PROMPTFLUX is an experimental VBScript dropper GTIG (Google Threat Intelligence Group) discovered in mid-2025 that queries a large language model (Google Gemini) during execution to generate or obfuscate its own VBScript "just-in-time." The family is currently assessed by Google as **developmental/testing** (not widely destructive yet) but represents a clear move toward adaptive, self-modifying AI-assisted malware that can complicate signature-based detection.

1) Executive findings (top takeaways)
    - **AI-in-the-loop obfuscation:** PROMPTFLUX contains a module (nicknamed **"Thinking Robot"** or variants like "Thinking") that sends prompts to the Gemini API to get obfuscated VBScript code or new variants to replace/save over itself. This enables *just-in-time* metamorphism and makes static signatures fragile.
    - **Language & delivery:** The dropper is implemented in **VBScript**, contains a decoy installer to mask activity, and persists by saving regenerated scripts to the **Startup** folder. It also attempts to propagate via removable drives and mapped network shares.
    - **Operational status:** GTIG labels PROMPTFLUX as **experimental / development** — some self-update routines are commented out and Google has taken actions to disable associated assets — but it demonstrates novel TTPs to watch for.
    - **Broader context:** GTIG reported PROMPTFLUX as part of a larger pattern of adversaries using LLMs mid-execution (also observed: PROMPTSTEAL, PROMPTLOCK, QUIETVAULT, etc.), marking a new operational phase where models are integrated into malware toolchains. Several independent outlets covered GTIG's findings.
    - **Mitigation impact:** Because PROMPTFLUX relies on external model API access and observable behaviours (network POSTs to model endpoints, write activity to Startup and %TEMP%), defenders can build high-value mitigations now — e.g., restricting model API tokens, network monitoring, EDR behavioural detection and model-usage telemetry.

2) Technical analysis (what PROMPTFLUX does)
    a. Primary components
        - *Dropper*: VBScript that decodes and executes an embedded decoy installer to distract analysts/users.

        - *Thinking Robot / Thinking module*: a routine that crafts and issues POST requests to Gemini (hard-coded API key in samples Google analysed) requesting VBScript code that performs anti-AV/obfuscation. The model name observed: **gemini-1.5-flash-latest** (actor uses -latest to avoid model deprecation). Responses are logged and — in variants — used to overwrite/regenerate the script.

    b. Persistence & spreading
        - Saves regenerated/obfuscated script into the Windows **Startup** folder to persist across reboots.
        - Attempts to copy itself to removable media and mapped network shares to achieve lateral movement and propagation.

c. Evidence of R&D status
    - GTIG found *commented out* update functions (e.g., Attempt To Update Self) and safeguards that limit API calls — signs the authors are testing capabilities rather than running a mature campaign. Google disabled assets tied to the activity.

3) Indicators of Compromise (IOCs) & artifact list

Primary artifacts and behaviour traces reported by GTIG (use these for hunts and detection): Google Services

a. Hard-coded model & endpoint behaviour
    - Model referenced: gemini-1.5-flash-latest.
    - POSTs to Gemini API endpoints containing prompts that explicitly request VBScript obfuscation and "output only the code".
b. File and path indicators
    - Logging of AI responses to: %TEMP%\thinking_robot_log.txt.
    - Writes to the **Startup** folder (new VBScript files saved there).
    - Example social-engineering filenames observed in samples: e.g., "crypted_ScreenRec_webinstall" (GTIG notes varied lures).
c. Behavioural IOCs
    - Processes or scripts making outbound HTTPS POSTs that include code-generation prompts (especially to Gemini/Hugging Face endpoints) from workstation hosts.
    - Frequent hourly rewrite or new file creation patterns of script files in user profile / Startup directories.
    - Mass copy activity to removable media or writes to network shares triggered by script hosts.

4) Detection guidance (practical hunts and signatures)
These are defensive detection ideas; adjust to your environment to reduce false positives.

Network telemetry

- Alert on **HTTPS POSTs** from endpoints to known LLM API domains where the POST body includes VBScript tokens (e.g., WScript, VBScript, Set fso, CreateObject("Scripting.FileSystemObject")) or the prompt text asks for “VBScript” or “obfuscate VBScript” — flagged for analyst review. (Correlate with hosts that then create/modify scripts.)
- Monitor for outbound calls to Gemini model endpoints — block / require explicit allowlisting of service accounts for any model usage.

Endpoint / EDR

Create behavioural rules to flag:

- Creation / modification of files inside %APPDATA%\Microsoft\Windows\Start Menu\Programs\Startup or other startup persistence directories by wscript.exe/cscript.exe.
- Any scripts writing thinking_robot_log.txt in %TEMP% or unusual logging of LLM responses.

- Scripts that enumerate and copy files to removable drives or mapped shares immediately after execution.

5) Mitigation & containment (recommended controls)
    a. Restrict LLM API access & rotate keys

        - Immediately audit and restrict which service accounts and hosts can call LLM APIs (Gemini, Hugging Face, Anthropic, etc.). Remove hard-coded/shared keys and require short-lived tokens where possible. GTIG noted hard-coded keys in observed samples.

    b. Network allowlisting / egress filtering

        - Block or proxy egress to model endpoints from general user workstations. Require explicit approval for any model API traffic from servers/workstations. Monitor and alert on POSTs containing code generation prompts.

    c. Harden script execution policies

        - Enforce AppLocker/Windows Defender Application Control to restrict execution of unsigned scripts (VBScript/PowerShell) from user directories and removable media. Prevent wscript.exe/cscript.exe from creating Startup entries unless explicitly allowed.

    d. EDR behavioral detection

        - Create rules for suspicious writes to Startup, %TEMP%\thinking_robot_log.txt, mass copy to removable drives, and irregular hourly file rewrites. Investigate hosts that show network requests to LLM endpoints followed by script modifications.

    e. User awareness & phishing controls

        - PROMPTFLUX samples use social engineering lures. Train users not to run downloaded installers or copy/paste commands from untrusted sources. Disable autorun for removable media and monitor for files with social-engineering filenames.

    f. Threat intel & patching

        - Subscribe to GTIG/major vendor feeds for updated IOCs. Keep AV/EDR signatures updated and apply principle of least privilege to network shares and removable media.

6) Risk assessment & outlook

        - **Near term:** PROMPTFLUX is experimental — limited operational impact reported — but demonstrates a proof point: LLMs can be

integrated mid-execution to generate obfuscated payloads or commands. GTIG disrupted assets tied to the samples, but others could replicate the technique.

- **Medium/long term:** As LLM access becomes cheaper and more widespread (and models improve), expect greater automation in malware TTPs: automated obfuscation, evasive payload generation, dynamic command generation for data exfiltration, and quicker adaptation to defender countermeasures. Defenders must shift more to behavioral and network-centric detection that does not solely depend on static signatures. Multiple news outlets and industry press have emphasized this shift following GTIG's disclosure.

7) References & further reading (primary sources)

- Google Threat Intelligence Group — *Advances in Threat Actor Usage of AI Tools* (GTIG technical whitepaper, Nov 2025). Primary technical source for PROMPTFLUX details. Google Services
- The Hacker News — "Google Uncovers PROMPTFLUX Malware That Uses Gemini…" (coverage summarizing GTIG findings). The Hacker News
- CSO Online — "Google researchers detect first operational use of LLMs in active malware campaigns" (analysis of GTIG report). CSO Online
- Cybersecurity Dive / Tom's Guide / other reporting summaries of GTIG findings. Cybersecurity Dive+1

PromptFlux Malware – Detailed Attack Flow

1. Initial Access

Tactic: *Execution / User Interaction*

- Attacker delivers a malicious VBScript dropper through:
  - Fake installers
  - Social-engineering lures
  - Email attachments / download links
- User is tricked into running the "installer" (the **decoy application** bundled inside the malware).

2. Execution of the Dropper

Tactic: *Execution (T1059 – Scripting)*

- VBScript begins running via wscript.exe or cscript.exe.
- The dropper:
  - Executes the decoy installer so the user thinks something legitimate is happening.
  - Unpacks the main malware logic in memory.

3. **AI-Assisted Code Generation (Thinking Robot Module)**

   Tactic: ***Defense Evasion (T1027 – Obfuscation)***

   This is the core innovation of PromptFlux.

   - The malware triggers its "Thinking Robot / Thinking" module.
   - It creates an HTTP POST request to Google Gemini API with a hard-coded key.
   - The malware sends prompts such as:
     - "Obfuscate this VBScript,"
     - "Generate new VBScript code,"
     - "Output VBScript only."

   - The Gemini API responds with:
     - Freshly generated obfuscated code **or**
     - A modified version of the existing VBScript.

4. **Self-Regeneration / Just-In-Time Mutation**

   Tactic: ***Defense Evasion (T1140 – Deobfuscation / decoding)***

   - The AI-generated VBScript payload is:
     - Logged into %TEMP%\thinking_robot_log.txt
     - Used to overwrite the original malware script
     - Saved as a new variant into user directories

- This allows hourly or periodic mutation, making signature detection extremely hard.

5. **Persistence Mechanism**

   Tactic: ***Persistence (T1547 – Startup Folder)***

   - PromptFlux creates a startup entry by saving the regenerated script into:

     Code:
     %APPDATA%\Microsoft\Windows\Start Menu\Programs\Startup\

   - Ensures execution on every reboot.

6. **Lateral Propagation**

   Tactic: ***Lateral Movement (T1091 – Removable Media)***

   - The malware attempts to copy itself to:
     - Removable USB drives
     - Network shares
   - Uses social-engineering filenames to trick other users into running it.

7. **Additional Module Execution**

   - Some variants include commented-out routines for update/self-repair.
   - Indicates the malware is experimental and evolving.

8. **Stealth & Evasion**

   Tactic: ***Defense Evasion***

   Prompt Flux avoids detection using:

   - Constant re-obfuscation via Gemini API

- Continual regeneration of script structure
- File rewriting in startup folder
- Minimal static signatures
- Legitimate user activity (decoy installer) to hide malicious routines

9. **Command & Control (AI-based)**

   Tactic: *Command & Control*

   - Instead of a traditional C2 server, PromptFlux relies on:
     - LLM API as a pseudo-C2 channel
     - AI responses act as:
       - Obfuscation engine
       - Code packer
       - Update generator
   - This makes the malware:
     - Harder to block
     - Harder to analyze
     - Highly adaptive

Complete PromptFlux Attack Flow Diagram