


Blog

DeepSurface Security Advisory: Local Privilege Escalation in Erlang on Windows (CVE-2021-29221)

 DeepSurface Admin(<https://deepsurface.com/author/it/>)

 April 4, 2021(<https://deepsurface.com/2021/04/04/>)

Overview

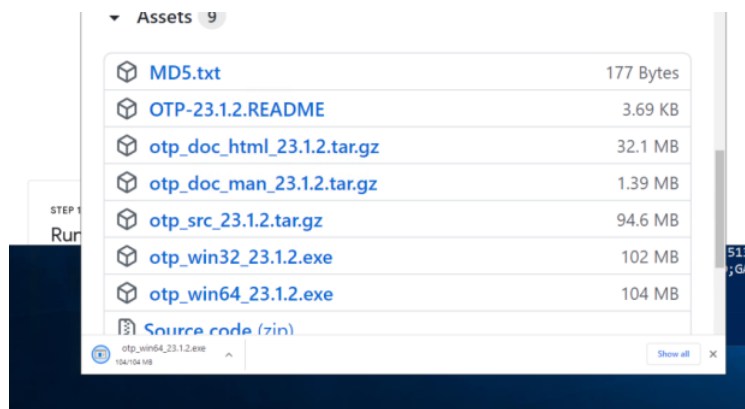
Erlang (<http://erlang.org/faq/introduction.html>) is a popular general-purpose programming language and runtime environment, with support for concurrency commonly found on many distributed systems. When distributed on Windows machines, the Erlang emulator can also be run as a service with the **erlsrv.exe** command. This seems to be commonly used with popular software, such as CouchDB.

These services, by default, run as Local System, and are thus an interesting target for privilege escalation attacks. Improperly configured services are a common vector for privilege escalation in Windows installations. For instance, if the directory storing a service executable has improperly configured permissions, it's often possible for an attacker to drop in a malicious DLL in the same directory which will then be loaded by the application. For more information on service permissions, DLL sideloading/hijacking, and related issues see our previous post (<https://deepsurface.com/windows-service-permissions-and-dll-sideloading/>) on the topic.

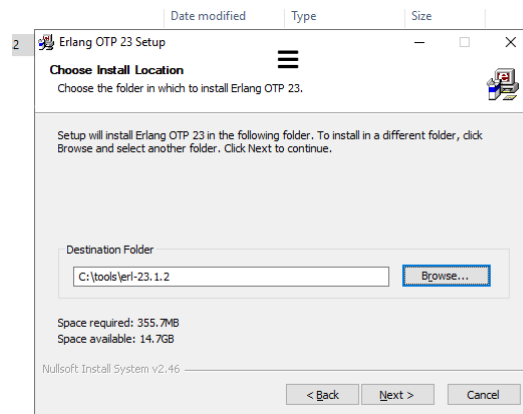
This is a writeup for CVE-2021-29221 (<https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-29221>), a DLL hijacking attack against Erlang which allows an unprivileged local attacker to escalate to Local System under certain conditions. Erlang/OTP prior to version 23.2.3 is affected by this vulnerability.

Exploitation

First, we downloaded the latest version of Erlang (at the time of discovery) from here: <https://github.com/erlang/otp/releases/tag/OTP-23.1.2> (<https://github.com/erlang/otp/releases/tag/OTP-23.1.2>). We tested with the 64 bit installer **otp_win64_23.1.2.exe** but the below steps should be reproducible across any vulnerable version.



Throughout the installation process, we use default settings except for the installation directory, which we change to C:\tools to demonstrate the permission misconfiguration.



Expanded Vulnerability Scanner Support

When DeepSurface launched last year, we could ingest vulnerability data from the Tenable line of vulnerability scanners. Today we're pleased to announce that we've expanded support to include Rapid7's InsightVM and Qualys VMDR.

Note the improperly configured permissions in the Erlang installation folder, which are inherited from the drive root. This gives any local user the ability to create arbitrary files in the installation directory, which we can then leverage in a DLL hijacking attack.

In order to weaponize this vulnerability, we need software to use the `erlsrv.exe`, or Erlang emulator service. To demonstrate this, we can simply set up a dummy service with `.erlsrv.exe add TestService`. This will create a service named "TestService" that calls into `erlang`, allowing us to then perform a DLL hijacking attack.

We can also easily confirm that the service is registered with the vulnerable Erlang installation.

In order to execute a DLL hijacking attack, we need to determine which DLLs the process loads. To do this, we can use Process Monitor (<https://docs.microsoft.com/en-us/sysinternals/downloads/procmon>) from Sysinternals.

Note that in this case, the DLL hijacking occurs against the **`erl.exe`** process, which is spawned from **`erlsrv.exe`**.

Opening up procmon, we add filters on the process name, result, and path. Note that we also exclude all paths that do not contain **`".dll"`** (which is not shown in the screenshot below).

We note that there are two potential targets for DLL hijacking, VCRUNTIME140.dll and IPHLPAPI.dll. In this case, we performed the attack against IPHLPAPI.dll.

With most DLL hijacking attacks, all that is required is a **DLLMain** function declaration. However, perhaps due to differences in how erl.exe loads libraries, defining **DLLMain** alone was not sufficient. Our tests suggest that the DLL was being searched for certain symbols, which if not present would cause the library load operation to abort.

In order to get around this additional constraint, we constructed a poisoned DLL with all of the symbols in the targeted DLL. While technically exporting all of the symbols is not required, it allows us to avoid seeing exactly what symbols erl.exe imports.

In order to get the list of DLL exports, we use DLL Export Viewer (https://www.nirsoft.net/utils/dll_export_viewer.html). From here, we pass the exported DLL functions into our DLL hijacking script (<https://github.com/deepsurfacesec/advisories/blob/main/erlang-priv-escalation/build.py>).

This will generate an **output.dll** which we can then rename to **IPHLPAPI.dll**, and drop into the current directory of ersrv.exe, **C:\tools\erl-23.1.2\erts-11.1.2\bin**. When executed successfully, the code in this DLL executes the user account that is being hijacked to **C:\pwned**.

To fully demonstrate the implications of this attack, create a new non-admin user account and log into it. Then drop the poisoned **IPHLPAPI.dll** into **C:\tools\erl-23.1.2\erts-11.1.2\bin**.

To trigger the exploit, either restart the computer or restart the "Test" service as an administrative user.

With the provided payload, you should see **C:\pwned** created, with **nt authority\system** logged as the user account. This demonstrates a successful escalation of privilege, from a non-admin user account to **nt authority\system**.

(https://github.com/erlang/otp/commit/42eed52a08e2fb0e59a7dafccb752dde6e2f9631),
the vendor patch explicitly sets the permissions in the installation directory. In order to
address this vulnerability, users should double-check the permissions on existing erlang
installation directories.
DeepSurface is a predictive vulnerability management platform that contextualizes vulnerabilities
and chains of vulnerabilities within your unique digital infrastructure predicting where an attacker could
cause the most damage to your business.
• Uninstalling the current installation and reinstalling the latest version of Erlang. Note that
the patch was made in version 23.2.3 (<https://github.com/erlang/otp/releases/tag/OTP-23.2.3>).
• Modifying the permissions on the existing installation directory, for example, with the script at
the bottom of our previous blog post (<https://deepsurface.com/windows-service-permissions-and-dll-sideload/>).
About (/About/) Privacy Policy (/Privacy-Policy/) Contact Us (/Contact/)

Vendor Notification

December 16th, 2020	Reported to erlang-security@erlang.org
December 17th, 2020	Erlang security team response
January 5th, 2021	Continued correspondence with Erlang security team
January 20th, 2021	Erlang version 23.2.3 released with fix OTP-17097
April 5th, 2021	CVE-2021-29221 issued by DeepSurface Security
April 5th, 2021	DeepSurface Advisory published