# **Detecting Advanced Phishing Campaigns by LAPSUS$, SocGholish, DPRK, Scattered Spider & Lazarus**

Phishing campaigns by groups like **LAPSUS$**, **SocGholish (FakeUpdates)**, North Korea’s **Lazarus Group** (including DPRK “remote IT worker” personas), and **Scattered Spider** often share common techniques. These include malicious Office documents with macros, JavaScript downloaders, PDF/ISO container payloads, and use of system binaries (LOLBins) for stealth.

Many of these campaigns are financially motivated or target financial institutions, using tailored lures and infrastructure. The following YARA Live Hunt rule is designed to catch such threats with a balanced approach – broad enough to cover multiple TTPs, yet with conditions to reduce false positives.

## **Rule Focus and Strategy**

* **Targeted File Types:** Office documents (.docx/.docm, .xlsm, .pptm), PDFs, ISO images, script files (.vbs, .js/.jse), Windows shortcuts (.lnk), and executables. These are commonly used to deliver malware in the mentioned campaigns

For example, Lazarus has used Word/Excel job offer documents with macros , SocGholish delivers malicious JavaScript often in ZIP or HTML form and many threat actors now send ISO attachments (with embedded LNK/EXE) to bypass security controls.

* **Macro and Script Heuristics:** The rule looks for signs of **VBA macros** in Office files (e.g., auto-execution keywords and suspicious API usage) and **malicious scripts** in JS/VBS/LNK. This includes keywords like AutoOpen, Document\_Open, or Workbook\_Open that trigger on document open, as well as use of CreateObject, Shell.Application, or WScript.Shell to launch other processes. Such patterns often indicate a macro attempting to execute code . Heuristic strings also catch obfuscation techniques (e.g., multiple Chr( or fromCharCode uses to build strings, eval( or Execute calls, and presence of embedded Base64 or shellcode). SocGholish scripts, for instance, heavily obfuscate their PowerShell loaders , and Lazarus macros have been observed decoding payloads from string arrays at runtime .

The rule flags these behaviors.

* **PDF/ISO Indicators:** In PDFs, the rule hunts for embedded script indicators like /JavaScript or /OpenAction, and in ISO files it attempts to detect embedded file names or code (e.g., .exe or .lnk strings inside the binary). Attackers frequently embed malware in PDFs (as links or attachments) and use ISO containers to carry LNK or DLL files that execute malware when mounted. Including these patterns helps catch phishing lures carrying PDF exploits or ISO-mounted malware.
* **Known C2 Domains/IoCs:** We incorporate known or suspected infrastructure keywords linked to these groups, especially those targeting financial organizations. For example, Lazarus (DPRK) campaigns have used domains like markettrendingcenter[.]com (seen in a Lazarus job lure macro downloading a decoy doc ) and honehsn[.]com . SocGholish campaigns have hosted payloads on attacker domains such as rzegzwre[.]top . Scattered Spider is known to register victim-themed domains (e.g. containing -sso for single sign-on portals) for phishing . The YARA rule includes fragments of these IoCs (domains and keywords) to directly match known bad infrastructure.
* **LOLBins and Living-off-the-Land:** The rule searches for strings related to **living-off-the-land binaries** – e.g. powershell, mshta, rundll32, regsvr32, certutil, bitsadmin, wuauclt, etc. – which may appear in macro code, scripts, or LNK command targets. Threat actors across these groups abuse such binaries for executing payloads or persistence. For instance, Lazarus’s “LolZarus” campaign used mshta.exe and even the Windows Update client (wuauclt.exe) to execute malicious code . SocGholish JS often invokes PowerShell via Wscript.Shell . By detecting these substrings, the rule can catch malware leveraging OS tools for evasion.
* **Financial Targeting Keywords (Optional):** While not strictly required in detection logic (to avoid false positives), the rule gives weight to content suggesting financial lures. Terms like “invoice”, “payment”, or specific bank names could appear in phishing decoys targeting finance staff. These aren’t mandatory triggers, but if found alongside malicious code, they reinforce the match. (Analysts can easily modify the rule to include particular keywords relevant to their organization’s threat profile.)

## **YARA Live Hunt Rule**

Below is the comprehensive YARA rule, **with comments explaining each section**. This rule can be used in VirusTotal LiveHunt to continuously flag files exhibiting the described traits:

--------------------------------------------------------

rule Advanced\_Phishing\_LiveHunt {  
 meta:  
 author = "YourName (Threat Research)"  
 description = "Detects phishing malware tied to LAPSUS$, SocGholish, Lazarus/DPRK, Scattered Spider, etc. \  
Use of Office macros, malicious JS, PDF/ISO lures, known C2 domains, and LOLBins."  
 date = "2025-02-28"  
 threat\_groups = "Lazarus (APT38), LAPSUS$, SocGholish (FakeUpdates), Scattered Spider, DPRK IT workers"  
 reference = "Based on known TTPs of listed groups&#8203;:contentReference[oaicite:18]{index=18}&#8203;:contentReference[oaicite:19]{index=19}&#8203;:contentReference[oaicite:20]{index=20}"  
 tlp = "WHITE"  
   
 strings:  
 // 1. Office Macro related strings (VBA auto-exec and typical malicious calls)  
 $vba\_autoopen = /Auto(Open|\_Open)/i // e.g., AutoOpen in Word, Auto\_Open in Excel  
 $vba\_doc\_open = /Document\_Open/i // Word Document open event  
 $vba\_wb\_open = /Workbook\_Open/i // Excel Workbook open event  
 $vba\_ppt\_open = /Presentation\_Open/i // PowerPoint open event (auto macro)  
 $vba\_createobj = /CreateObject/i // CreateObject call (to create COM objects)  
 $vba\_shell\_app = /Shell\.Application/i // Shell.Application COM (often to launch executables)  
 $vba\_wscript = /WScript\.Shell/i // WScript.Shell usage (e.g., to run commands)  
 $vba\_urlmon = /URLDownloadToFileA?/i // URLDownloadToFile API (downloading payload)  
 $vba\_xmlhttp = /Msxml2\.XMLHTTP/i // XMLHTTP (another way to download)  
 $vba\_powershell = /PowerShell/i // Possibly invoking PowerShell from macro  
 $vba\_reg\_write = /CreateTextFile|WriteText/i // Writing to files (dropping payloads)  
 $vba\_chr\_concat = /Chr\(/i // Use of Chr() for obfuscation (common in macro malware)  
   
 // 2. Malicious Script/HTA/JS/VBS related strings (obfuscation and ActiveX use)  
 $js\_eval = /eval\(/i // Use of eval() in script  
 $js\_unescape = /unescape\(/i // unescape (often used for string decode)  
 $js\_fromChar = /fromCharCode/i // building strings from char codes  
 $js\_activeX = /ActiveXObject/i // ActiveXObject (often WScript.CreateObject in JS)  
 $js\_execute = /Execute(Global)?\s\*\(/i // VBS execute string or ExecuteGlobal   
 $js\_enc\_ps = /-enc\s+[A-Za-z0-9+/]{50}/i // Encoded PowerShell command (base64 chunk)  
 $js\_c2\_url = /https?:\/\/[A-Za-z0-9.\_\-]+\//i // Any HTTP/HTTPS URL pattern  
 $js\_long\_str = /[A-Za-z0-9+\/]{100,}/ // Very long base64/obfuscated string blob  
   
 // 3. PDF/ISO specific indicators  
 $pdf\_js = /\/JavaScript|\/JS/i // PDF JavaScript tag  
 $pdf\_openaction = /\/OpenAction/i // PDF auto action  
 $pdf\_embed\_file = /\/EmbeddedFile/i // PDF with embedded file   
 $iso\_lnk\_name = /\\.lnk(;1)?/i // “.lnk” filename inside ISO (ISO9660 format)  
 $iso\_exe\_name = /\\.exe(;1)?/i // “.exe” inside ISO  
 $iso\_powershell = /powershell.exe/i // Powershell present in ISO content (e.g., in LNK target)  
   
 // 4. Known malicious domains or keywords (C2 infrastructure)  
 $dom\_lazarus\_1 = "markettrendingcenter" // Lazarus job lure domain&#8203;:contentReference[oaicite:21]{index=21}  
 $dom\_lazarus\_2 = "lm-career" // Lazarus (Lockheed Martin career lure) domain&#8203;:contentReference[oaicite:22]{index=22}  
 $dom\_lazarus\_3 = "honehsn.com" // Lazarus APT IOC domain&#8203;:contentReference[oaicite:23]{index=23}  
 $dom\_socgholish\_1 = "rzegzwre.top" // SocGholish payload host&#8203;:contentReference[oaicite:24]{index=24}  
 $dom\_scatter\_1 = "-sso." // Scattered Spider phishing SSO domains&#8203;:contentReference[oaicite:25]{index=25}  
 $dom\_generic\_fin1 = "login.htm" // Generic credential phish page indicator  
 $dom\_generic\_fin2 = "account-update" // Common phrasing in phishing URLs  
   
 // 5. LOLBins and Living-off-the-land binaries usage  
 $lolbin\_powershell= "powershell" nocase // PowerShell execution  
 $lolbin\_mshta = "mshta" nocase // mshta (HTA application runner)  
 $lolbin\_rundll32 = "rundll32" nocase // rundll32 (DLL loader)  
 $lolbin\_regsvr32 = "regsvr32" nocase // regsvr32 (DLL registration, often abuse for remote SCF)  
 $lolbin\_wscript = "wscript.exe" nocase // Windows Script Host  
 $lolbin\_cscript = "cscript.exe" nocase // Console Script Host  
 $lolbin\_bitsadmin = "bitsadmin" nocase // BITSAdmin (file download)  
 $lolbin\_certutil = "certutil" nocase // CertUtil (often used to decode or download)  
 $lolbin\_wuauclt = "wuauclt" nocase // Windows Update client (used abusively by Lazarus&#8203;:contentReference[oaicite:26]{index=26})  
 $lolbin\_msiexec = "msiexec" nocase // Windows Installer  
 $lolbin\_installutil = "installutil" nocase // InstallUtil .NET utility  
 $lolbin\_schtasks = "schtasks" nocase // Creating Scheduled Tasks (persistence)  
   
 // 6. Lure content keywords (financial and corporate themes)  
 $lure\_invoice = /Invoice|Payment/i // Financial lure terms  
 $lure\_bank = /Bank\s+Statement|Wire Transfer/i // Banking terms  
 $lure\_job = /Resume|CV|Job\s?Offer/i // Job lure terms (DPRK/Lazarus themes)  
   
 condition:  
 // File type check: target specific extensions or file formats  
 uint16(0) == 0x4D5A or /\* PE executable (MZ header) \*/   
 uint32(0) == 0x504B0304 or /\* ZIP (Office Open XML, e.g., docx) \*/  
 uint32(0) == 0x25504446 or /\* PDF (%PDF) \*/  
 uint32(0) == 0x2142444E or /\* Windows .doc/.xls Binary OLE (D0 CF 11 E0 in little-endian) \*/  
 uint32(0) == 0xEDABEEDB or /\* Microsoft Installer or Compound File alternate magic \*/  
 uint32(0) == 0x00000100 or /\* ISO 9660 (0x00000100 "CD001" appears later in file) \*/   
 ext matches /(exe|dll|scr|doc|docm|docx|xls|xlsm|pptm|ppt|pdf|iso|js|jse|vbs|lnk)$/i  
 /\*   
 The above checks ensure the file is one of the targeted types:  
 - PE executables/DLLs (.exe, .dll, .scr)  
 - Office docs (OLE format or OpenXML)  
 - PDF files  
 - ISO images  
 - Script files (JS, JSE, VBS)  
 - Windows LNK shortcuts  
 \*/  
 and  
 (  
 // Condition branch 1: Malicious Office documents with macros  
 (ext matches /(doc|docm|docx|xls|xlsm|ppt|pptm)$/i and any of ($vba\_\*))   
 and ( // require some additional suspicious signs in the doc  
 any of ($lolbin\_\*) or any of ($dom\_\*) or any of ($js\_long\_str, $vba\_chr\_concat)   
 or $pdf\_js or $pdf\_openaction // (in case of doc embedding PDF objects or similar)  
 or any of ($lure\_\*) // lure keywords (not strictly required for match, but counted)  
 )  
 ) or   
 (  
 // Condition branch 2: Malicious scripts (JS, VBS, HTA, or LNK calling scripts)  
 (ext matches /(js|jse|vbs|lnk)$/i and (any of ($js\_\*) or any of ($lolbin\_\*)))  
 and (any of ($lolbin\_\*) or any of ($dom\_\*) or $js\_long\_str or $js\_enc\_ps)  
 ) or   
 (  
 // Condition branch 3: PDF files with script or attachments  
 (ext matches /pdf$/i and ($pdf\_js or $pdf\_openaction or $pdf\_embed\_file))  
 and (any of ($dom\_\*) or any of ($lolbin\_\*) or any of ($lure\_\*))  
 ) or  
 (  
 // Condition branch 4: ISO container with embedded suspicious content  
 (ext matches /iso$/i and ($iso\_lnk\_name or $iso\_exe\_name))  
 and (any of ($lolbin\_\*) or any of ($dom\_\*) or $iso\_powershell)  
 )  
 }

---------------------------------------------------------------------------------------------

**How it works:** The rule is structured to first ensure the file is of an expected type (by magic bytes or extension). Then it checks multiple *branches* to cover different scenarios, requiring both a primary indicator and a secondary corroborating indicator for a positive match:

* **Office Docs:** Must contain macro-related strings *and* at least one other suspicious element (like a LOLBin call, an embedded URL, obfuscated code, or lure keywords). This catches documents that not only have macros (which could rarely be benign) but are doing something fishy like launching PowerShell or reaching out to a known bad domain.
* **Scripts (JS/VBS/LNK):** Must contain script obfuscation or ActiveX usage and a sign of network/C2 or LOLBin usage (e.g. an obfuscated PowerShell command). This addresses JavaScript downloaders like SocGholish (which often include fromCharCode obfuscation **and** PowerShell invocation) as well as malicious LNK files that execute scripts.
* **PDFs:** Must contain PDF scripting/automation tags and also either known bad domains, LOLBin keywords, or lure terms. Legitimate PDFs rarely have embedded JavaScript or auto-actions, so this is a strong indicator when combined with, say, a known phishing domain or a bank-related term in the text.
* **ISOs:** Must contain indications of embedded executables (filenames) and also a second indicator like a PowerShell string or known bad domain. This helps catch ISO files carrying a malicious LNK/EXE that launches malware (for example, an ISO containing update.lnk that runs PowerShell downloader). The ISO’s raw content is scanned for telltale strings (ISO 9660 directory entries are in plaintext, e.g. EVILAPP.EXE;1) to identify malicious content inside.

By requiring multiple conditions (e.g. macro + URL, or script + LOLBin), the rule reduces false positives and homes in on the **techniques specifically associated with these threat actors**. It also prioritizes known IoCs from campaigns targeting financial institutions and big enterprises – for instance, Lazarus’s history of bank intrusions via spearphishing and Scattered Spider’s customized phishing domains for large companies are accounted for in the strings.

## **Usage and Tuning**

This YARA rule is intended for **Live Hunt** on VirusTotal, meaning it will continuously scan incoming files. The chosen strings and conditions cover a wide range of malicious behavior: if a file matches, it is very likely part of a phishing/malware campaign related to the mentioned TTPs. Nonetheless, analysts should review hits to verify intent, especially if only generic indicators (like common terms) caused the match.

The rule can be tuned further by:

* Adding or removing domains and hashes specific to your threat intel (under the $dom\_\* strings or as additional $hash\_\* strings if needed).
* Adjusting sensitivity of obfuscation checks (e.g. the length threshold for $js\_long\_str or the number of Chr( occurrences required).
* Incorporating new LOLBins or macro techniques as threat actors evolve (the meta section can track versions/updates).

With this rule deployed, defenders gain a powerful tool to **hunt** for phishing payloads from groups like LAPSUS$, SocGholish, Lazarus, Scattered Spider, etc., catching them by their known tricks – from malicious macros to sneaky scripts and abused system tools – while keeping false alarms low through layered conditions. **Deploy this rule in LiveHunt to get alerts on any files that exhibit these red flags, enabling rapid investigation and response.**

**Sources:**

1. Qualys Threat Research – *“LolZarus” Lazarus campaign uses job-themed documents with macros and LOLBins*
2. Red Canary – *SocGholish delivers malicious JavaScript masquerading as software updates*
3. [r](https://redcanary.com/threat-detection-report/threats/socgholish/#:~:text=SocGholish)CISA Advisory (AA23-320A) – *Scattered Spider uses phishing (emails/SMS) and customized domains for initial access*
4. SecurityAffairs – *SocGholish campaign uses heavily obfuscated PowerShell loaders and fileless RAT* Rewterz Threat Advisory – *Lazarus Group targeting financial orgs via spear-phishing (Active IOCs listed)*
5. Red Canary – *ISO attachments used to bypass security (MOTW) in phishing campaigns*