SYNACKTIV
DIGITAL SECURITY

Cross-Site Scripting vulnerabilities in Zend Server < 9.1.3

Security advisory

2018-04-23

Thomas Chauchefoin
Julien Egloff

Vulnerability description

Presentation of *Zend Server / Zend Debugger*

"With Zend Server, development and operation teams are equipped with the software infrastructure, tools, and best practices for productive collaboration and continuous delivery of their mobile and web apps with exceptional performance, reliability, and security."[1]

"The Zend Debugger is the PHP extension which should be installed on your Web server in order to perform optimal remote debugging and profiling using Zend Studio. […] The Zend Debugger comes bundled with both Zend Core and Zend Platform and does not need to be installed separately."[2]

The issue

Synacktiv discovered that *Zend Debugger* does not correctly encode user-controlled data before using it in error messages. Thus, an attacker could abuse it to craft arbitrary HTML elements in server's response, leading to a possibility of *Cross-Site Scripting* (XSS) attacks. Technical details are documented in the *Attack scenario* part of this document.

The only requirement is that the *Zend Debugger* module has to be loaded and enabled. It does not require any prior authentication and will work for any domain served by the web server where this module is present.

It should be noted that this vulnerability does not appear to be present in *production* mode and that development instances of *Zend Server* are not supposed to be exposed to the Internet.

Affected versions

The following versions are known to be affected:

- *Zend Debugger,* bundled with *Zend Server 9.0.1* up to 9.1.3 and up to 8.5.9.

Official mitigation

Zend Server 8.5.9 and 9.1.3 bundle a fixed version of *Zend Debugger.*

Timeline

Date	Action
2018-02-02	Advisory sent to Zend.
2018-04-17	Zend publishes *Zend Server* 8.5.9 and 9.3.1[3].
2018-04-19	MITRE assigns CVE-2018-10230 to the vulnerability.
2018-04-23	Advisory published.

1 http://www.zend.com/en/products/zend_server

2 https://www.zend.com/topics/Zend-Debugger-Installation-Guide.pdf

3 http://forums.zend.com/viewtopic.php?f=8&t=134223

Technical description and proof-of-concept

Initial vulnerability discovery

In *ZendDebugger.so*, the function at 0x00015D10 is responsible of handling user's connection and retrieving the actual configuration directives (from URL parameters or configuration file). When asking the debugger to connect back to a host, *Zend Debugger* will try to resolve its hostname (if applicable) and then connect to it. If it fails to resolve the domain name or connect to the host, the value extracted from the URL parameter `debug_host` will be inserted into server's response without a proper encoding:

```
  while ( 1 )
  {
    [...]
    if ( zend_debugger_lookup_hostname(host_2, &addr_1.sa_data[2]) == -1 )
    {
      zend_realloc_printf(v41, "Failed to resolve host '%s'. ", host_2, v41);
      goto SKIP_CONNECT;
    }
    if ( zend_debugger_check_ip_allowed(&addr_1, 16LL, host_2, v41) )
      goto SKIP_CONNECT;
    if ( !connect(v4, &addr_1, 0x10u) )
      break;
    zend_realloc_printf(v41, "Failed to connect to host '%s'. ", host_2, v41);
```

At 0x0001E3E0, some code is responsible for matching denied / allowed hosts with the one provided in `debug_host`, as well as returning an error message if it is not allowed:

```
    if ( allowed_hosts )
    {
      if ( *allowed_hosts )
      {
LABEL_24:
        if ( zend_match_host_masks(host, allowed_hosts, ',') == -1 )
        {
          zend_realloc_printf(
            out_buffer,
            "Host '%s' is not allowed to open debug sessions - please configure
zend_debugger.allow_hosts in the ini file.",
            host);
          return 0xFFFFFFFFLL;
        }
      }
    }
```

Once again, the user-controlled value is not properly encoded before being inserted into the response. All the other error messages formatted using `debug_host` are affected and must be fixed.

Proof of concept

On any host where the module is installed and enabled, an attacker can trigger this vulnerability by adding his payload in the parameter `debug_host`, like in the following example:

```
index.php?debug_host=synacktiv.ninja<script>alert(document.domain)</script>&start_debug=1
```

Impact

A successful exploitation allows an unauthenticated attacker to craft hyperlinks that, once accessed by a victim, would execute arbitrary *JavaScript* code in his browser in the context of the website. Thus, if this victim is currently authenticated to this website (for instance via session cookies), the attacker would be able to perform actions on this website on victim's behalf. If the victim is not authenticated, this could still allow performing phishing attacks to steal credentials.