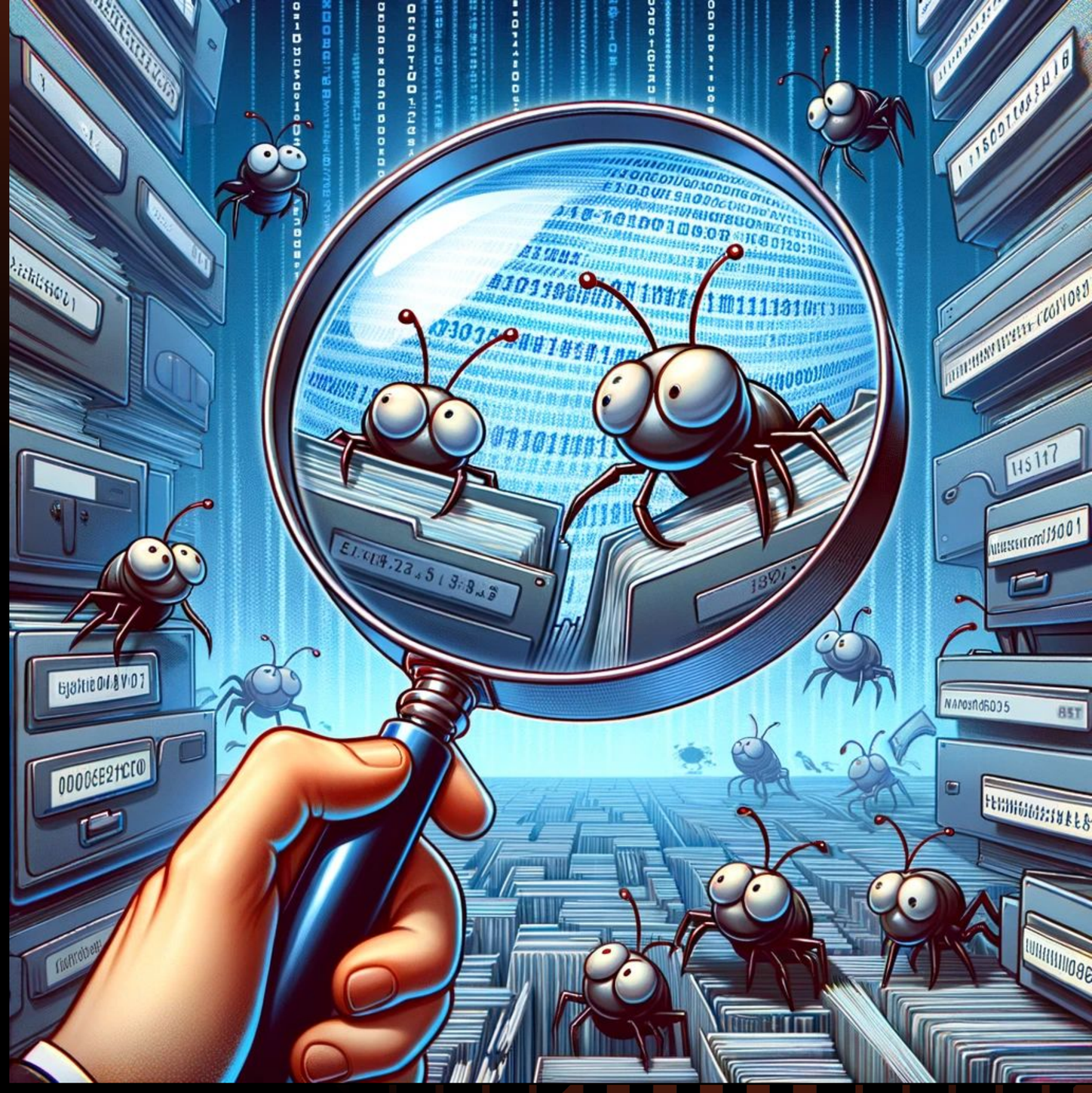
Detouring Danger: Hunting Privileged File Operation Vulnerabilities in OT/ICS software



Asher Davila - Principal Security Researcher

ICS Village

@asher_davila

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

$ Whoami

- Proudly Mexican 🇲🇽
- Worked:
  - **Pentester** @ Mission Critical Security - Projects with mid and large enterprise in Mexico and Spain.
  - **Security Analyst** @ Baxter int. - In-house SOC in Guadalajara, Mexico
  - **Freelancer Mobile Application Security Tester** - Mexico
  - **Security Researcher** @ Zingbox Inc. - IoT malware analysis and detection; IoT vulnerability hunting - Silicon Valley
  - **Security Researcher** @ Palo Alto Networks - IoT malware hunting; IoT vulnerability hunting - Silicon Valley
  - **Vulnerability Researcher** @ Microsoft, MORSE - Vulnerability hunter and exploiter - Seattle
  - **Vulnerability Researcher** @ Palo Alto Networks, CDSS Research - Mainly focused on ICS/SCADA, IoT - critical assets - Silicon Valley
- Hobbies: Music (if you find my name in music stream platforms it is possibly me), outdoor activities, retrocomputing

Opinions expressed are my own, not necessarily those of my employer.

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Your PC ran into a problem and needs to restart. We're just collecting some error info, and then we'll restart for you.

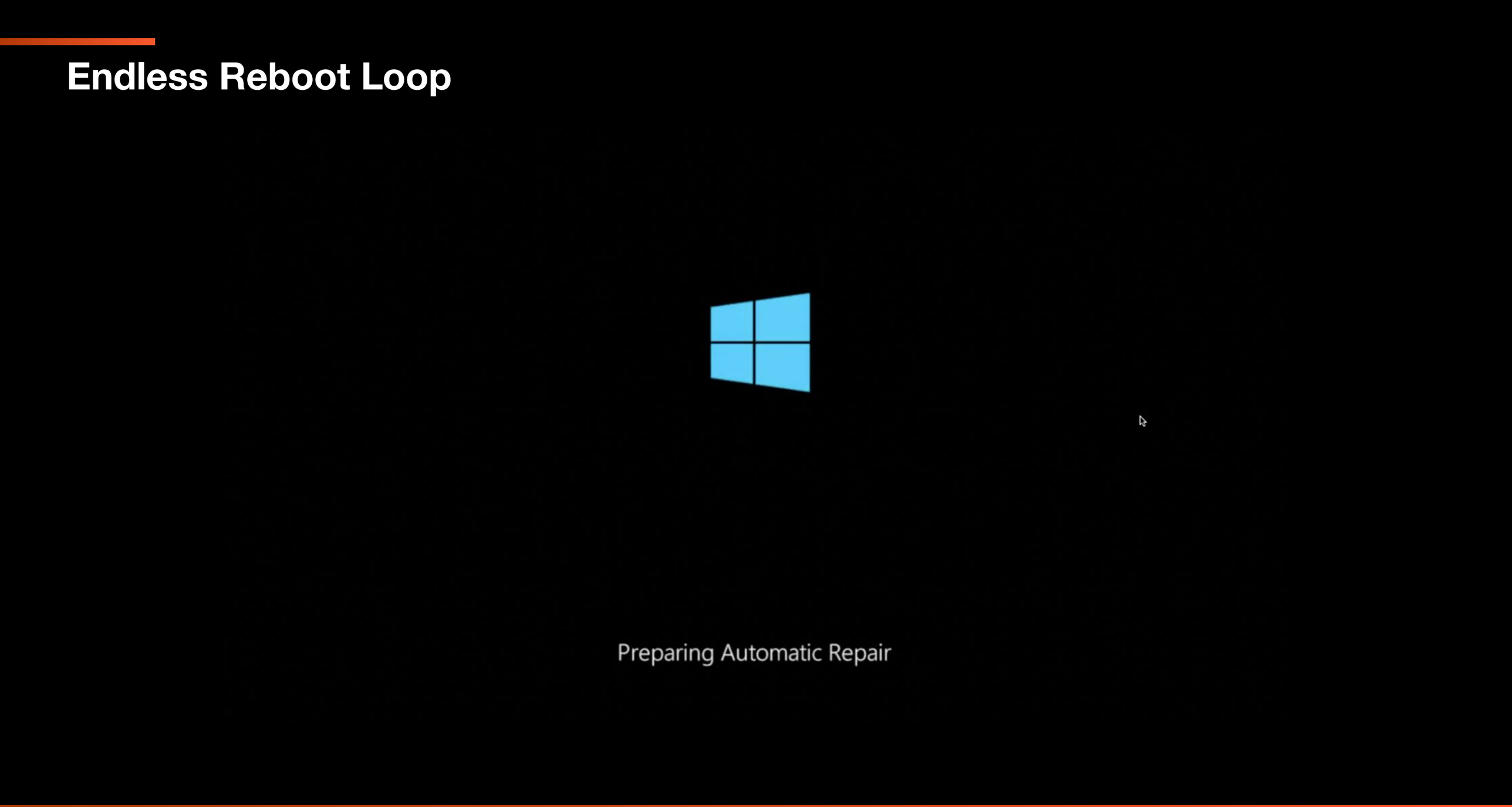
85% complete

For more information about this issue and possible fixes, visit our website.

If you call a support person, give them this info:

Stop code: 0x00000000

Endless Reboot Loop



© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Don’t know what is happening or where this error comes from?

- The error seems to be a corrupted driver.
- The error does not seem to be caused by a known ICS malware
- … Nor by the EDR

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

File System Operation Vulnerability

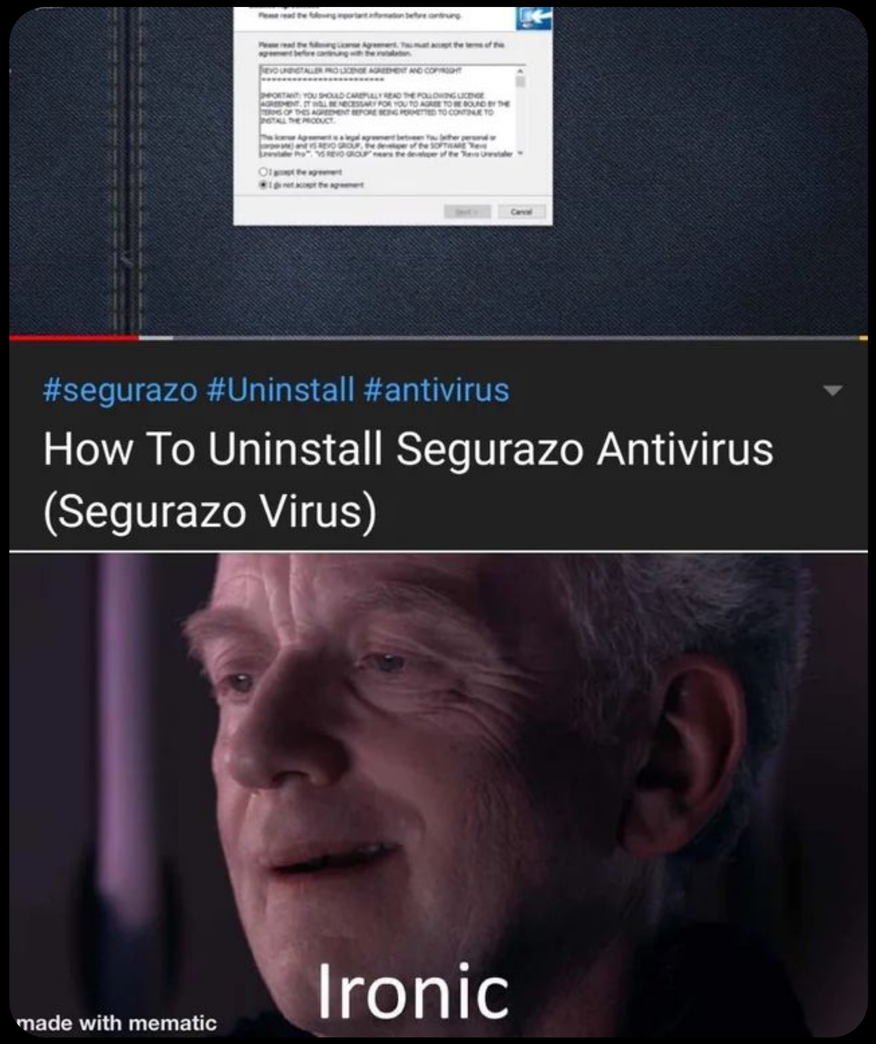
File System Operation vulnerability

- **Elevated privilege processes risk** security **when accessing unprivileged files** or folders without strict safeguards, potentially leading to operation abuse.
- **Well-known attacks** include **DLL planting/DLL hijacking** (and their variants). Despite being a known issue, many developers oversight this.
- **Abusing** other **file system operations** (like create, copy, move, or delete) is not as popular but equally or even more dangerous. Some of these operations can lead you to a DoS, information leakage, or even ✨**SYSTEM**✨
- **Logical vulnerabilities**, involving **stable filesystem manipulations without memory corruption**, often remain through code refactoring and are independent of processor architecture, making them highly valuable to attackers.

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Typical example of this issue.

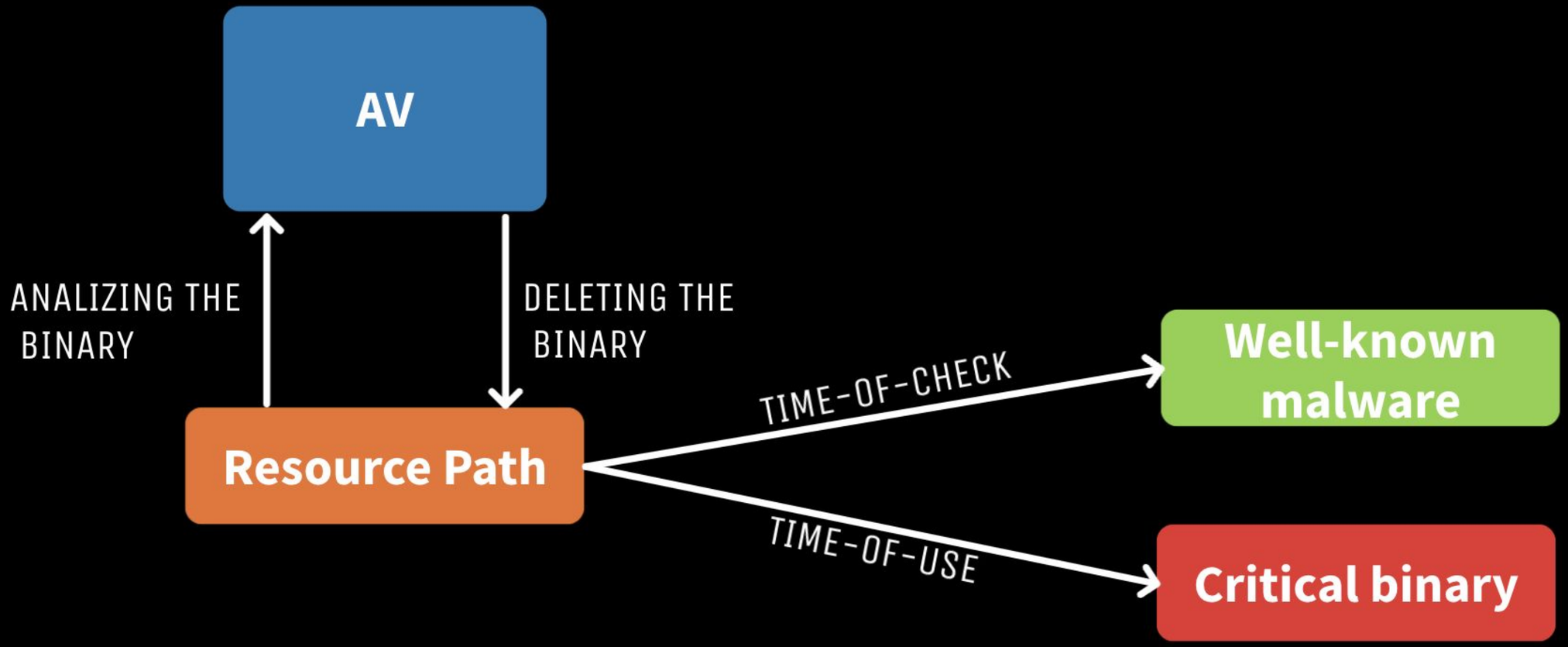
- The damn **AV**
- Installers/Uninstallers
- **Time-of-Check/Time-of-Use** - TOCTTOU
  - Opportunistic Lock
- **Write** / **Overwrite**
- **Move** / **Copy**
- **Delete**
- **Opening** -> Information disclosure



Internet meme, Author unknown

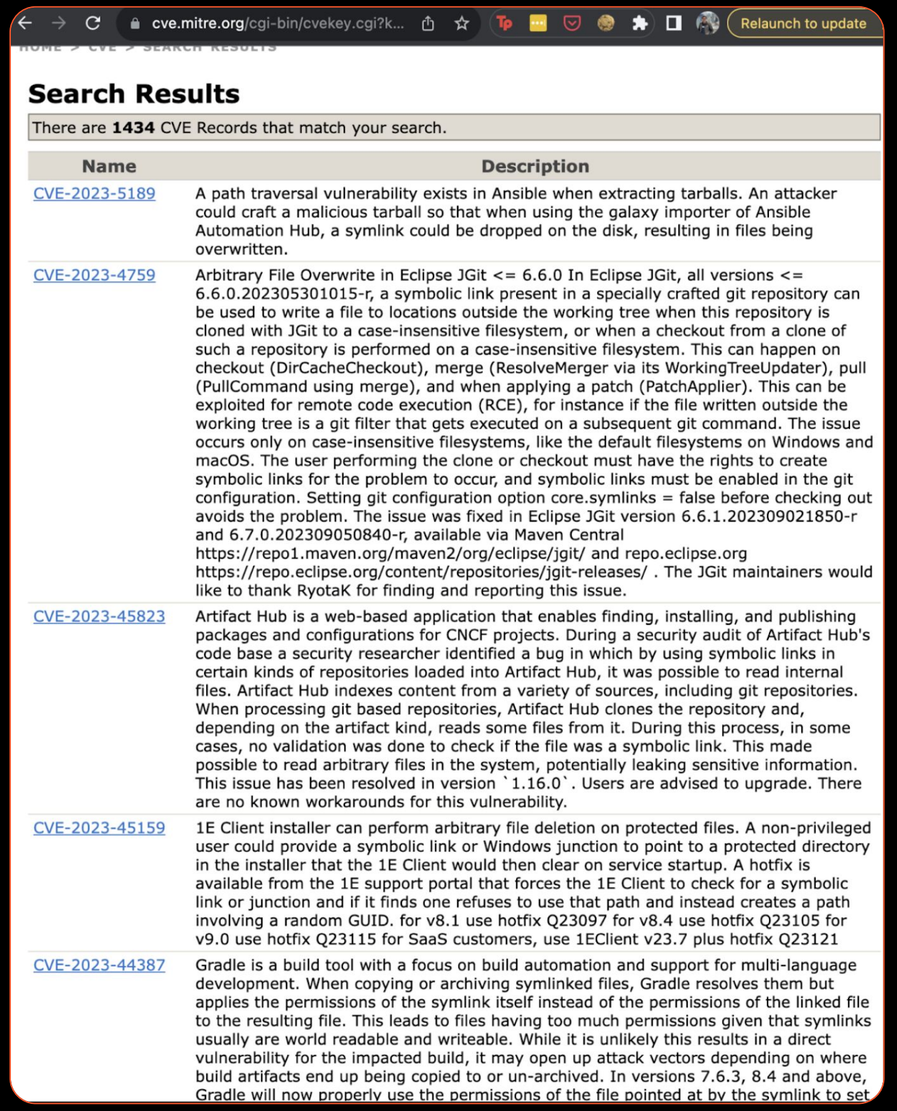
© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Time-of-Check/Time-of-Use - TOCTTOU



© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Landscape



cve.mitre.org/cgi-bin/cvekey.cgi?k... Relaunch to update

HOME > CVE > SEARCH RESULTS

Search Results

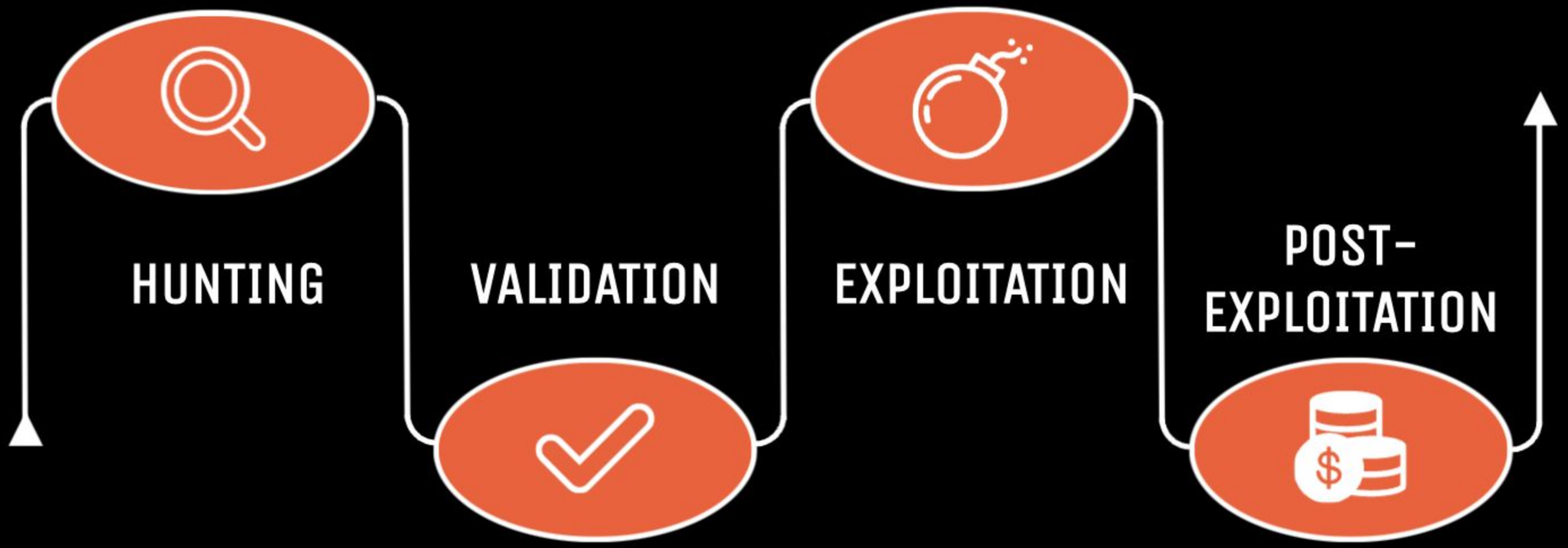
There are **1434** CVE Records that match your search.

Name	Description
CVE-2023-5189	A path traversal vulnerability exists in Ansible when extracting tarballs. An attacker could craft a malicious tarball so that when using the galaxy importer of Ansible Automation Hub, a symlink could be dropped on the disk, resulting in files being overwritten.
CVE-2023-4759	Arbitrary File Overwrite in Eclipse JGit <= 6.6.0 In Eclipse JGit, all versions <= 6.6.0.202305301015-r, a symbolic link present in a specially crafted git repository can be used to write a file to locations outside the working tree when this repository is cloned with JGit to a case-insensitive filesystem, or when a checkout from a clone of such a repository is performed on a case-insensitive filesystem. This can happen on checkout (DirCacheCheckout), merge (ResolveMerger via its WorkingTreeUpdater), pull (PullCommand using merge), and when applying a patch (PatchApplier). This can be exploited for remote code execution (RCE), for instance if the file written outside the working tree is a git filter that gets executed on a subsequent git command. The issue occurs only on case-insensitive filesystems, like the default filesystems on Windows and macOS. The user performing the clone or checkout must have the rights to create symbolic links for the problem to occur, and symbolic links must be enabled in the git configuration. Setting git configuration option core.symlinks = false before checking out avoids the problem. The issue was fixed in Eclipse JGit version 6.6.1.202309021850-r and 6.7.0.202309050840-r, available via Maven Central https://repo1.maven.org/maven2/org/eclipse/jgit/ and repo.eclipse.org https://repo.eclipse.org/content/repositories/jgit-releases/ . The JGit maintainers would like to thank RyotaK for finding and reporting this issue.
CVE-2023-45823	Artifact Hub is a web-based application that enables finding, installing, and publishing packages and configurations for CNCF projects. During a security audit of Artifact Hub's code base a security researcher identified a bug in which by using symbolic links in certain kinds of repositories loaded into Artifact Hub, it was possible to read internal files. Artifact Hub indexes content from a variety of sources, including git repositories. When processing git based repositories, Artifact Hub clones the repository and, depending on the artifact kind, reads some files from it. During this process, in some cases, no validation was done to check if the file was a symbolic link. This made possible to read arbitrary files in the system, potentially leaking sensitive information. This issue has been resolved in version `1.16.0`. Users are advised to upgrade. There are no known workarounds for this vulnerability.
CVE-2023-45159	1E Client installer can perform arbitrary file deletion on protected files. A non-privileged user could provide a symbolic link or Windows junction to point to a protected directory in the installer that the 1E Client would then clear on service startup. A hotfix is available from the 1E support portal that forces the 1E Client to check for a symbolic link or junction and if it finds one refuses to use that path and instead creates a path involving a random GUID. for v8.1 use hotfix Q23097 for v8.4 use hotfix Q23105 for v9.0 use hotfix Q23115 for SaaS customers, use 1EClient v23.7 plus hotfix Q23121
CVE-2023-44387	Gradle is a build tool with a focus on build automation and support for multi-language development. When copying or archiving symlinked files, Gradle resolves them but applies the permissions of the symlink itself instead of the permissions of the linked file to the resulting file. This leads to files having too much permissions given that symlinks usually are world readable and writeable. While it is unlikely this results in a direct vulnerability for the impacted build, it may open up attack vectors depending on where build artifacts end up being copied to or un-archived. In versions 7.6.3, 8.4 and above, Gradle will now properly use the permissions of the file pointed at by the symlink to set

Product	ID	Vulnerability	Arbitrary file	Reported	Fix
Symantec Endpoint Protection 12 & 14	CVE-2017-13680	TOCTOU in the quarantine GUI	Deletion Read	09/2017	Available 11/2017
Symantec Endpoint Protection 12 & 14	CVE-2018-5236	TOCTOU during file deletion	Deletion	11/2017	Available 06/2018
Symantec Endpoint Protection 12 & 14	CVE-2018-5237	Check bypass in file restore	Write	11/2017	Available 06/2018
AV product A	TBD	Over-privileged file deletion	Deletion	03/2018	In progress
AV product B	TBD	Over-privileged file restore	Write	05/2018	In progress
McAfee Endpoint Security 10	CVE-2019-3582	Overpermissive access rights Over-privileged file creation	Write Deletion	05/2018	Available 10/2018 & 02/2019
AV product C	TBD	TOCTOU during file deletion	Deletion	05/2018	In progress
AV product D	TBD	TOCTOU during file deletion	Deletion	05/2018	In progress
F-Secure SAFE/CS/CP	(none)	Over-privileged file copy	Write Read Delete	07/2018	Available 08/2018

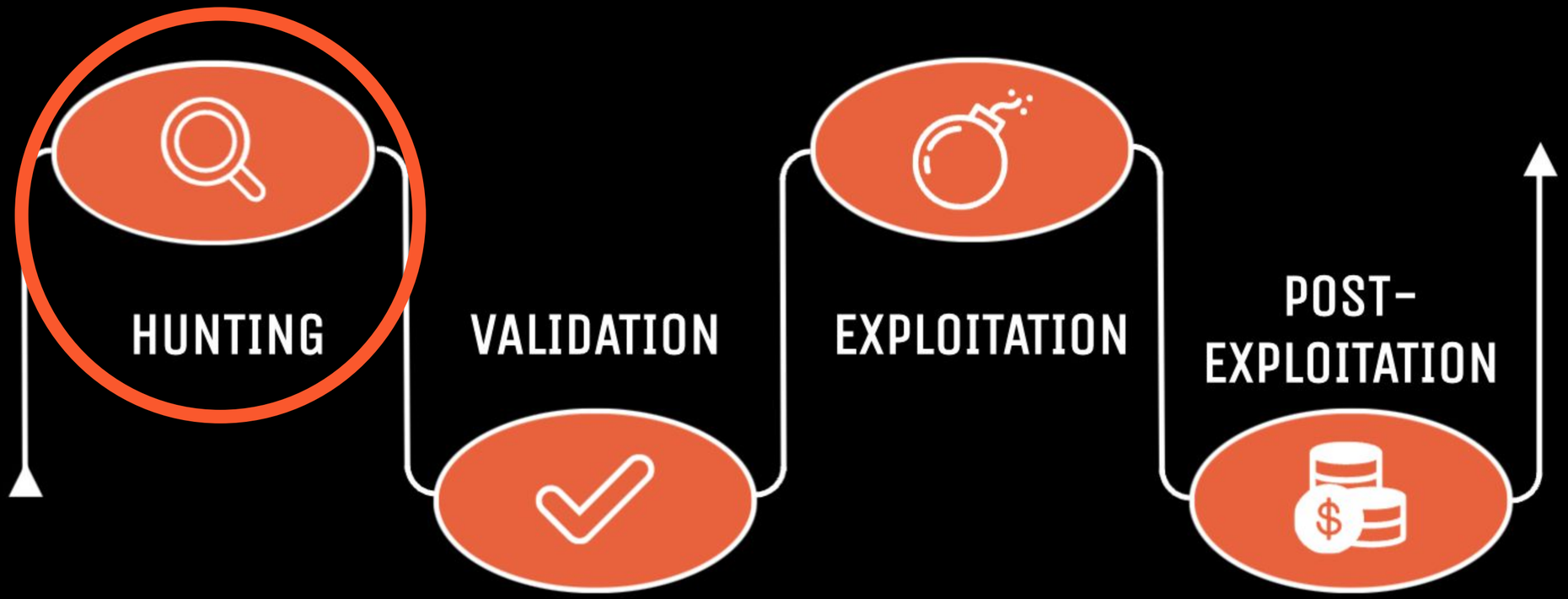
© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Vulnerability Research Process Flow



© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Vulnerability Research Process Flow - HUNT!



© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Static testing… most likely won’t help us here

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Static analysis won’t save you…

- The **challenge** for developers and testers to **identify** this issues **through manual code reviews**; automated code analysis (typically based on templates or rules), SAST; usage of linters; dependency scanning; etc. relies on the lack of execution CONTEXT.
  - The APIs used for file system operation are not deprecated nor insecure.
  - The logic of the program could be too complex to identify all the use cases scenarios of the file system operation.
  - Many code scanners and automated do not even look for this type of issues.
  - Many others do not even have the capability to follow the logic and identify.
- The only “static” method that could prevent your applications from having this issues is threat modeling during the design phase… -> Shifting left.

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

What about dynamic “manual” testing

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

…Yea, but it might take longer and we might lost some info

- Process Monitor
  - Filters on the product's privileged processes

- Debugger

- Process Explorer
  - icacls
  - AccessChk
  - Get-Acl
  - Any way to view ACLs on files / folders

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Instrumentation might save us

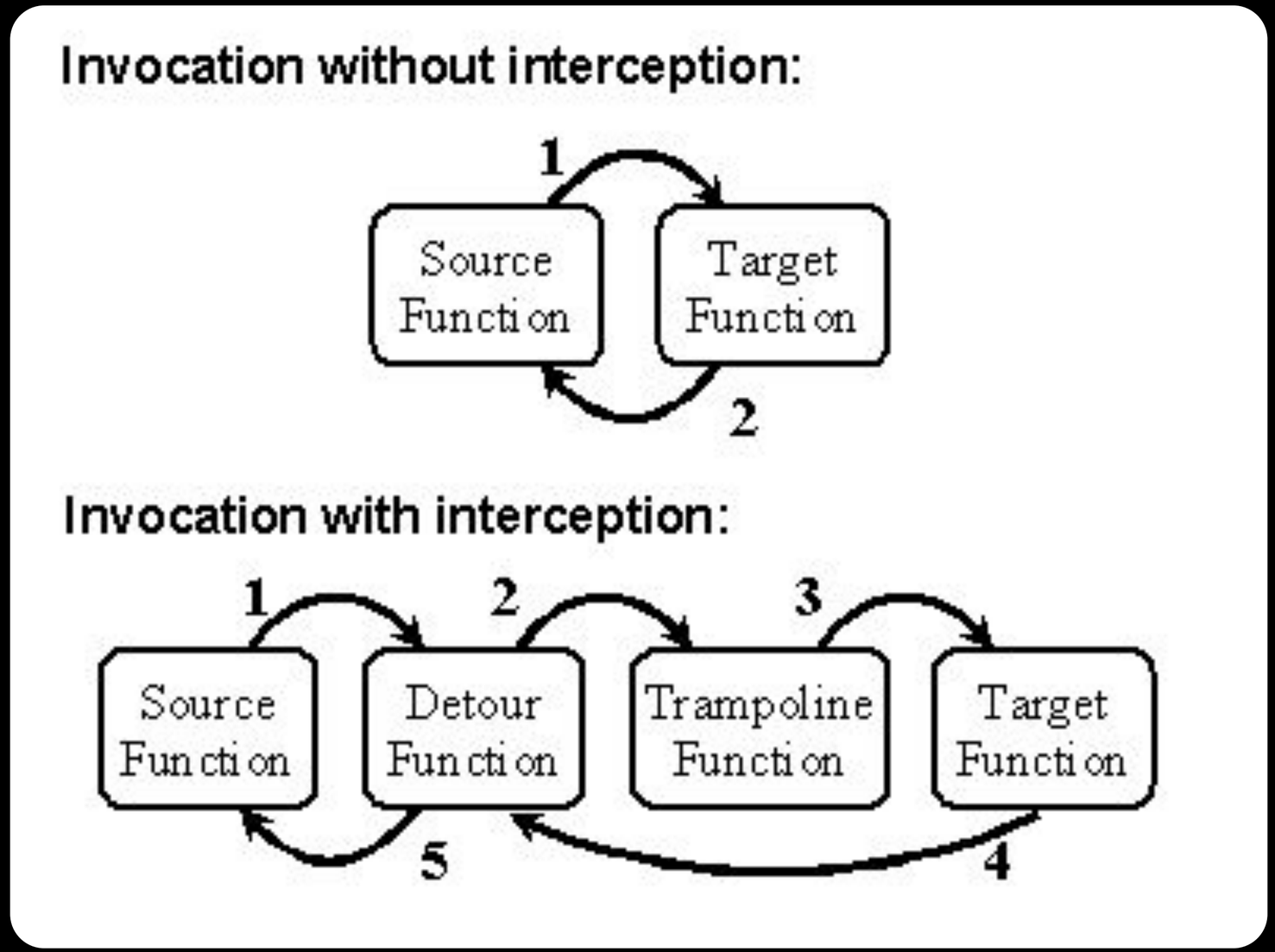
© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Dynamic Instrumentation

- Dynamic binary instrumentation is the **process of modifying the instructions of a binary program during its execution.**
- It **uses dynamic code injection techniques.** No source code modification needed.
  - **No special preparation** or **recompilation** of the executable is **necessary**, since the **instrumentation code is generated during** the **execution** of the application.
- Some uses:
  - Testing
  - Debugging & Reverse Engineering.
  - Software Analysis
  - Performance analysis
  - Program optimization and quality assurance.
  - **H4cKING**

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Trampoline functions



Detours' documentation diagram

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Most common tool used for instrumentation…

FRIDA

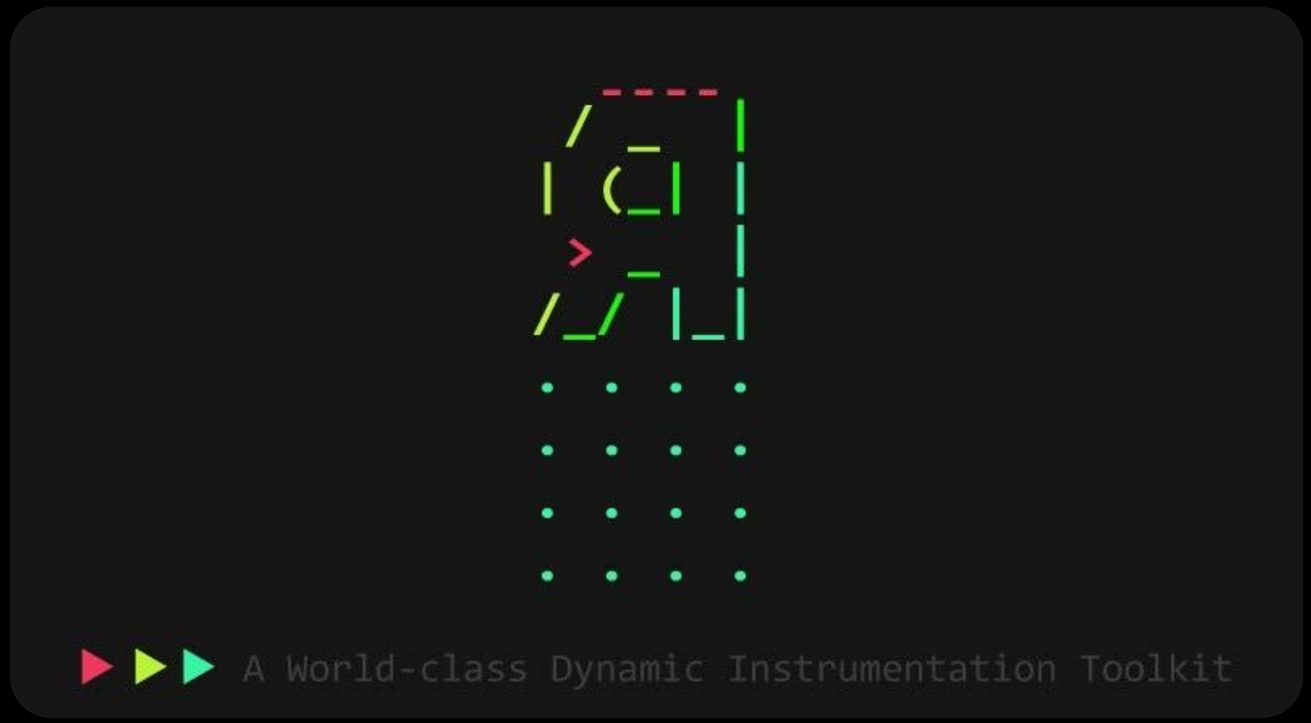
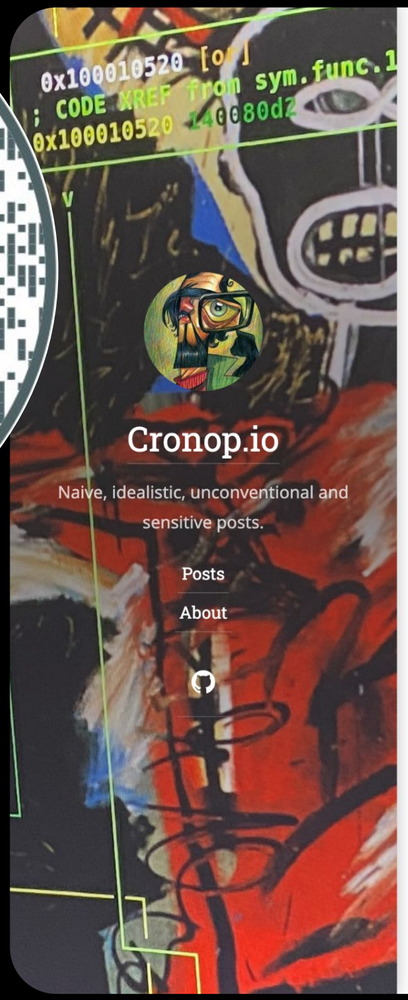


Image source: https://miro.medium.com/v2/resize:fit:1400/format:webp/1*3L7zcl_sv9NF68EKZmAjGw.jpeg

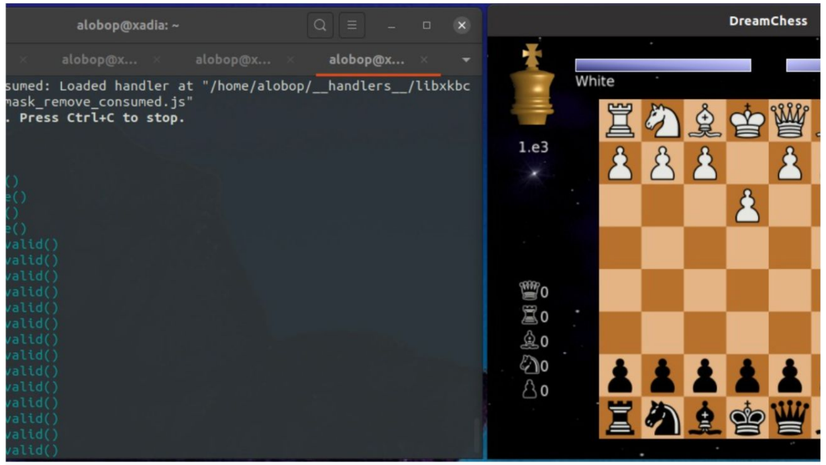
© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Dreamchess for Linux vs Stockfish



Instrumenting a Chess game with FЯIDA

25 Jun 2020 in Posts / Binary analysis on Radare2, Gdb, Reversing, Frida, Instrumentation



Instrumenting dreamchess for Linux to play against StockFish, the famous chess engine.

Table of contents

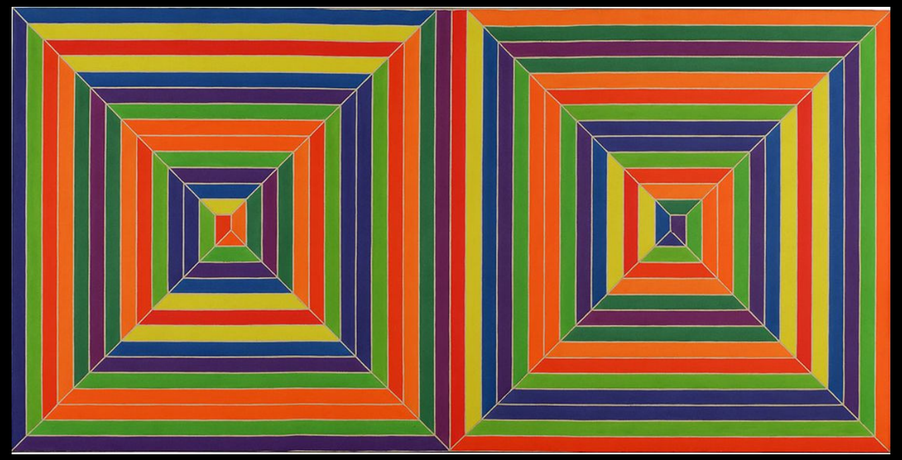
- Table of contents
- Requirements
- Static binary analysis
- Dynamic binary analysis

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Native-First Development / Minimalist software Design

- "**Native-First Development**": Prioritizing native tools and solutions that come built into the operating system or development environment before considering third-party options.

- "Minimalist Software Design" (Not the UI concept): Emphasizes using the least amount of external dependencies and the simplest possible approach to solve problems.



What you see is what you see - Frank Stella

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Microsoft Detours

README | Code of conduct | MIT license | MIT license | Security

Microsoft Research Detours Package

Detours is a software package for monitoring and instrumenting API calls on Windows. Detours has been used by many ISVs and is also used by product teams at Microsoft. Detours is now available under a standard open source license (MIT). This simplifies licensing for programmers using Detours and allows the community to support Detours using open source tools and processes.

Detours is compatible with the Windows NT family of operating systems: Windows NT, Windows XP, Windows Server 2003, Windows 7, Windows 8, and Windows 10. It cannot be used by Windows Store apps because Detours requires APIs not available to those applications. This repo contains the source code for version 4.0.1 of Detours.

For technical documentation on Detours, see the Detours Wiki. For directions on how to build and run samples, see the samples README.txt file.

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Detours’ advocate

Guided Hacking

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Using Detours - The simplest way is to use it through a DLL

```
BOOL WINAPI DllMain(HINSTANCE hinst, DWORD dwReason, LPVOID reserved)
{
    if (DetourIsHelperProcess()) {
        return TRUE;
    }

    if (dwReason == DLL_PROCESS_ATTACH) {
        DetourRestoreAfterWith();

        DetourTransactionBegin();
        DetourUpdateThread(GetCurrentThread());
        DetourAttach(&(PVOID&)TrueSleep, TimedSleep);
        DetourTransactionCommit();
    } else if (dwReason == DLL_PROCESS_DETACH) {
        DetourTransactionBegin();
        DetourUpdateThread(GetCurrentThread());
        DetourDetach(&(PVOID&)TrueSleep, TimedSleep);
        DetourTransactionCommit();
    }
    return TRUE;
}
```

A detour transaction is marked by calls to the DetourTransactionBegin API and the DetourTransactionCommit API

The DetourUpdateThread API enlists threads in the transaction so that their instruction pointers are appropriately updated when the transaction commits.

Interception of the target function is enabled by invoking the DetourAttach API within a detour transaction: API takes two arguments: the address of the target pointer and the pointer to the detour function.

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

But what should I instrument then?

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

First take a look at the desired API

CreateFileW function (fileapi.h)

Article • 02/08/2023 Feedback

In this article

Syntax
Parameters
Return value
Remarks
Show 2 more

Creates or opens a file or I/O device. The most commonly used I/O devices are as follows: file, file stream, directory, physical disk, volume, console buffer, tape drive, communications resource, mailslot, and pipe. The function returns a handle that can be used to access the file or device for various types of I/O depending on the file or device and the flags and attributes specified.

To perform this operation as a transacted operation, which results in a handle that can be used for transacted I/O, use the CreateFileTransacted function.

Syntax

C++ Copy

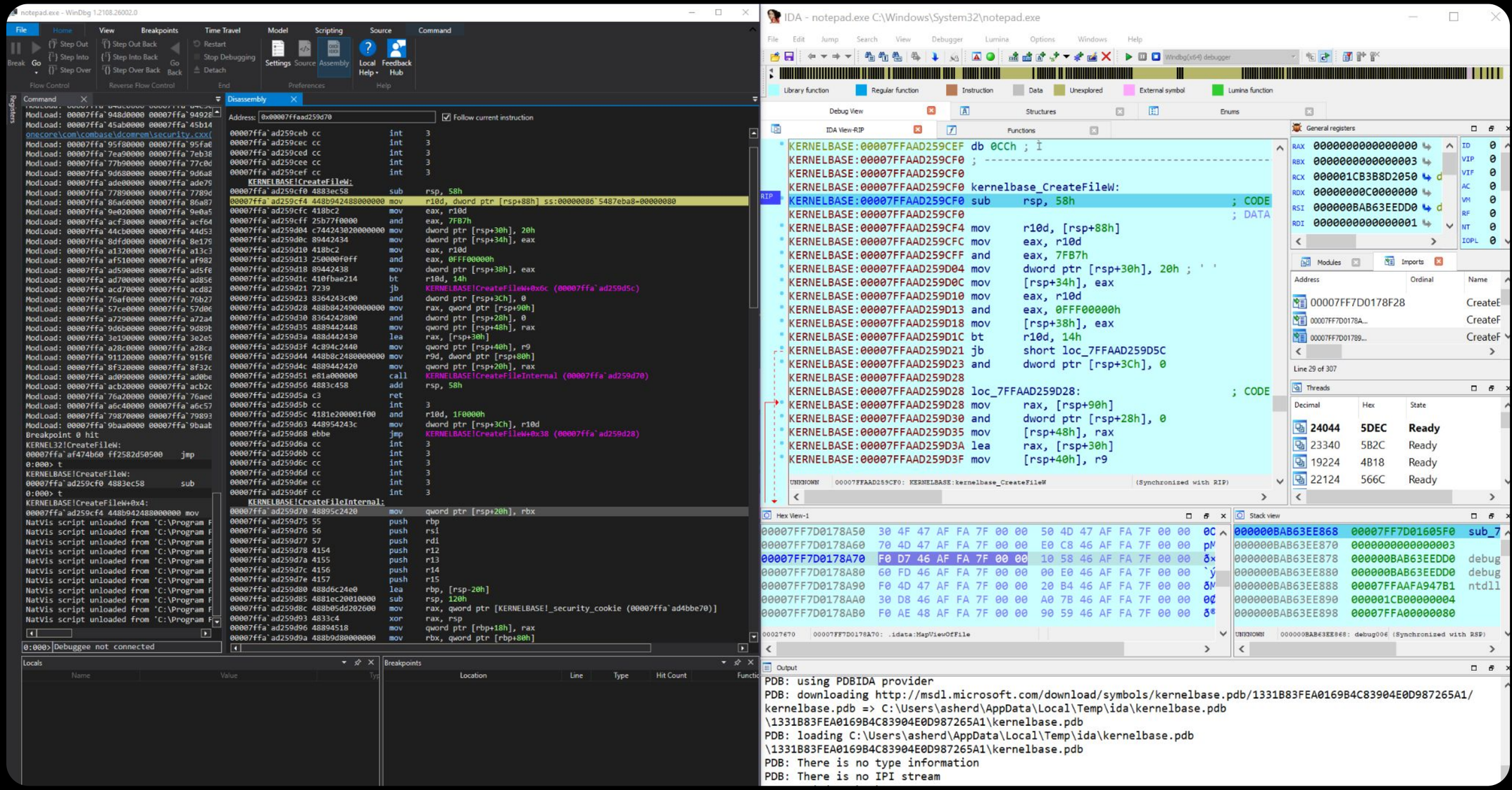
```
HANDLE CreateFileW(
  [in]           LPCWSTR               lpFileName,
  [in]           DWORD                 dwDesiredAccess,
  [in]           DWORD                 dwShareMode,
  [in, optional] LPSECURITY_ATTRIBUTES lpSecurityAttributes,
  [in]           DWORD                 dwCreationDisposition,
  [in]           DWORD                 dwFlagsAndAttributes,
  [in, optional] HANDLE                hTemplateFile
);
```

Windows API documentation

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Always aim for the entrails

- Play it safe and always aim for the most inner function to avoid symbol problems.



© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Wrappers and more wrappers

```
180035f80  int64_t CreateFileW(int64_t arg1, int32_t arg2, enum FILE_SHARE_MODE arg3, int32_t arg4, int32_t arg5,
180035f80      int64_t arg6)

180035f94      int32_t var_28 = 0x20
180035f9c      int32_t var_24 = arg5 & 0x7fb7
180035fa8      int32_t var_20 = arg5 & 0xfff00000
180035fb1      if (test_bit(arg5, 0x14))
180035ff3          int32_t var_1c_1 = arg5 & 0x1f0000
180035fb3      else
180035fb3          int32_t var_1c = 0
180035fc5      int64_t var_10 = arg6
180035fcf      int64_t r9
180035fcf      int64_t var_18 = r9
180035fea      return CreateFileInternal(arg1, arg2, arg3, arg4, &var_28, 0)
```

```
int64_t CreateFileInternal(int64_t arg1, int32_t arg2, enum FILE_SHARE_MODE arg3, int32_t arg4, int32_t* arg5, int32_t arg6)
    if (rax_48 != 0 && rax_48 != -0x7e0 && *(rax_48 + 0x810) != 0)
        rcx_23 = *(rax_48 + 0x7f8)
    r15_2.b = rcx_23 == 0x6000200000000
uint32_t var_118_1 = rdi_3
int64_t var_120_1 = rbx_3
var_128.d = var_108
int32_t r14_2 = rax_2 & 0x5affa7
uint32_t r13_1 = r13 | 0x100080
var_130.d = r12_1
enum FILE_SHARE_MODE var_138_1 = arg3
var_140.d = r14_2
var_148.q = 0
HANDLE var_e8
NTSTATUS rax_53 = NtCreateFile(FileHandle: &var_e8, DesiredAccess: r13_1, ObjectAttributes: &var_88, IoStatusBlock: &var_e
```

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Most of the time everything converges

```
1800840e0  int64_t (* const ntdll:ZwCreateFile)() = 0xb2e6a
1800840e8  int64_t (* const ntdll:ZwQueryInformationFile)() = 0xb2e7a
1800840f0  int64_t (* const ntdll:ZwCreateSection)() = 0xb2e94
1800840f8  int64_t (* const ntdll:ZwQueryDirectoryFile)() = 0xb2ea6
180084100  int64_t (* const ntdll:RtlNtPathNameToDosPathName)() = 0xb2ebe
180084108  int64_t (* const ntdll:RtlGetNativeSystemInformation)() = 0xb2ec
180084110  int64_t (* const ntdll:ZwQuerySystemInformation)() = 0xb2efc
180084118  int64_t (* const ntdll:ZwUnmapViewOfSection)() = 0xb2f18
180084120  int64_t (* const ntdll:ZwMapViewOfSection)() = 0xb2f30
180084128  int64_t (* const ntdll:VerSetConditionMask)() = 0xb2f46
180084130  int64_t (* const ntdll:RtlVerifyVersionInfo)() = 0xb2f5c
```

```
18009db40  int64_t NtCreateFile()

18009db40  4c8bd1            mov     r10, rcx
18009db43  b855000000        mov     eax, 0x55
18009db48  f604250803fe7f01  test    byte [0x7ffe0308], 0x1
18009db50  7503              jne     0x18009db55  {0x7ffe0308}

18009db52  0f05              syscall
18009db54  c3                retn     {__return_addr}

18009db55  cd2e              int     0x2e
18009db57  c3                retn
```

```
; Exported entry 287. NtCreateFile
; Exported entry 1871. ZwCreateFile


public ZwCreateFile
ZwCreateFile proc near
mov     r10, rcx        ; NtCreateFile
mov     eax, 55h ; 'U'
test    byte ptr ds:7FFE0308h, 1
jnz     short loc_18009D815
```

```
syscall                 ; Low latency system call
retn
```

```
loc_18009D815:          ; DOS 2+ internal - EXECUTE COMMAND
int     2Eh             ; DS:SI -> counted CR-terminated command string
retn
ZwCreateFile endp
```

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Windows Syscalls table by j00ru

https://j00ru.vexillium.org/syscalls/nt/64/

Windows X86-64 System Call Table (XP/2003/Vista/7/8/10/2022/11)

Author: Mateusz "j00ru" Jurczyk (j00ru.vx tech blog)

See also: Windows System Call Tables in CSV/JSON formats on GitHub

Special thanks to: MeMek, Wandering Glitch

Layout by Metasploit Team

Enter the Syscall ID to highlight (hex):

Highlight

Show all | Hide all

System Call Symbol	Windows XP (show)		Windows Server 2003 (show)				Windows Vista (show)			Windows 7 (show)		Windows 8 (show)		Windows 10 (show)	Windows 11 and Server (show)				
NtAcceptConnectPort																			
NtAccessCheck																			
NtAccessCheckAndAuditAlarm																			
NtAccessCheckByType																			
NtAccessCheckByTypeAndAuditAlarm																			
NtAccessCheckByTypeResultList																			
NtAccessCheckByTypeResultListAndAuditAlarm																			
NtAccessCheckByTypeResultListAndAuditAlarmByHandle																			
NtAcquireCMFViewOwnership																			
NtAcquireCrossVmMutant																			
NtAcquireProcessActivityReference																			
NtAddAtom																			
NtAddAtomEx																			
NtAddBootEntry																			
NtAddDriverEntry																			
NtAdjustGroupsToken																			
NtAdjustPrivilegesToken																			
NtAdjustTokenClaimsAndDeviceGroups																			

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Snippets of Detours - Define function pointers

```
NTSTATUS(NTAPI* ActualNTCreateFile)
(PHANDLE, ACCESS_MASK, POBJECT_ATTRIBUTES, PIO_STATUS_BLOCK, PLARGE_INTEGER,
 ULONG, ULONG, ULONG, ULONG, PVOID, ULONG) = NULL;
```

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Snippets of Detours - Define function pointers

```
ActualNTCreateFile = ((NTSTATUS(NTAPI*)(
        PHANDLE, ACCESS_MASK, POBJECT_ATTRIBUTES, PIO_STATUS_BLOCK,
        PLARGE_INTEGER, ULONG, ULONG, ULONG, ULONG, PVOID,
        ULONG))DetourFindFunction("ntdll.dll", "NtCreateFile"));
```

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Snippets of Detours - Define the SIDs (Security identifiers) you are interested in

```
VOID convertSIDs(VOID)
{
  const char* ccpSIDsOfInterest[] = {
      "S-1-5-11" /*Authenticated User*/,
      "S-1-5-13" /*Terminal Server user*/,
      "S-1-5-32-555" /*Remote Desktop user*/,
      "S-1-5-4" /*Interactive*/,
      "S-1-5-32-545" /*Users*/,
      "S-1-2-1" /*Console login*/,
      "S-1-5-32-546" /*Guests*/,
      "S-1-5-7" /*Anonymous*/,
      "S-1-1-0" /*Everyone*/,
      "S-1-16-4096" /*LowIL*/,
      "S-1-5-12" /*RC*/,
      "S-1-15-2-1" /*ac*/,
      "S-1-15-3-1" /*Internet client*/
  };
```

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Snippets of Detours

```
NTSTATUS APIENTRY HookedNtCreateFile(
    PHANDLE FileHandle, ACCESS_MASK DesiredAccess,
    POBJECT_ATTRIBUTES ObjectAttributes, PIO_STATUS_BLOCK IoStatusBlock,
    PLARGE_INTEGER AllocationSize, ULONG FileAttributes, ULONG ShareAccess,
    ULONG CreateDisposition, ULONG CreateOptions, PVOID EaBuffer,
    ULONG EaLength)
{
  const IL ilIntegrity = getTokenAndIL();

  if (ilIntegrity <= IL::MEDIUM)
  {
    return ActualNTCreateFile(FileHandle, DesiredAccess, ObjectAttributes,
                              IoStatusBlock, AllocationSize, FileAttributes,
                              ShareAccess, CreateDisposition, CreateOptions,
                              EaBuffer, EaLength);
  }

  const DWORD dwCreateOptionsAction = CreateOptions & FILE_OPERATION_MASK;
  const PWSTR psFileName = ObjectAttributes->ObjectName->Buffer;
  const bool isILOfInterest = getACL(psFileName);

  if (isILOfInterest)
  {

    if (dwCreateOptionsAction == 0x60 &&
        (CreateDisposition == FILE_CREATE ||
         CreateDisposition == FILE_OVERWRITE ||
         CreateDisposition == FILE_OVERWRITE_IF))
    {
      ETW_INFO("NTCreateFile", TraceLoggingString("Create", "Operation"),
               TraceLoggingWideString(psFileName, "File"));
    }
  }

  return ActualNTCreateFile(FileHandle, DesiredAccess, ObjectAttributes,
                            IoStatusBlock, AllocationSize, FileAttributes,
                            ShareAccess, CreateDisposition, CreateOptions,
                            EaBuffer, EaLength);
}
```

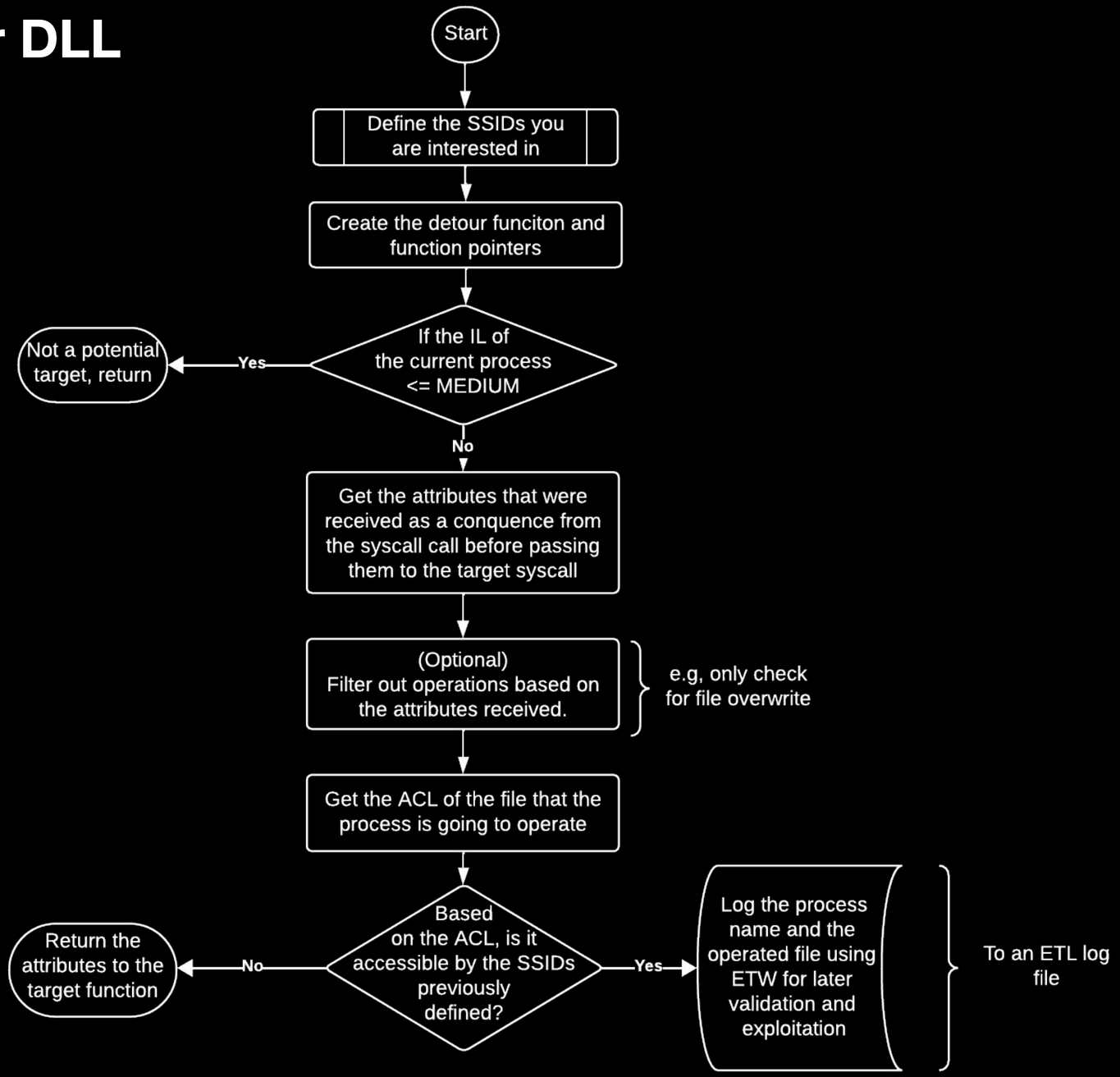
© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

DLLMain

```
__declspec(dllexport) BOOL APIENTRY
    DllMain(HMODULE hModule, DWORD dw_reason_for_call, LPVOID lpReserved)
{
  switch (dw_reason_for_call)
  {
    case DLL_PROCESS_ATTACH:
    {
      DllInjectTraceInit();

      ActualNTCreateFile = ((NTSTATUS(NTAPI*)(
          PHANDLE, ACCESS_MASK, POBJECT_ATTRIBUTES, PIO_STATUS_BLOCK,
          PLARGE_INTEGER, ULONG, ULONG, ULONG, ULONG, PVOID,
          ULONG))DetourFindFunction("ntdll.dll", "NtCreateFile"));

      if ((ActualNTCreateFile == NULL || ActualNtOpenFile == NULL))
      {
        return FALSE;
      }

      convertSIDs();

      DetourTransactionBegin();
      DetourUpdateThread(GetCurrentThread());
      DetourAttach(&(PVOID&)ActualNTCreateFile, HookedNtCreateFile);

      const LONG lError = DetourTransactionCommit();

      if (lError != NO_ERROR)
      {
        ETW_ERROR("Error Attaching", TraceLoggingHexInt32(lError, "Error"));

        return FALSE;
      }
    }
    break;

    case DLL_PROCESS_DETACH:
    {
      DetourTransactionBegin();
      DetourUpdateThread(GetCurrentThread());
      DetourDetach(&(PVOID&)ActualNTCreateFile, HookedNtCreateFile);

      const LONG lError = DetourTransactionCommit();

      if (lError != NO_ERROR)
      {
        ETW_ERROR("Error Detaching", TraceLoggingHexInt32(lError, "Error"));
      }

      DllInjectTraceDeInit();
    }
    break;

    case DLL_THREAD_ATTACH:
      break;
    case DLL_THREAD_DETACH:
      break;
  }

  return TRUE;
}
```

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

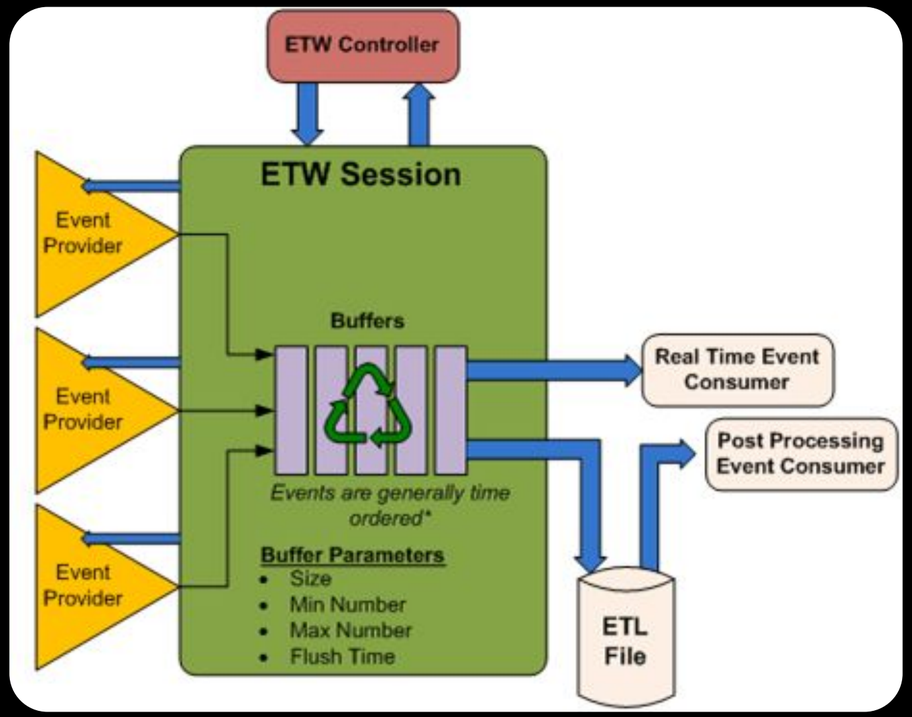
Our Detour DLL



© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

ETW: Relaying on native tools

- Event Tracing for Windows (ETW) offers a way to monitor and log events from both user-mode applications and kernel-mode drivers. Built into the Windows operating system, ETW provides developers with a fast, reliable, and flexible toolkit for event tracing.
- Super efficient and super reliable. You won’t have deadlocks.

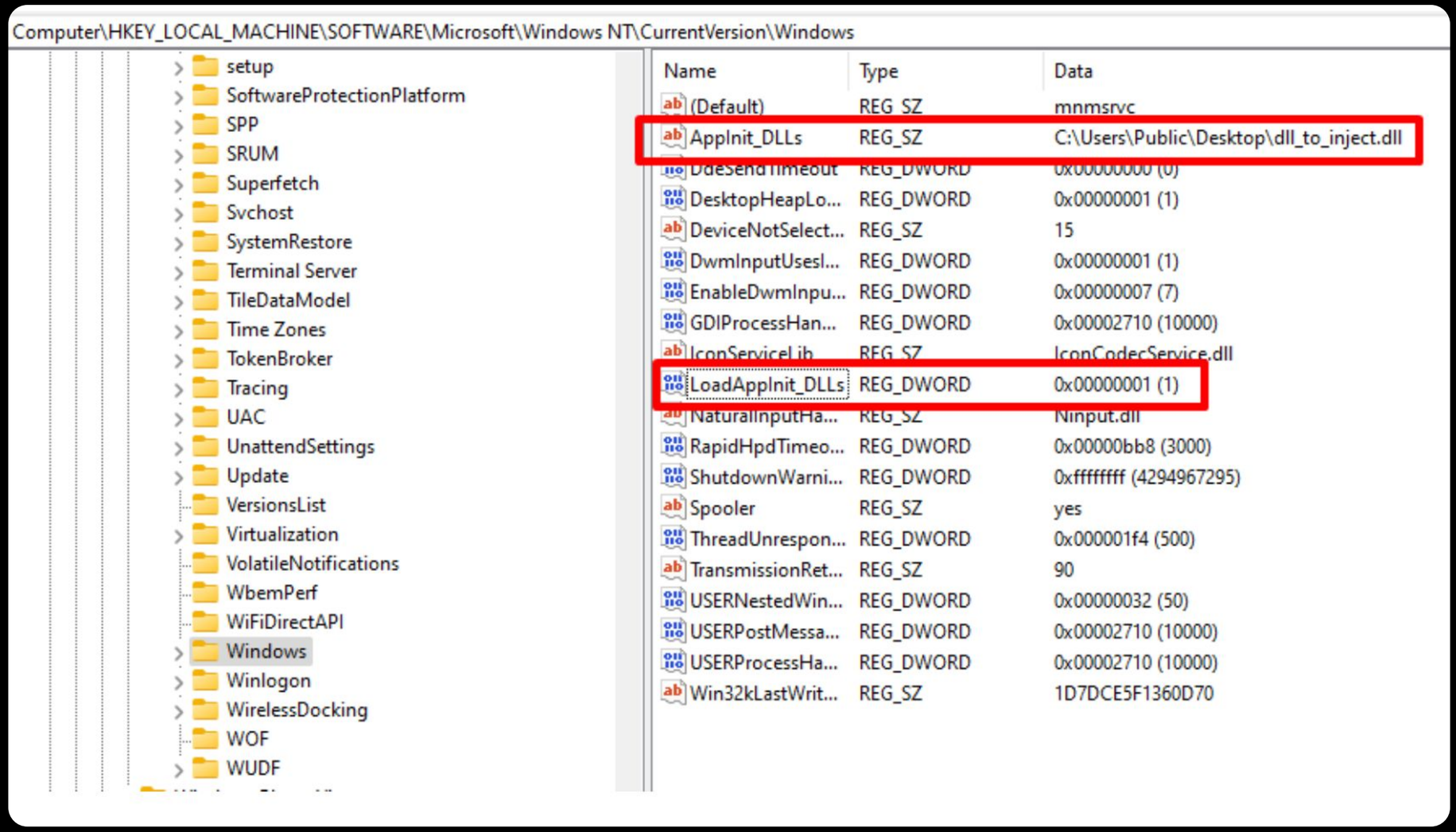


Event Tracing for Windows documentation

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Inject everywhere

- The AppInit_DLLs infrastructure provides an easy way to hook system APIs by allowing custom DLLs to be loaded into the address space of every interactive application.



© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Vulnerability Research Process Flow - Is it actually something that can be exploitable?

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Now that we have a target it’s a good time for using them

- Process Monitor
  - Filters on the product’s privileged processes

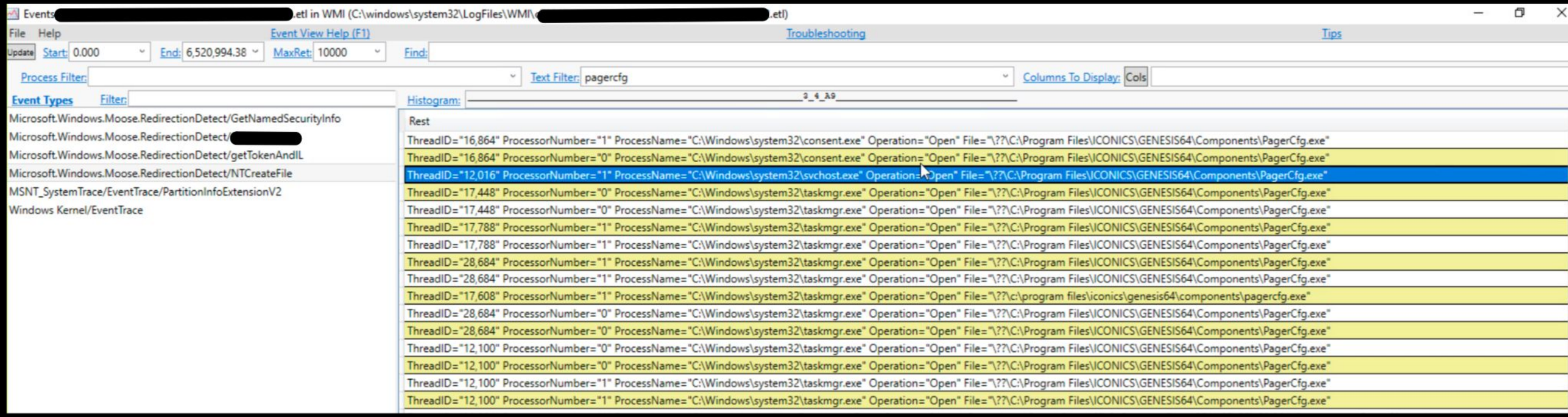
- Debugger
- **ETW viewers**

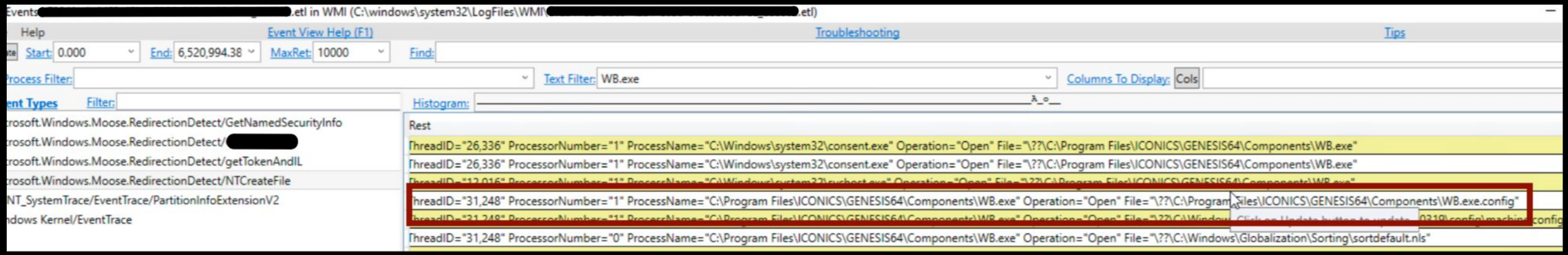
- Process Explorer
  - icacls
  - AccessChk
  - Get-Acl
  - Any way to view ACLs on files / folders

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Analyzing results

- Use the ETW viewer of your preference.





© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

What is “Iconics”?

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Iconics Suite - Critical SCADA Software by Mitsubishi

- Iconics Suite is an HMI SCADA solution suite that, according to its website, has over 375,000 unique installations in over 100 countries. This suite is commonly used in critical infrastructure sectors such as:
  - Government
  - Military
  - Oil and gas
  - Automotive
  - Manufacturing
  - Water and wastewater
  - Utilities
  - Energy

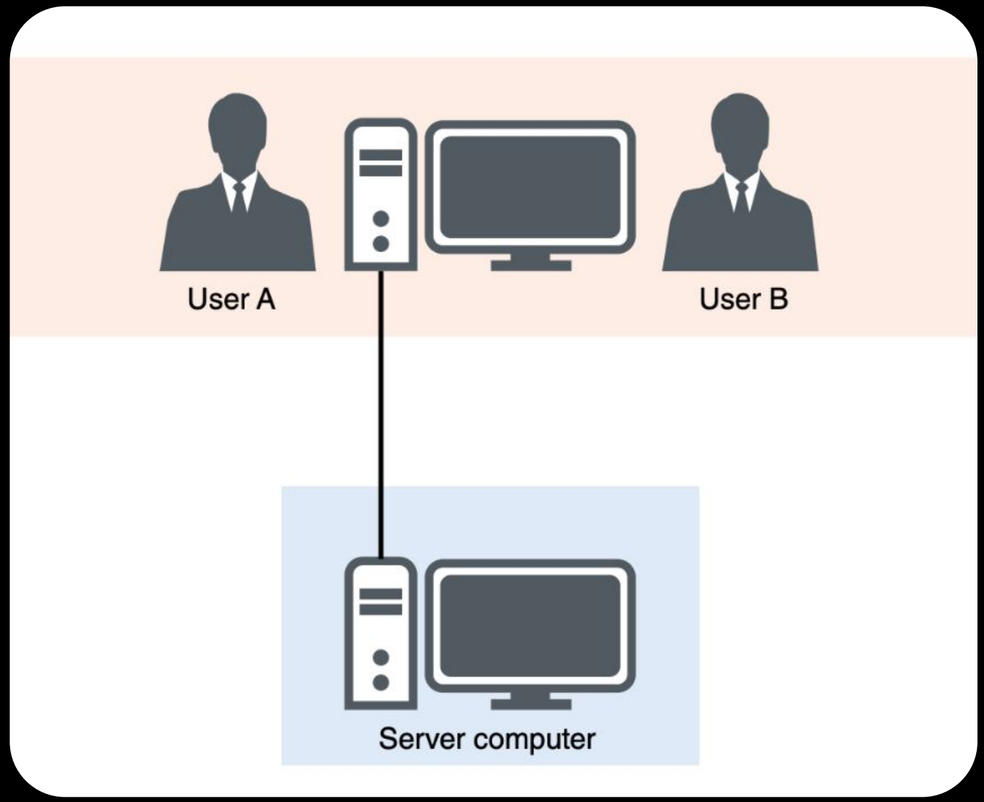
ICONICS website image

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Iconics Suite - Critical SCADA Software by Mitsubishi

- It allows the administration to assign multiple roles in the software. Which is critical for our exploitation scenario.

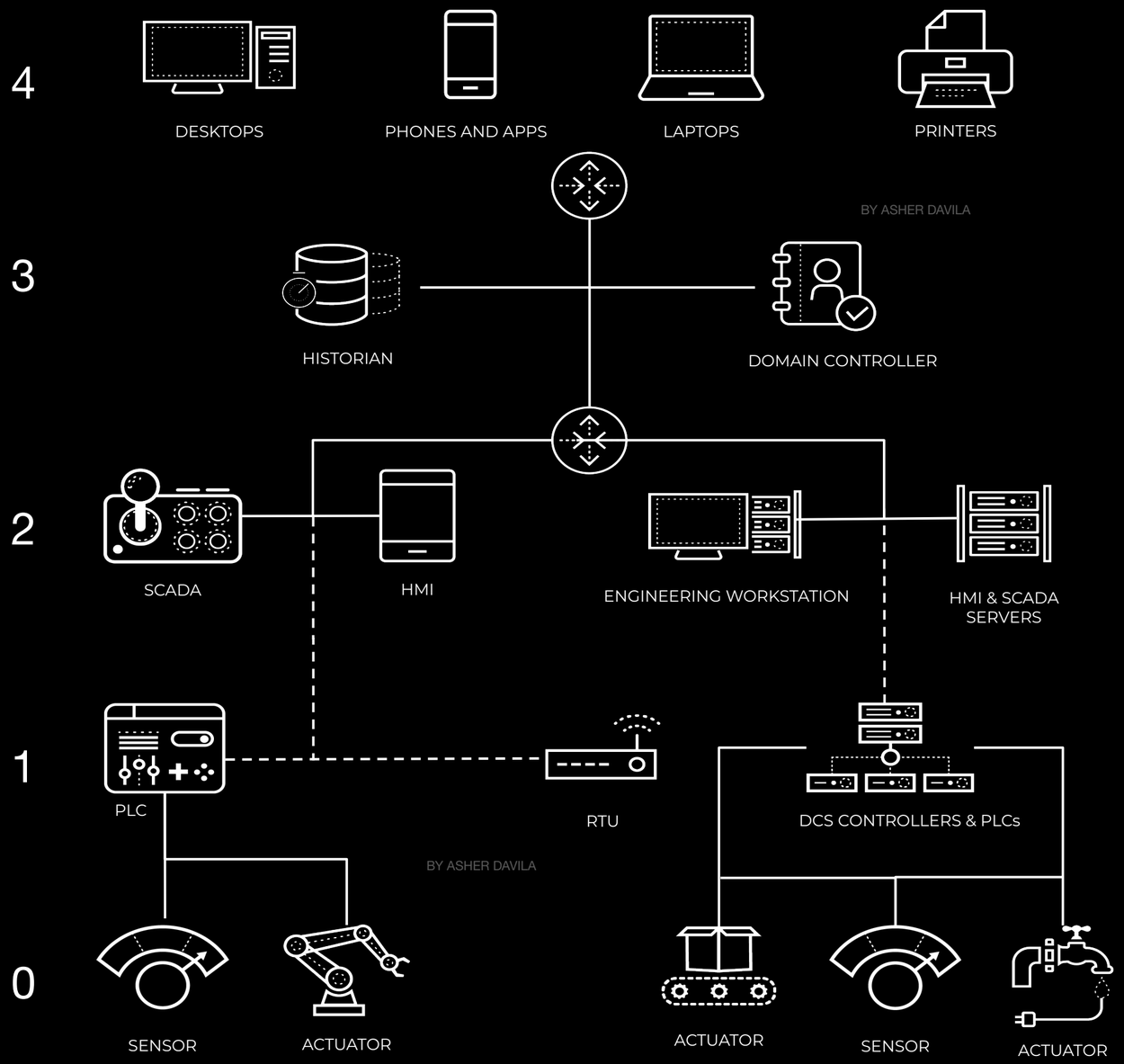
😈



ICONICS Suite documentation

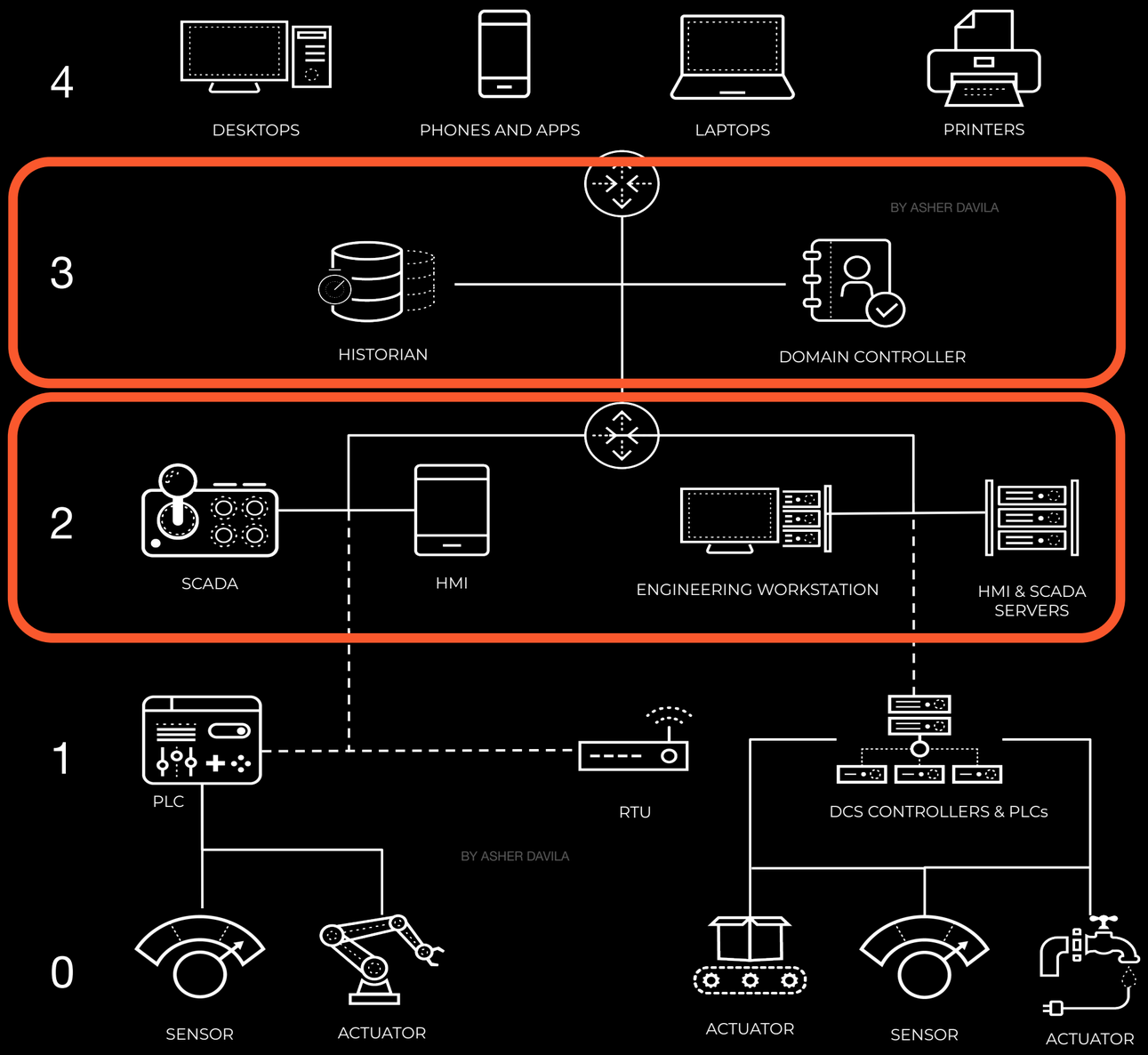
© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Yet again, let's refer to the Purdue model



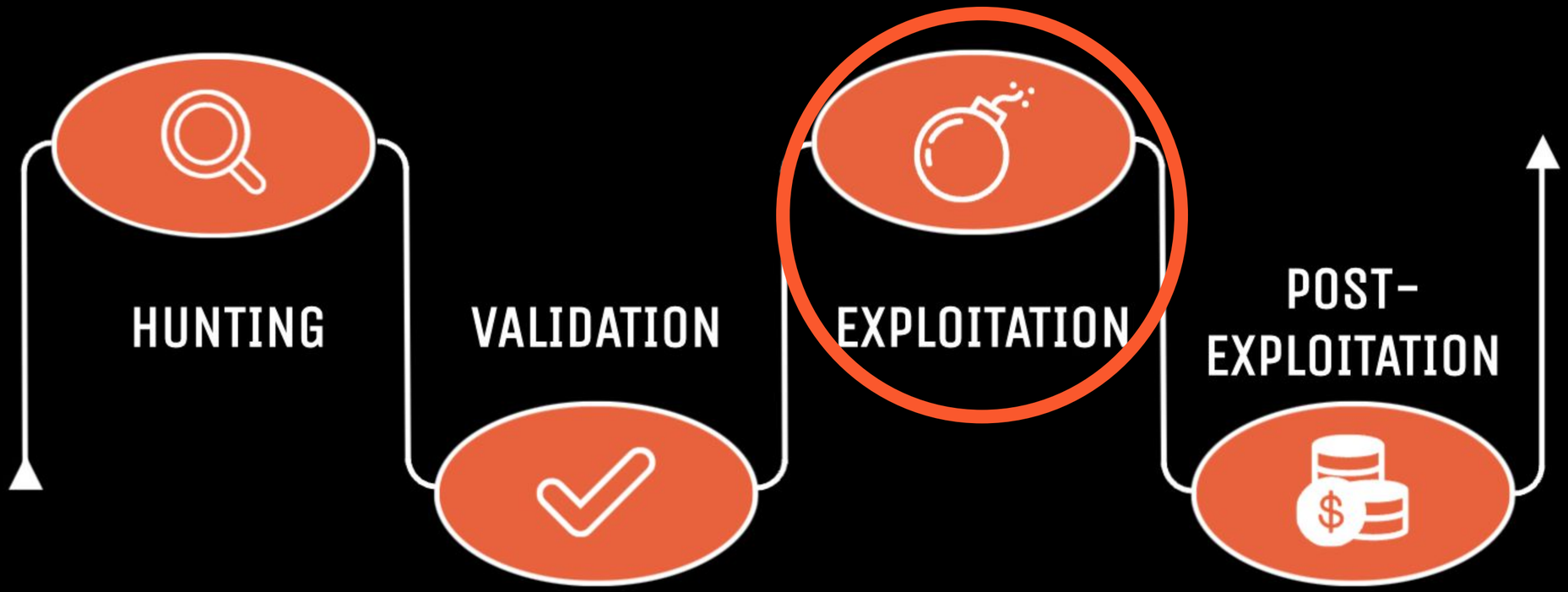
© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Yet again, let's refer to the Purdue model



© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Vulnerability Research Process Flow - h4Ck



© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Now what? How do I exploit it?

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

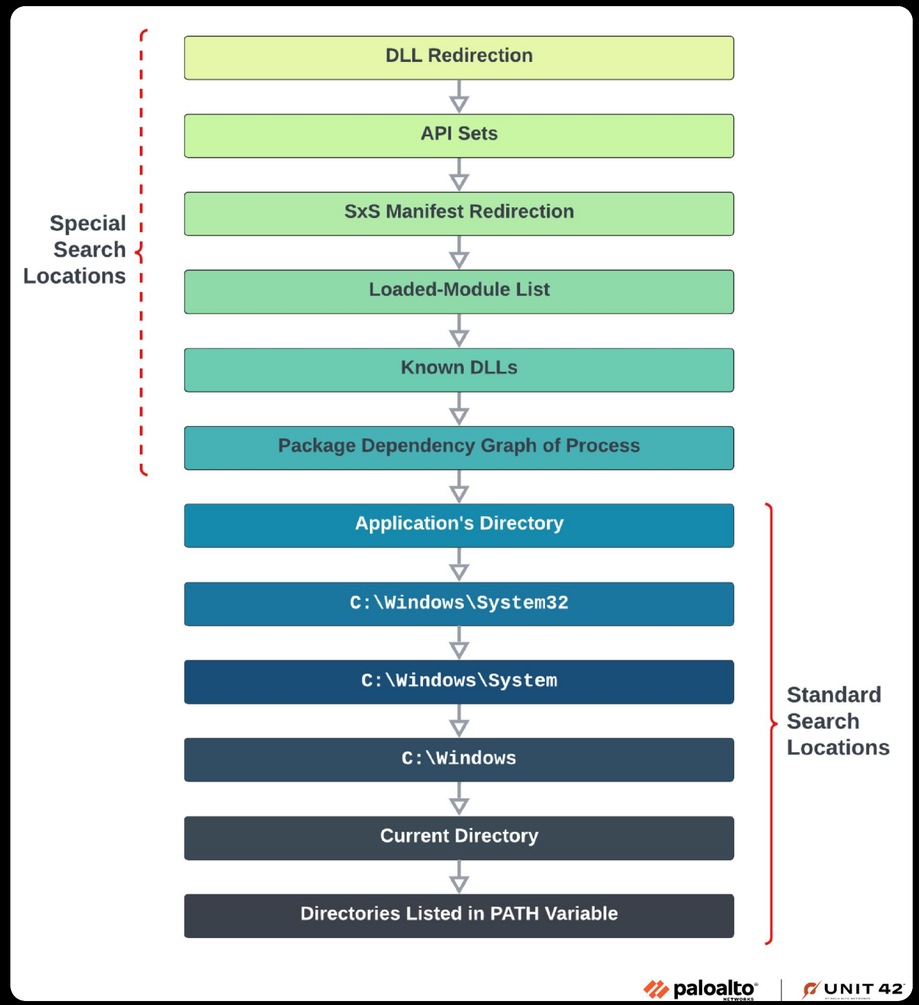
Multiple DLL Phantom hijacking vulnerabilities found on different components in ICONICS Suite

By Asher Davila and Malav Vyas, Palo Alto Networks Researchers

Iconics Suite is a collection of software tools and solutions primarily focused on automation, building management, manufacturing, and industrial applications. It offers a range of functionalities including:

- SCADA (Supervisory Control and Data Acquisition): Provides real-time monitoring and control of industrial, infrastructure, and facility-based processes.
- HMI (Human-Machine Interface): Offers interactive interfaces for operators to monitor and manage industrial and building automation systems.
- Building Automation and Energy Management: Helps in managing and optimizing building systems like HVAC, lighting, and power systems for efficiency and sustainability.
- Manufacturing Intelligence: Provides analytics and reporting tools for optimizing manufacturing processes and improving productivity.
- Asset Management: Assists in managing and tracking the performance and maintenance of industrial assets.

Phantom DLL Hijacking is a cybersecurity attack method where an attacker takes advantage of the way applications load Dynamic Link Libraries (DLLs). Unlike DLL hijacking, which involves replacing a legitimate DLL with a malicious one, Phantom DLL Hijacking involves reintroducing an obsolete or no longer used legitimate DLL back into the system. This obsolete DLL is modified to perform malicious activities. This attack exploits the process by which applications load externa DLL files. It is a variant of DLL hijacking but with a subtle difference in approach. In Phantom DLL Hijacking, an attacker places an obsolete or unused legitimate DLL into a location where the application would typically load it. The application, thinking it is loading a genuine and required DLL, executes the code within the phantom DLL..



© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Exploitation and tooling via file system redirection: *Standing on the shoulders of giants*

- James Forshaw
- Clavoillotte
- Mentors and colleagues

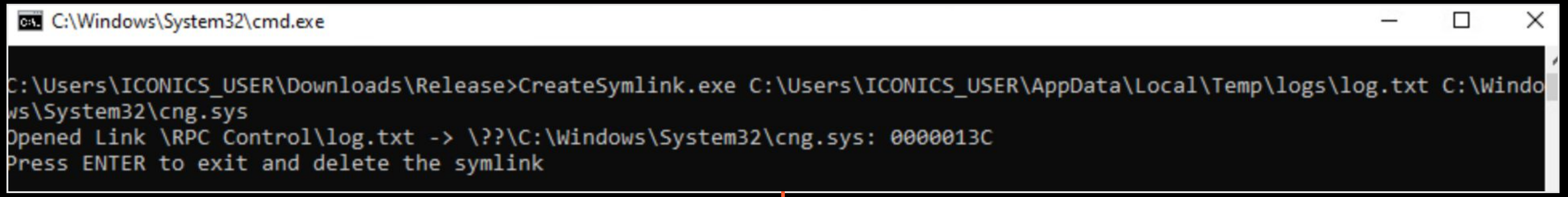
Symboliclink-testing-tools-toolkit & NtApiDotnet

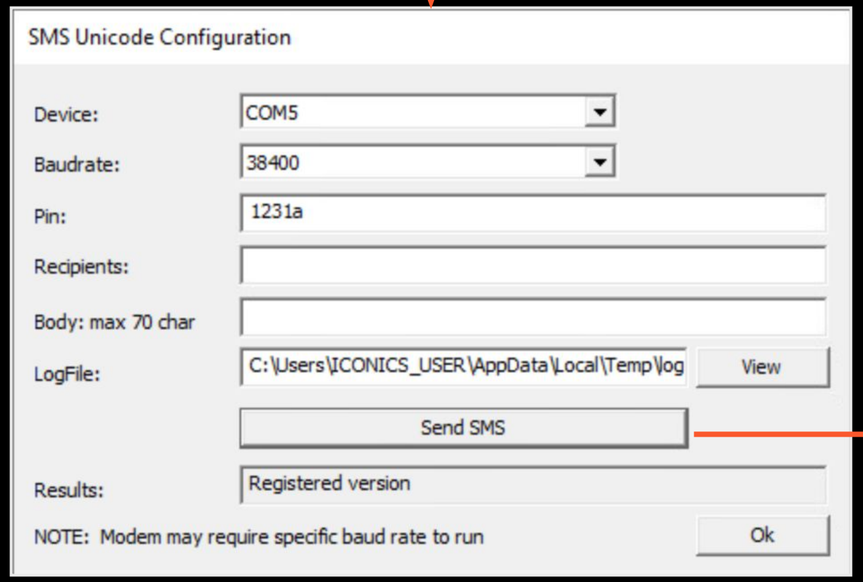
```
symboliclink-testing-tools

(c) Google Inc. 2015
Developed by James Forshaw

This is a small suite of tools to test various symbolic link types of Windows. It consists of the
following
tools:

BaitAndSwitch : Creates a symbolic link and uses an OPLOCK to win a TOCTOU
CreateDosDeviceSymlink: Creates a object manager symbolic link using csrss
CreateMountPoint: Create an arbitrary file mount point
CreateNtfsSymlink: Create an NTFS symbolic link
CreateObjectDirectory: Create a new object manager directory
CreateRegSymlink: Create a registry key symbolic link
DeleteMountPoint: Delete a mount point
DumpReparsePoint: Delete the reparse point data
NativeSymlink: Create an object manager symbolic link
SetOpLock: Tool to create oplocks on arbitrary files or directories

The tools can be built with Visual Studio 2013
```

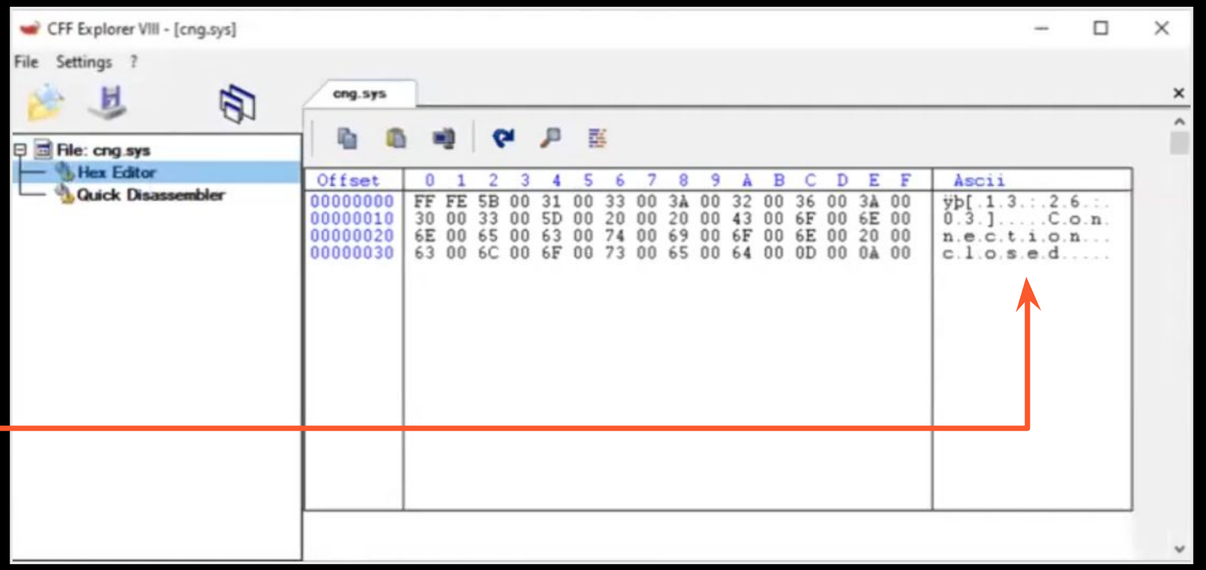
- https://github.com/googleprojectzero/symboliclink-testing-tools
- https://github.com/googleprojectzero/sandbox-attacksurface-analysis-tools/tree/main/NtApiDotNet

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

The attack: Create a corrupted driver

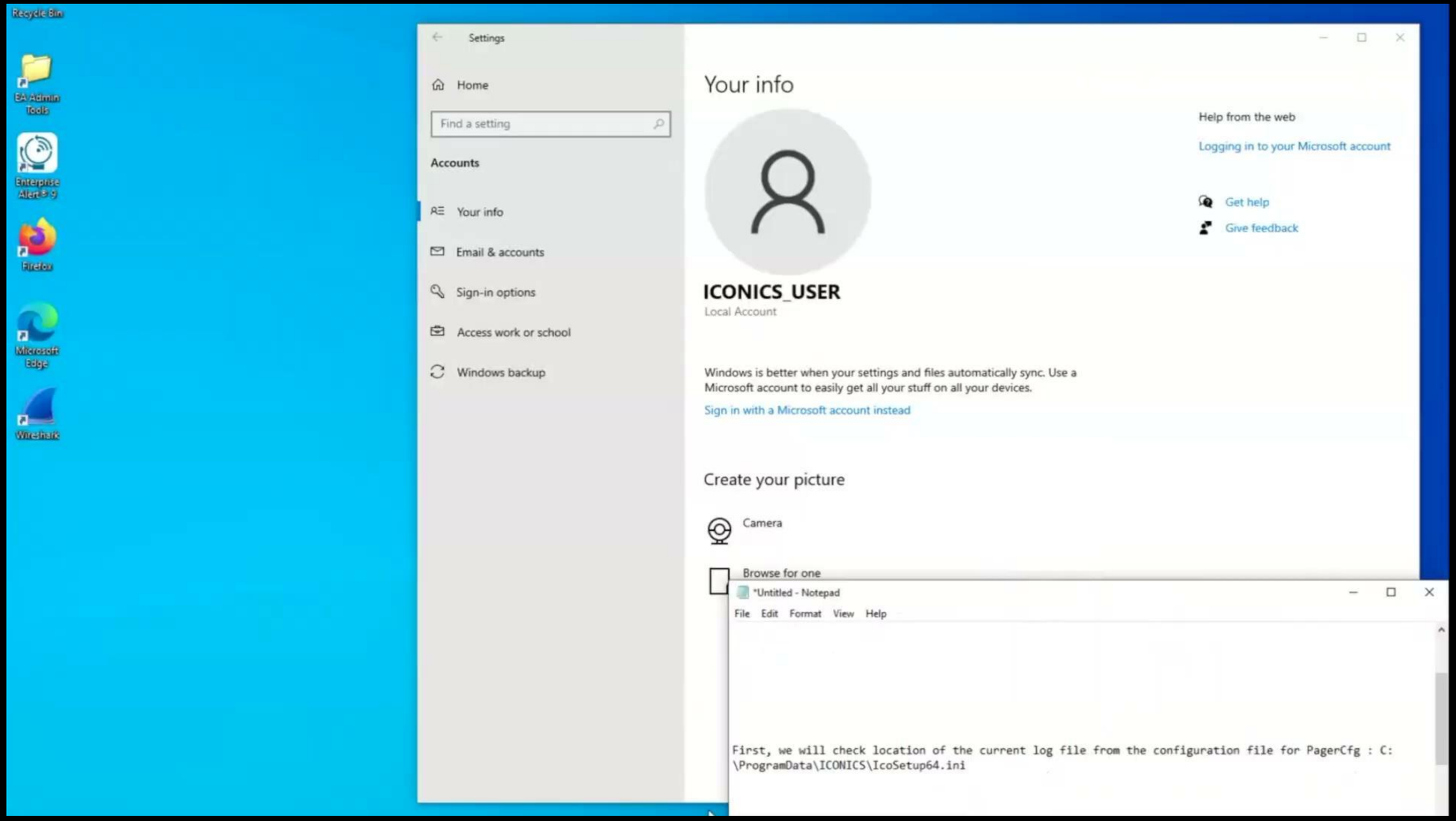






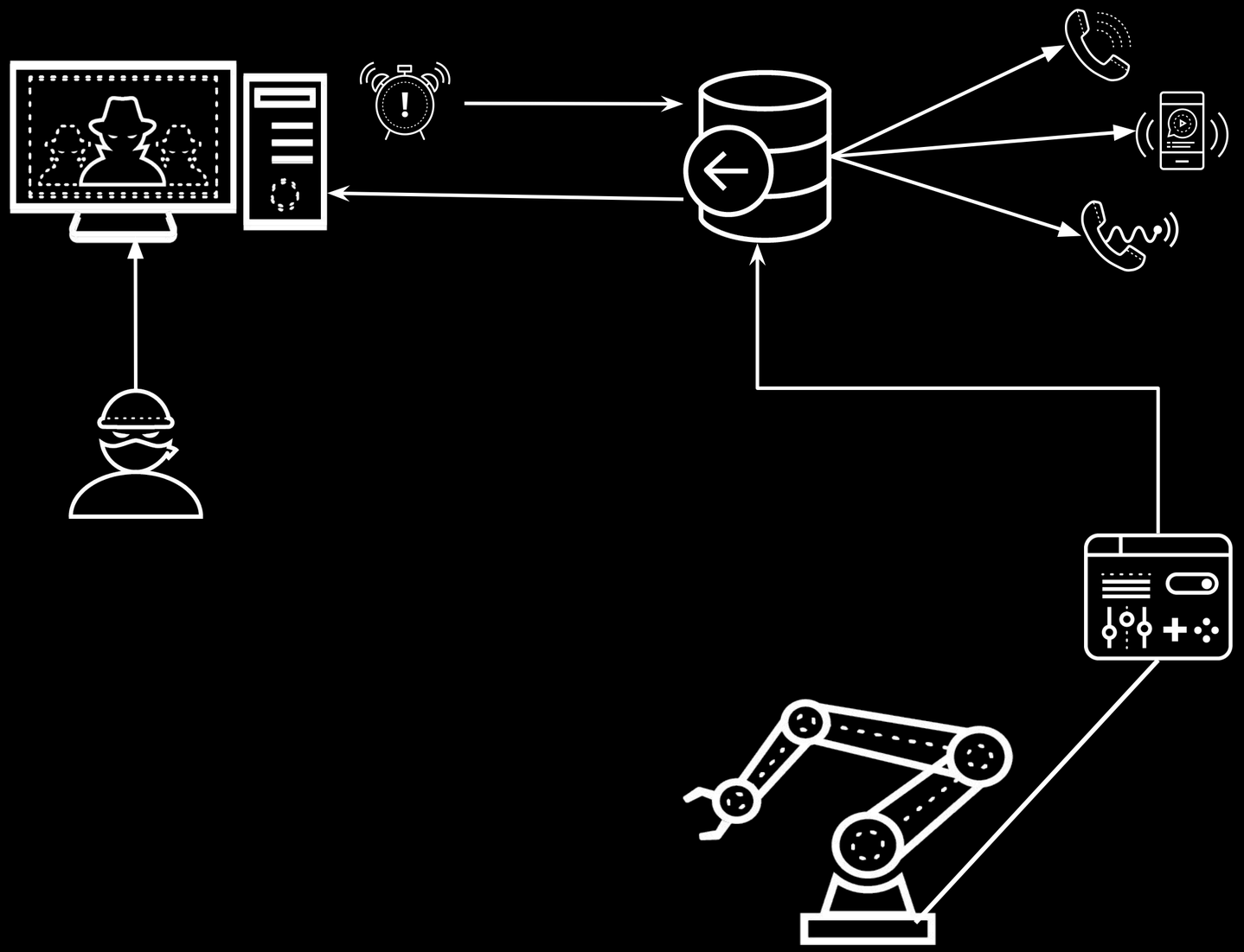
© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Demo:



© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

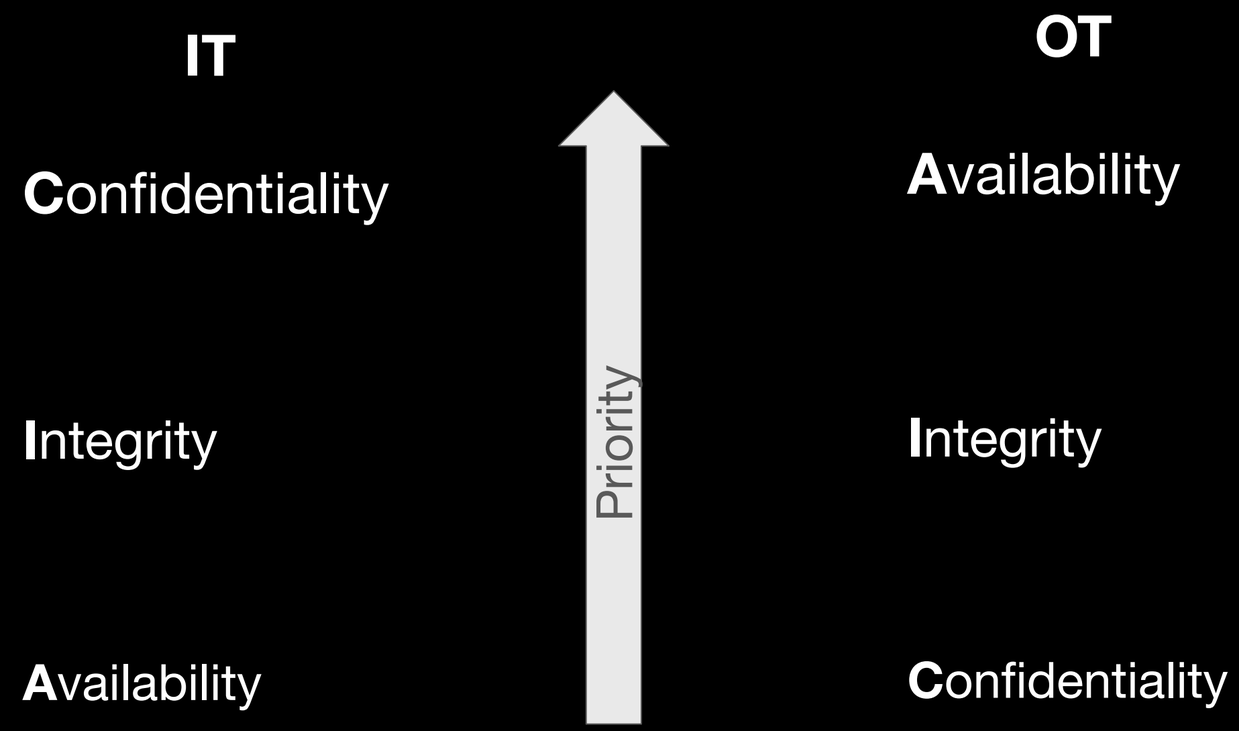
Attack diagram



© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

CIA becomes AIC in OT/ICS infrastructure

- OT has different priorities than IT



© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

ICS-Advisory: ICSA-24-184-03

ICS ADVISORY

ICONICS and Mitsubishi Electric Products

Release Date: July 02, 2024 **Alert Code:** ICSA-24-184-03

RELATED TOPICS: INDUSTRIAL CONTROL SYSTEM VULNERABILITIES, INDUSTRIAL CONTROL SYSTEMS

View CSAF

1. EXECUTIVE SUMMARY

- **CVSS v3 7.0**
- **ATTENTION**: Exploitable remotely
- **Vendor**: ICONICS, Mitsubishi Electric
- **Equipment**: ICONICS Product Suite
- **Vulnerabilities**: Allocation of Resources Without Limits or Throttling, Improper Neutralization, Uncontrolled Search Path Element, Improper Authentication, Unsafe Reflection

2. RISK EVALUATION

Successful exploitation of these vulnerabilities could result in denial of service, improper privilege management, or potentially remote code execution.

3. TECHNICAL DETAILS

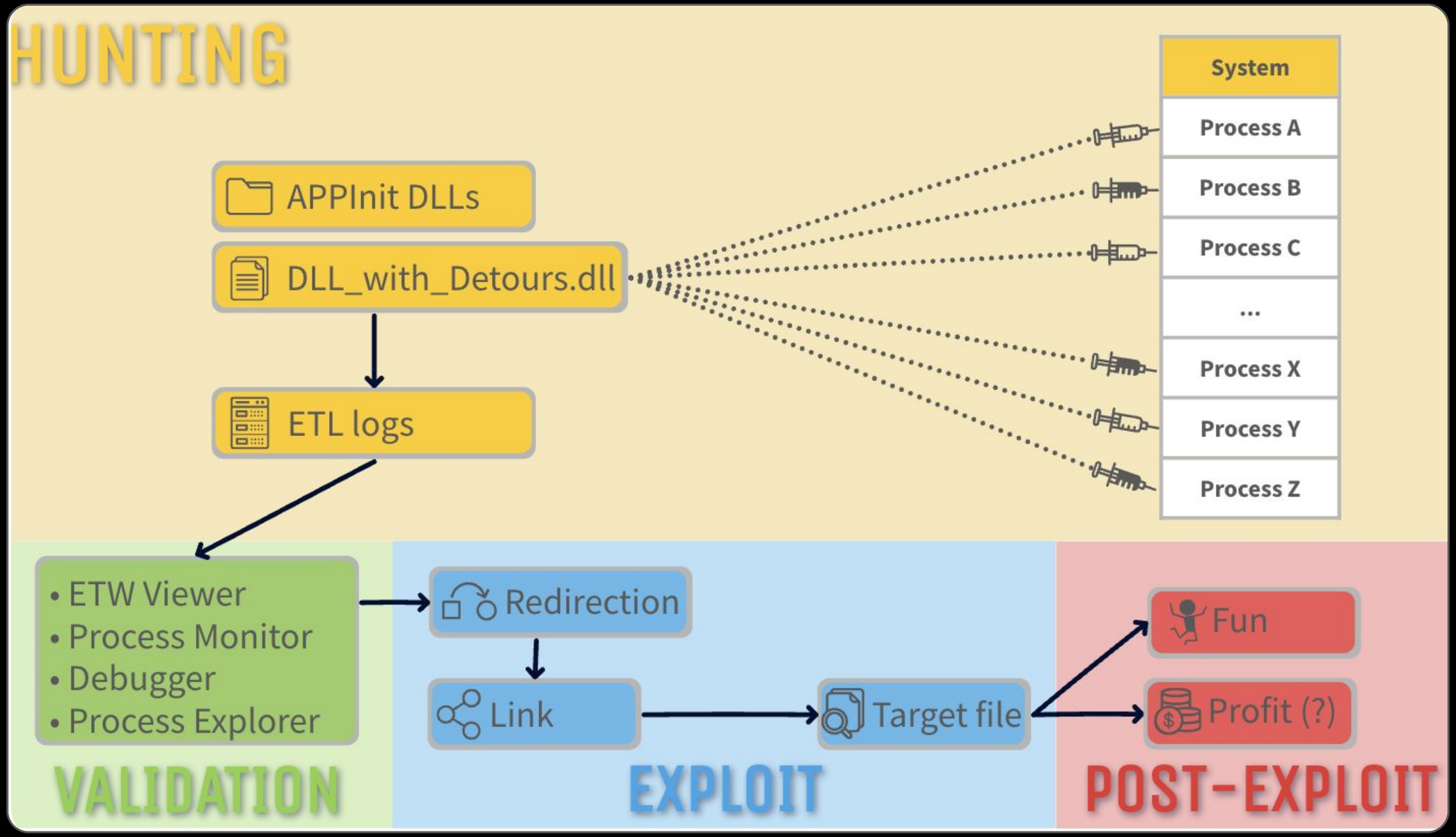
3.1 AFFECTED PRODUCTS

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Next publications

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Overall diagram



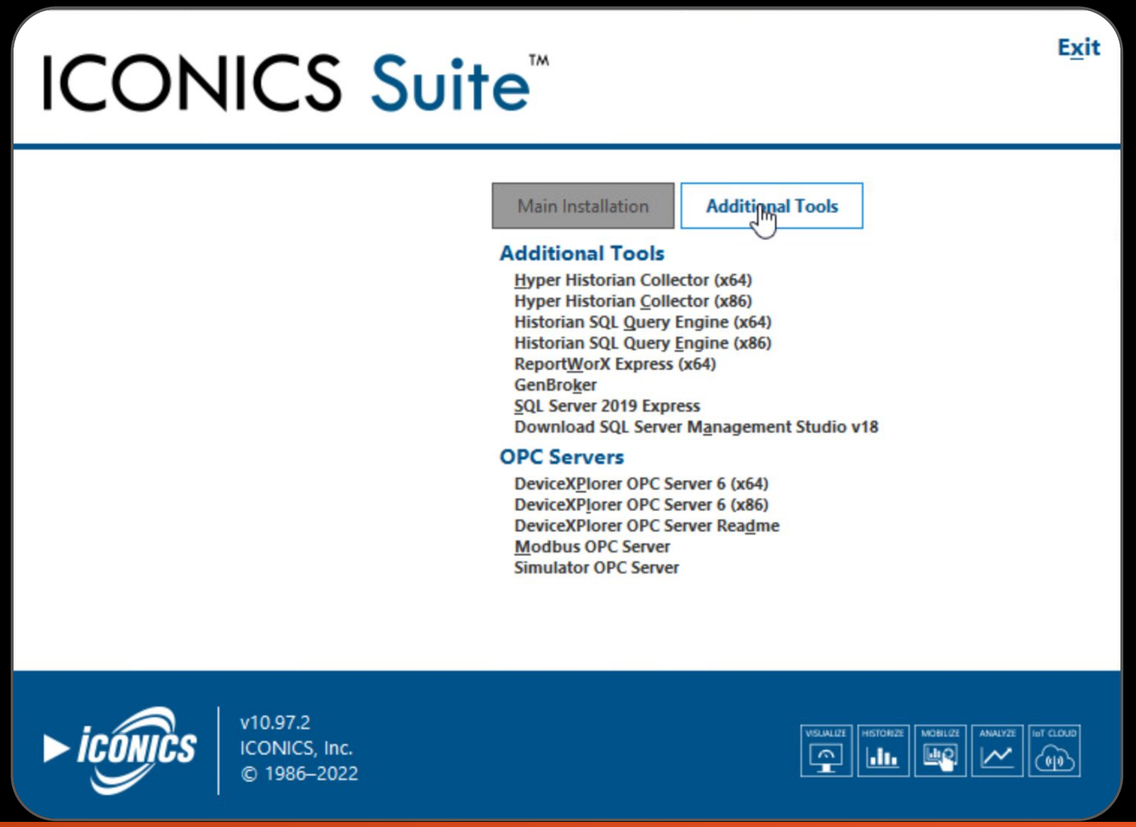
© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Caveats and limitations

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Caveats and limitations

- Every time gets harder and harder to exploit vulnerabilities using file redirection method. However, keep in mind that this was the method I used to exploit this specific vulnerability. The exploitation method will vary per case.
- According to their response, GenBroker32 should not be installed on top of Iconics Suite although that is never warned during the installation process.



© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Q&A

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.

Thanks!

@asher_davila

https://cronop-io.github.io/

© 2024 Palo Alto Networks, Inc. All rights reserved. Proprietary and confidential information.