



HORIZON3.ai
TRUST BUT VERIFY



ManageEngine CVE- 2022-47966 Technical Deep Dive



Introduction

On January 10, 2023, ManageEngine released a security advisory for [CVE-2022-47966](#) (discovered by [Khoadha of Viettel Cyber Security](#)) affecting a wide range of products. The vulnerability allows an attacker to gain remote code execution by issuing a HTTP POST request containing a malicious SAML response. This vulnerability is a result of using an outdated version of [Apache Santuario](#) for XML signature validation.

Patch Analysis

We started our initial research by examining the differences between ServiceDesk Plus version 14003 and version 14004. By default, Service Desk is installed into `C:\Program Files\ManageEngine\ServiceDesk`. We installed both versions and extracted the jar files for comparison.

While there are many jar files that have been updated, we notice that there was a single jar file that has been completely changed. `libxmlsec` from Apache Santuario was updated from 1.4.1 to 2.2.3. Version 1.4.1 is over a decade old.



HORIZON3.ai
TRUST BUT VERIFY

struts-elt-1.1.jar	204.4 KB	Wed 11 Jan 2023 11:09:50	struts-elt-1.1.jar	204.4 KB	Wed 11 Jan 2023 11:10:30
struts-elt-1.1.1.jar	39.7 KB	Wed 11 Jan 2023 11:09:50	struts-elt-1.1.1.jar	39.7 KB	Wed 11 Jan 2023 11:10:30
struts-taglib-1.1.1.jar	231.6 KB	Wed 11 Jan 2023 11:09:50	struts-taglib-1.1.1.jar	231.6 KB	Wed 11 Jan 2023 11:10:30
struts-taglib-1.1.1.jar	119.6 KB	Wed 11 Jan 2023 11:09:50	struts-taglib-1.1.1.jar	119.6 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	499.4 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	499.4 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	162.8 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	162.8 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	59.7 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	59.7 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	437.4 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	437.4 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	100.1 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	100.1 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	132.7 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	132.7 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	8.6 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	8.6 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	2.2 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	2.2 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	2.8 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	2.8 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	162.8 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	162.8 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	1.5 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	1.5 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	80.8 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	80.8 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	224.4 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	224.4 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	2.6 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	2.6 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	414.1 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	414.1 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	54.7 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	54.7 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	127.7 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	127.7 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	41.7 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	41.7 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	100.1 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	100.1 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	319.6 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	319.6 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	102.4 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	102.4 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	102.4 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	102.4 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	1.5 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	1.5 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	276.1 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	276.1 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	54.0 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	54.0 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	16.8 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	16.8 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	75.3 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	75.3 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	36.7 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	36.7 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	45.1 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	45.1 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	71.9 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	71.9 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	2.1 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	2.1 KB	Wed 11 Jan 2023 11:10:30
struts-core.jar	102.2 KB	Wed 11 Jan 2023 11:09:50	struts-core.jar	102.2 KB	Wed 11 Jan 2023 11:10:30

Jar differences

That is a large version jump, but if we start with the [1.4.2 release notes](#) we find an interesting change:

- Switch order of XML Signature validation steps. See Issue 44629.

Issue 44629 can be found [here](#). It describes switching the order of XML signature validation steps and the security implications.

XML Signature Validation

XML signature validation is a complex beast, but it can be simplified down to the the following two steps:

- Reference Validation – validate that each <Reference> element within the <SignedInfo> element has a valid digest value.
- Signature Validation – cryptographically validate the <SignedInfo> element. This assures that the <SignedInfo> element has not been tampered with.

While the official XML signature validation spec lists reference validation followed by signature validation, these two steps can be performed in any order. Since the reference validation step can involve processing attacker controlled XML Transforms, one should always perform the signature

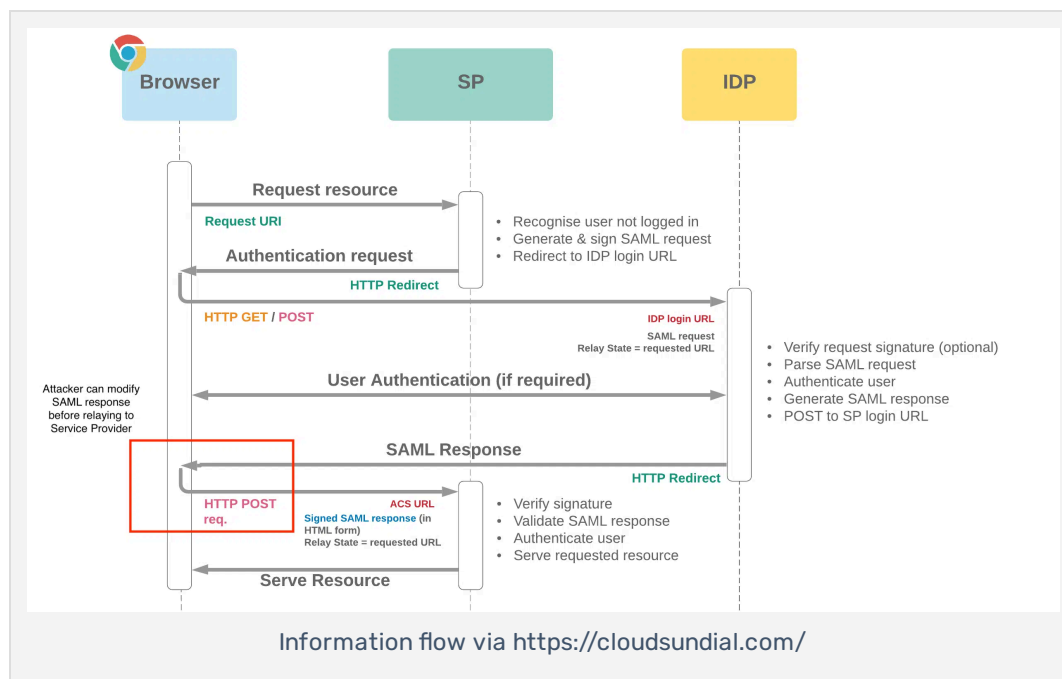


HORIZON3.ai
TRUST BUT VERIFY

SAML Information Flow Refresher

Applications that support single sign-on typically use an authorization solution like SAML. When a user logs into a remote service, that service forwards the authentication request to the SAML Identity Provider. The SAML Identity Provider will then validate that the user credentials are correct and that they are authorized to access the specified service. The Identity Provider then returns a response to the client which is forwarded to the Service Provider.

The information flow of a login request via SAML can be seen below. One of the critical pieces is understanding that the information flow uses the client's browser to relay all information between the Service Provider (SP) and the Identity Provider (IDP). In this attack, we send a request containing malicious SAML XML directly to the service provider's Assertion Consumer (ACS) URL.





Vulnerability Ingredient 1: SAML Validation Order

Understanding that SAML information flow allows an attacker to introduce or modify the SAML data in transit, it should now be clear why the Apache Santuario update to now perform signature validation to occur before reference validation was so important. This vulnerability will abuse the verification order as the first step in exploitation. See below for the diff between v1.4.1 and v1.4.2.

```
@@ -579,11 +593,7 @@ private Element signatureValueElement;
// If _followManifestsDuringValidation is true it will do the same for
// References inside a Manifest.
try {
-    SignedInfo si=this.getSignedInfo();
-    if (!si.verify(this._followManifestsDuringValidation)) {
-        return false;
-    }
+
+    SignedInfo si=this.getSignedInfo();
+    //create a SignatureAlgorithms from the SignatureMethod inside
+    //SignedInfo. This is used to validate the signature.
+    SignatureAlgorithm sa =si.getSignatureAlgorithm();
@@ -600,20 +610,22 @@ private Element signatureValueElement;
OutputStream bos=new UnsyncBufferedOutputStream(so);
si.signInOctectStream(bos);
try {
-    bos.close();
-    } catch (IOException e) {
-        //Impossible
-    }
+
+    bos.close();
+    } catch (IOException e) {
+        //Impossible
+    }

//retrieve the byte[] from the stored signature
byte sigBytes[] = this.getSignatureValue();

//Have SignatureAlgorithm sign the input bytes and compare them to the
//bytes that were stored in the signature.
-    boolean verify = sa.verify(sigBytes);
+    if (!sa.verify(sigBytes)) {
+        log.warn("Signature verification failed.");
+        return false;
+    }

-    return verify;
+    return si.verify(this._followManifestsDuringValidation);
} catch (XMLSecurityException ex) {
    throw new XMLSignatureException("empty", ex);
}
```



with the call to `si.verify()`. In v1.4.2, the call to `si.verify()` was moved to the end of the function after the signature verification in `sa.verify(sigBytes)`.

Vulnerability Ingredient 2: XSLT Injection

Furthermore, each `<Reference>` element can contain a `<Transform>` element responsible for describing how to modify an element before calculating its digest. Transforms allow for arbitrarily complex operations through the use of [XSL Transformations](#) (XSLT).

These transforms are executed in

`src/org/apache/xml/security/signature/Reference.java`
which is eventually called from `si.verify()` from above.

```
private XMLSignatureInput getContentsAfterTransformation(XMLSignatureInput input, OutputStream os)
    throws XMLSignatureException {

    try {
        Transforms transforms = this.getTransforms();
        XMLSignatureInput output = null;

        if (transforms != null) {
            output = transforms.performTransforms(input, os);
            this._transformsOutput = output; //new XMLSignatureInput(output.getBytes());

            //this._transformsOutput.setSourceURI(output.getSourceURI());
        } else {
            output = input;
        }

        return output;
    } catch (ResourceResolverException ex) {
        throw new XMLSignatureException("empty", ex);
    } catch (CanonicalizationException ex) {
        throw new XMLSignatureException("empty", ex);
    } catch (InvalidCanonicalizerException ex) {
        throw new XMLSignatureException("empty", ex);
    } catch (TransformationException ex) {
        throw new XMLSignatureException("empty", ex);
    } catch (XMLSecurityException ex) {
        throw new XMLSignatureException("empty", ex);
    }
}
```



environment, it is capable of executing arbitrary Java code. We can supply the following snippet to execute an arbitrary system command:

```
<ds:Transform Algorithm="http://www.w3.org/TR/1999/REC-
  <xsl:stylesheet version="1.0" xmlns:xsl="http://www
    <xsl:template match="/">
      <xsl:variable name="rtobject" select="rt:ge
      <xsl:variable name="process" select="rt:exe
      <xsl:variable name="processString" select="
      <xsl:value-of select="$processString"/>
    </xsl:template>
  </xsl:stylesheet>
</ds:Transform>
```

Abusing the order of SAML validation in Apache Santuario v1.4.1 and Java's XSLT library providing access to run arbitrary Java classes, we can exploit this vulnerability in ManageEngine products to gain remote code execution.

SAML SSO Configuration

[Security Assertion Markup Language \(SAML\)](#) is a specification for sharing authentication and authorization information between an application or service provider and an identity provider. SAML with single sign on allows users to not have to worry about maintaining credentials for all of the apps they use and it gives IT administrators a centralized location for user management.

SAML uses XML signature verification to ensure the secure transfer of messages passed between service providers and identity providers.

We can enable SAML SSO by navigating to Admin -> Users & Permissions -> SAML Single Sign On where we can enter our identity provider information. Once properly configured, we will see "Log in with SAML Single Sign On" on the logon page:



HORIZON3.ai

TRUST BUT VERIFY

ServiceDesk Plus

Username

Password

☐ Keep me signed in

Log in

[Forgot Password?](#)

OR

[Log in with SAML Single Sign On](#)

Log in as

Administrator Guest

Service Desk SAML logon

Proof of Concept

Our proof of concept can be found [here](#).

After configuring SAML, the Assertion Consumer URL will now be active at `https://<hostname>:8080/SamlResponseServlet` and we can send our malicious SAML Response.

```
python3 CVE-2022-47966.py --url https://10.0.40.64:8080
```

Since ServiceDesk runs as a service, there is no desktop to display the GUI for `notepad.exe` so we use ProcessExplorer to check the success of the exploit.

java.exe	< 0.01	885,228 K	855,204 K	2556 Zulu Platform x64 Architecture	Azul Systems Inc.
notepad.exe	< 0.01	1,944 K	9,480 K	5416 Notepad	Microsoft Corporation



HORIZON3.ai
TRUST BUT VERIFY

expect this POC to work unmodified on many of the ManageEngine products that share some of their codebase with ServiceDesk Plus or EndpointCentral.

Notably, the AD-related products (AdManager, etc) have additional checks on the SAML responses that must pass. They perform checks to verify that the SAML response looks like it came from the expected identity provider. Our POC has an optional `--issuer` argument to provide information to use for the `<Issuer>` element. Additionally, AD-related products have a different SAML logon endpoint URL that contains a guid. How to determine this information in an automated fashion is left as an exercise for the reader.

```
python3 CVE-2022-47966.py --url https://10.0.40.90:8443
```

Summary

In summary, when Apache Santuario is \leq v1.4.1, the vulnerability is trivially exploitable and made possible via several conditions:

- Reference validation is performed before signature validation, allowing for the execution of malicious XSLT transforms.
- Execution of XSLT transforms allows an attacker to execute arbitrary Java code.

This vulnerability is still exploitable even when Apache Santuario is between v1.4.1 and v2.2.3, which some of the affected ManageEngine products were using at the time, such as Password Manager Pro. The original research, Khoadha, documents further bypasses of validation in their [research](#) and is definitely worth a read.



HORIZON3.ai
TRUST BUT VERIFY

Let our experts walk you through a demonstration of NodeZero, so you can see how to put it to work for your company.

SCHEDULE A DEMO

Contact Us

info@horizon3.ai
press@horizon3.ai
650-445-4457

Follow Us

Subscribe to Community Updates



HORIZON3.ai
TRUST BUT VERIFY

© 2022 All Rights Reserved. | [Privacy Policy](#) | [Support Policy](#) | [Terms of Service](#)