to download Google Authenticator but instead retrieved a suspicious file named `application_setup.js` from a malicious domain.

- **File Name:** application_setup.js
- **SHA256 Hash:** 4bed34b1cd5663a5a857b3bbf81cc5413c61cb561e9a90067b57da08b01ae70b
- **File Type:** ASCII text
- **Malicious Payload:** Embedded call to download further content via `GetObject("scriptlet:...")`

Confirmed Malware Activity

A PowerShell script identified as `pas.ps1` was confirmed as a malicious payload used during the infection. This matched patterns seen in previous incidents (Incident-1, Incident-2).

- **Indicator of Compromise (IoC):** `pas.ps1`
- **Type:** PowerShell script
- **Hash:** a833f27c2bb4cad31344e70386c44b5c221f031d7cd2f2a6b8601919e790161e

Infected Host Identified

Analysis of the provided PCAP file confirmed that the compromised Windows host was:

- **Internal IP:** 10.1.17.215
- **Observed Communication** with known malicious infrastructure

Malicious Infrastructure Identified

Multiple domains and IP addresses were involved in hosting and serving the malicious content, including:

- **C2 IP:** `5.252.153[.]241:80` – served payload via scriptlet
- **Malicious Domains:**
  - `microsoft-teams-download.burleson-appliance[.]net`
  - `microsoft.teams-live[.]com`
  - Originated from a malicious Bing ad click (`bing[.]com/aclk?...`)
- **Associated IP:** 82.221.136.26 (HTTPS traffic on port 443)

Additional Malicious Artifacts Discovered

Forensic analysis revealed several suspicious files on the affected host, consistent with abuse of remote access tools and script-based payloads:

Filename	Type	Suspicion	Hash
pas.ps1	PowerShell	Likely primary payload script	a833f27c...
skqllz.ps1	PowerShell	Possibly secondary script	fd045fce...
TeamViewer.exe	EXE	Possibly modified or packed RAT	904280f2...
Teamviewer_Resource_fr.dll	DLL	Abused DLL	9634ecaf...
TV.dll	DLL	Possibly injected	3448da03..

Threat Containment

- The infected system was isolated.
- All identified C2 servers and malicious domains were denylisted.
- No further outbound connections to known malicious infrastructure have been observed post-containment.

Root Cause Analysis

User Behavior/Social Engineering

The root cause of the incident was a user-initiated download of a malicious JavaScript file (application_setup.js) after interacting with a **malicious advertisement** masquerading as a Google Authenticator download.

The domain: *google-authenticator.burleson-appliance[.]net* was designed to convince targets of its legitimacy, increasing the likelihood of user interaction/download.

Malvertising

A malvertising campaign on Bing - an increasing issue in today's cyber-security landscape, is to blame for the initial infection. Clicking on this particular malvertisement initiated the download of a JavaScript dropper which then invoked the Windows Script Host to execute code from one of the threat actor's C2 remote servers - `5.252.153[.]241`.

Execution of Remote Payload Script

The JavaScript file then used a call to download and execute the `pas.ps1` malicious script from this remote server.

This method bypasses traditional means of downloading, leveraging trusted Windows system components to directly execute code. This severely limits the ability of anti-virus scanners and endpoint defense measures to detect any malicious downloads.

Lack of Application Allowlisting

The initial file was a tiny 72-byte JavaScript file without even a line terminator. A file of this size can potentially bypass antivirus/content scanners.

Bluemoontuesday's online work environment allowed for the execution of scripts by users without necessary and sufficient security restrictions. This oversight is what ultimately expedited the rapid compromise of the system.

Additionally, the malicious domains involved in this incident were not denylisted by bluemoontuesday's web filtering systems. Furthermore, the redirection chain through Bing was

not blocked by DNS filtering or browser security plugins. An insecure browser environment is partly to blame.

Insufficient Detection

Detection only occurred after network traffic analysis via PCAP, indicating that initial infection and payload delivery went unnoticed by endpoint security or behavioral monitoring systems.

Recommendations

To prevent similar incidents in the future, the following actions are recommended for immediate implementation:

DNS Filtering and Web Content Controls

- Deploy a ***DNS filtering solution*** to block access to known malicious domains.
- Enable category-based restriction web filtering to restrict access to potentially threatening domains such as file-sharing and uncategorized websites.

Strengthen Endpoint Protection

- Ensure all endpoints are equipped with ***modern EDR (Endpoint Detection and Response)*** tools capable of detecting malicious scripts, anomalous behavior, and unauthorized network activity.
- Configure endpoint policies to block or alert on script execution from non-standard paths (e.g., PowerShell, WScript, and cscript used in user folders).

Enhance Browser and Ad Security

- Deploy ad blockers and safe browsing extensions on all corporate browsers to reduce risk from malvertising.
- Consider disabling or restricting ***JavaScript execution*** from untrusted domains or unfamiliar sources.

User Awareness Training

- Conduct **targeted phishing and malvertising awareness training** focused on identifying deceptive ads and verifying download sources.
- Emphasize **safe software download practices**, including verifying URLs and preferring official app stores or trusted repositories.

PowerShell Logging and Restriction

- Enable **PowerShell transcription and module logging** via Group Policy or endpoint configuration to gain insight into script usage.
- Apply **PowerShell Constrained Language Mode** for non-administrative users to prevent advanced exploitation.

Network Monitoring and Segmentation

- Improve **network traffic monitoring** with deeper packet inspection and alerts for unusual outbound connections (e.g., IP-based C2 over port 443).
- Consider **segmentation of high-risk endpoints**, such as those used for administrative access or by users with broad internet access needs.

Regular Threat Intelligence Integration

- Keep denylists of malicious domains, IPs, and hashes **regularly updated** through integration with threat intel feeds.
- Cross-reference new alerts or incidents with prior known IoCs (e.g., `pas.ps1`, `TeamViewer.exe` variants).

Email and Browser Isolation (Optional)

- Evaluate remote browser isolation for high-risk users or departments that frequently browse external sites.
- Consider isolating email links or attachments using sandboxing or virtualization.