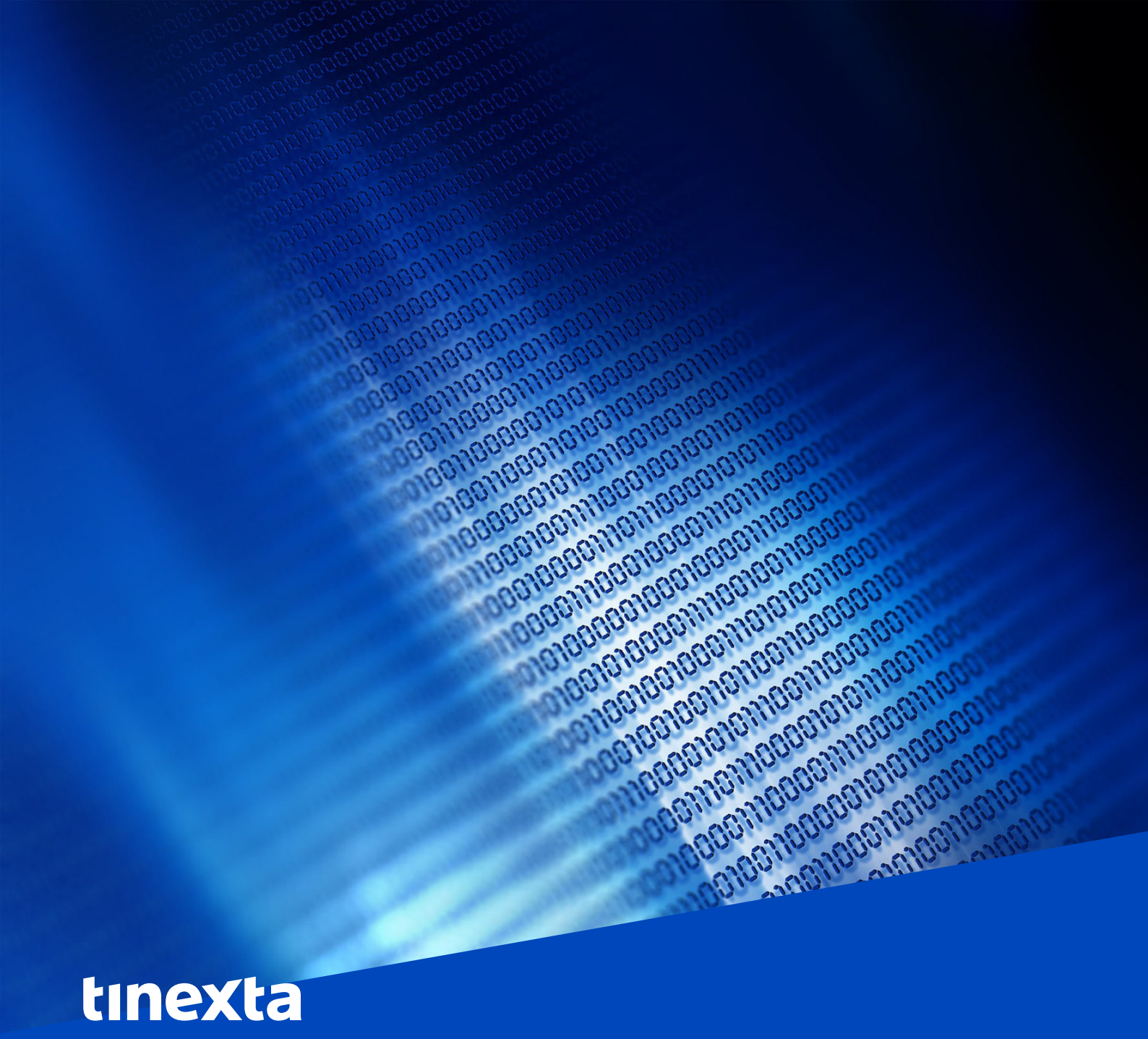


tinexta defence

JetBrains: CVE-2025-23385

Vulnerability Analysis Report

#TinextaDefenceBusiness

Summary

Our Malware Lab

Tinexta Defence Malware Lab daily performs dissection of malware with the aim of timely understanding the technological evolutions of attacks, consolidating the knowledge of necessary to make more effective and faster the process of incidents responding, contributing to spreading information about emerging threats into the expert's community and among its clients.

Malware Lab analysts are continuously engaged in searching and experimenting new analysis tools, for increasing accuracy and scope of action with regard to the proliferation of new evasion and anti-analysis techniques adopted by malware.

The Malware Lab is also committed to the development of proprietary tools for malware analysis and supporting the management and response of incidents.

Besides malware analysis, Malware Lab ideated and implemented an automatic process of extraction of **Indicators of Compromise (IOC)** that is daily run on dozens of new malwares, intercepted in the wide for populating our Knowledge Base.

Corrado Aaron Visaggio

Group Chief Scientist Officer
& Malware Lab Director

a.visaggio@defencetech.it

Executive Summary

Our Malware Lab team discovered two Local Privilege Escalation (LPE) vulnerabilities on JetBrains ETW Host Service.

ETW Host Service is a component used for profiling third-party applications distributed as part of the JetBrains software development tools suites.

Specifically, the affected binary is `JetBrains.Etw.Collector.Host.exe`, which is part of the JetBrains `dotTrace`[1] performance profiler tool and potentially other debugging components. Such a tool could be automatically installed along with the JetBrains unified installer.

We have privately disclosed these issues to JetBrains through their feedback platform; as a result, they promptly fixed them. Both the two vulnerabilities have been assigned with the CVE-2025-23385[2] identifier.

[1] https://www.jetbrains.com/profiler/

[2] https://www.cve.org/CVERecord?id=CVE-2025-23385

Inter-Process Communication

The ETW Host Service is a process that runs with `SYSTEM` privileges. Its purpose is to set up diagnostic tracing sessions without requiring users to have administrative rights.

This is a common scenario in enterprise software development, since employees usually do not have elevated privileges on endpoints.

In this deployment, the user interacts with a GUI frontend which communicates with the service via a named pipe called `\\.\pipe\jetbrains.profiler.etw.host.v16`. Although this is a common pattern in Windows, it is crucial to ensure that the service cannot be abused to execute arbitrary code with elevated privileges.

From a quick analysis of the service and the software installation directory, we discovered the `JetBrains.Process.Elevator.dll` library. This can be used to directly communicate with the service without the need to reverse-engineer the full protocol. We also found out that the exact purpose of the service is to run certain JetBrains-signed executables as `SYSTEM` when needed.

Consumers of this API can call the `CreateProcessElevator` function to allocate a "Process Elevator" instance and then invoke `ElevateProcess` to send a request for starting a new process with arbitrary arguments.

The service has been designed with safeguards to prevent abuse. In particular, it checks the metadata and signature of the target executable file to allow only a subset of files signed by JetBrains, as well as to properly defend from Time-of-Check to Time-of-Use (TOCTOU)[3] attacks.

However, we discovered two vulnerabilities which can be exploited to bypass these checks and execute code as `SYSTEM`.

[3] https://cwe.mitre.org/data/definitions/367.html

Vulnerability 1: Arbitrary library loading

When the service receives an elevation request, it attempts to load a resource named `CompatibilityVersion` from the target PE file as part of the metadata initialization. The pseudocode for this check is as follows:

```
target = LoadLibraryW(requested_file);
if ( !target )
  goto library_failed;
rsesource = FindResourceW(target, L"CompatibilityVersion", 0xA);
if (resource)
// Continue loading the resource...
```

However, this is dangerous since `LoadLibraryW` will load the target PE file, and in case of a DLL, it will execute the `DllMain` function. This can be abused to execute arbitrary code within the context of the service.

This aspect is particularly unfortunate because it happens before the signature gets validated; as a result, any PE file will work at this stage.

To exploit this vulnerability, we crafted a malicious DLL that spawns a `cmd` process as soon as it is loaded. The following code snippet demonstrates how to achieve the mentioned goal:

```
// build with
//   cl /LD example.c

#include <Windows.h>
#include <stdio.h>
```

```
BOOL APIENTRY DllMain(HMODULE hModule, DWORD
ul_reason_for_call, LPVOID lpReserved)
{
    if (ul_reason_for_call == DLL_PROCESS_ATTACH)
        system("cmd /c whoami > C:\\exploit_success.txt");

    return TRUE;
}
```

At this point, we could simply use the `ElevateProcess` API to pass the path of our malicious DLL to the service.

While the elevation request itself will fail due to the various checks, the malicious DLL will be loaded into the service and the code in `DllMain` will be executed automatically.

Mitigation

To properly inspect the resources of a PE file without allowing for code execution, Microsoft recommends the use of `LoadLibraryExW` with the `LOAD_LIBRARY_AS_DATAFILE` flag[4].

[4] https://learn.microsoft.com/en-us/windows/win32/api/libloaderapi/nf-libloaderapi-loadlibraryexa

Vulnerability 2: Signature check bypass

During the final phase of the elevation request, the service will perform the following actions:

1. The target PE file is copied to a randomly generated directory within `C:\Windows\Temp`, effectively preventing TOCTOU attacks;
2. The OEM metadata, such as the `ProductName` field, is checked against a hardcoded whitelist of JetBrains products;
3. The signature of the file is checked using the `WinVerifyTrust`[5] system API;
4. The signer certificate is checked to ensure that the Common Name (CN) contains the string `JetBrains s.r.o.`;
5. Finally, the binary is executed with elevated privileges.

While this approach appears solid, PE files can contain multiple signatures, and the service will check just the name of the first signer – assuming that it is valid.

We can abuse this aspect by first signing a fake self-signed certificate under the name `JetBrains s.r.o.` and then signing the binary a second time with any valid certificate, including development certificates installed on the target machine itself.

This technique is most effective when using a leaked code-signing certificate that has not been revoked yet. Such kind of certificates are still commonly found in malware campaigns.

[5] https://learn.microsoft.com/en-us/windows/win32/api/wintrust/nf-wintrust-winverifytrust

To produce one such binary, the process consists of the following steps:

1. Produce a self-signed certificate with the JetBrains CN;
2. Use it to sign the target executable file: `signtool.exe sign /v /fd sha256 /f fake_cert.pfx malicious_payload.exe`;
3. Use the signtool's /as flag to append a second signature to the executable: `signtool.exe sign /as /v /fd sha256 /f leaked_cert.pfx /p leaked_password malicious_payload.exe`.

This configuration bypasses the `WinVerifyTrust` validation because the second signature is considered valid; as a result, it manages to bypass the certificate name check because the first signature has the correct CN, although it is not valid.

Mitigation

JetBrains has fixed this vulnerability by ensuring that the target executable has only one valid signature.

Conclusion

JetBrains products are widely used by developers and DevOps teams for writing, testing and deploying code, and are frequently installed on systems with access to source code repositories and critical build pipelines.

The two vulnerabilities identified in JetBrains software allow a potential threat actor with a non-administrative account to escalate privileges and potentially gain full control of the system. These issues could be exploited as part of a post-compromise phase of the attack chain, significantly increasing the impact of the initial breach.

For instance, an attacker who compromises a developer's low-privileged account could use these vulnerabilities to access or tamper with build pipelines, inject malicious code or exfiltrate credentials, thus compromising the integrity of both the development and deployment environments.

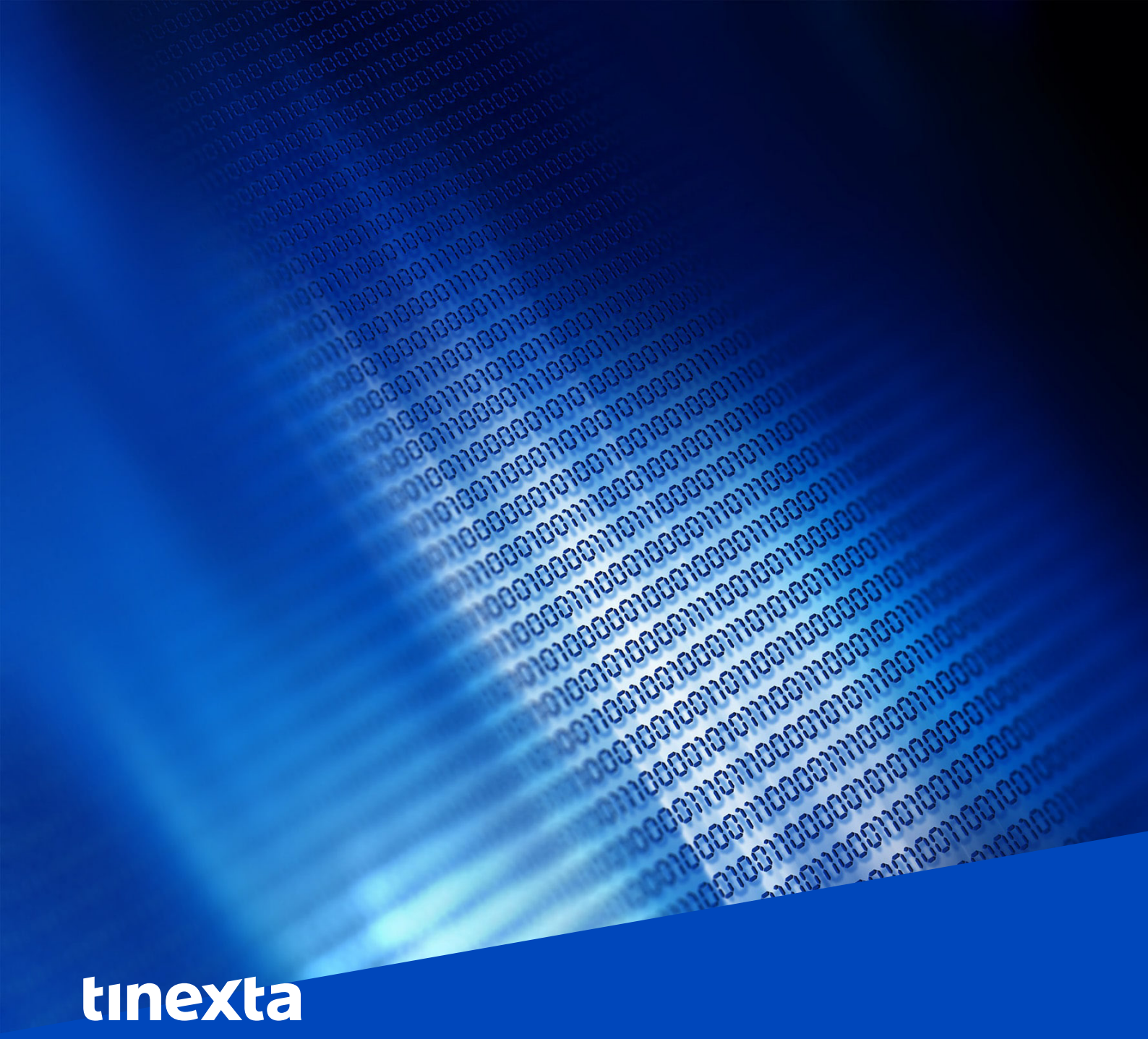
Given the fundamental role that the JetBrains tools constitute in modern software development workflows, consistent patching is always strongly recommended in order to mitigate the likelihood of exploitation.

Disclosure timeline and vendor response

- **January 10th, 2025: Initial report to JetBrains.**
- **January 10th, 2025: JetBrains acknowledged the report and started investigating.**
- **February 20th, 2025: JetBrains released a fix and closed the report.**

According to the vendor, the following software versions include the needed fix: `2024.1.7`, `2024.2.8` and `2024.3.4`.

In particular, the fixed version of ETW Host Service shows version number `16.43`.



tinexta
defence

Defence Tech | Next |
Donexit | Foramil | Innodesi

Via Giacomo Peroni, 452 – 00131 Roma
tel. 06.45752720 – info@defencetech.it
www.tinextadefence.it

#TinextaDefenceBusiness