3rd-Party Software Report for splunk-add-on-for-jira-cloud

This document lists the open source software and other third-party components used in splunk-add-on-for-jira-cloud.

Any information contained in this report is subject to the terms of your FOSSA agreement and any applicable export control laws.

Generated
August 11, 2026

Revision
c4ff75d864169804ad4c0f6b0a5359091a612282

Dependencies Summary

This section shows the summary of all packages in splunk-add-on-for-jira-cloud.

Package	Declared License(s)	Discovered License(s)
@splunk/splunk-utils 2.3.4	Apache-2.0, proprietary-license	None
PySocks 1.7.1	BSD-3-Clause	Apache-2.0
certifi 2024.12.14	MPL-2.0	None
charset-normalizer 3.3.2	MIT	None
defusedxml 0.7.1	None	PSF-2.0
deprecation 2.1.0	Apache-2.0	None
idna 3.18	BSD-3-Clause	None
packaging 24.0	Apache-2.0	None
raw-loader ^4.0.2	MIT	None
requests 2.32.5	Apache-2.0	None
solnlib 7.0.0	Apache-2.0	None

sortedcontainers 2.4.0	Apache-2.0	None
splunk-sdk 2.1.0	Apache-2.0	None
splunktaucclib 8.2.0	Apache-2.0	None
underscore 1.13.8	MIT	None
urllib3 1.26.20	MIT	None

Dependencies Details

This section shows the details for each package in splunk-add-on-for-jira-cloud.

@splunk/splunk-utils 2.3.4 npm+@splunk/splunk-utils$2.3.4 • Direct

Description

Library of common Splunk Enterprise utilities

Package Info

Package Author(s)	devinfo@splunk.com, vtsvetkov@splunk.com, webplatform@splunk.com
Package Manager	NPM
Project URL	https://www.npmjs.com/package/@splunk/splunk-utils
Package Download URL	https://registry.npmjs.org/@splunk/splunk-utils/-/splunk-utils-2.3.4.tgz
Dependency Path	@splunk/splunk-utils

Declared License(s)

Apache-2.0

Copyright(s)	None

proprietary-license

Copyright(s)	Splunk Inc.

Discovered License(s)

None

Notice File(s)

None

raw-loader ^4.0.2

npm+raw-loader$^4.0.2 • Direct

Description

A loader for webpack that allows importing files as a String

Package Info

Package Author(s)	bebraw@gmail.com, eric@smarterspam.com, mail@johannesewald.de, michael.ciniawsky@gmail.com, sean.larkin@cuw.edu, sheo13666q@gmail.com, tobias.koppers@googlemail.com, wiens.joshua@gmail.com
Package Manager	NPM
Project URL	https://www.npmjs.com/package/raw-loader
Package Download URL	https://registry.npmjs.org/raw-loader/-/raw-loader-4.0.2.tgz
Dependency Path	raw-loader

Declared License(s)

MIT

Copyright(s)	JS Foundation and other contributors

Discovered License(s)

None

Notice File(s)

None

requests 2.32.5

pip+requests$2.32.5 • Direct

Description

Python HTTP for Humans.

Package Info

Package Author(s)	Kenneth Reitz <me@kennethreitz.org>
Package Manager	Pip
Project URL	https://pypi.org/project/requests/
Package Download URL	https://files.pythonhosted.org/packages/c9/74/b3ff8e6c8446842c3f5c837e9c3dfcfe2018ea6ecef224c710c85ef728f4/requests-2.32.5.tar.gz
Dependency Path	requests

Declared License(s)

Apache-2.0

Copyright(s)	None

Discovered License(s)

None

Notice File(s)

NOTICE

File Path	NOTICE

Notice Text	Requests Copyright 2019 Kenneth Reitz
Copyright(s)	Copyright © 2019 Kenneth Reitz

solnlib 7.0.0

pip+solnlib$7.0.0 • Direct

Description

The Splunk Software Development Kit for Splunk Solutions

Package Info

Package Author(s)	addonfactory@splunk.com
Package Manager	Pip
Project URL	https://github.com/splunk/addonfactory-solutions-library-python
Package Download URL	https://files.pythonhosted.org/packages/25/2c/21ed535f7701af12799eda0980858367f8b57f834513ca5a05475e448ea1/solnlib-7.0.0.tar.gz
Dependency Path	solnlib, splunktaucclib > solnlib

Declared License(s)

Apache-2.0

Copyright(s)	2021 Splunk Inc.

Discovered License(s)

None

Notice File(s)

None

splunk-sdk 2.1.0 pip+splunk-sdk$2.1.0 • Direct

Description

None

Package Info

Package Author(s)	devinfo@splunk.com
Package Manager	Pip
Project URL	http://github.com/splunk/splunk-sdk-python
Package Download URL	https://files.pythonhosted.org/packages/2c/89/31974e00319b750a5a126717d286e8addb683554454fed1ae6c86977cbcf/splunk-sdk-2.1.0.tar.gz
Dependency Path	solnlib > splunk-sdk, splunk-sdk, splunktaucclib > solnlib > splunk-sdk, splunktaucclib > splunk-sdk

Declared License(s)

Apache-2.0

Copyright(s)	2011-2015 Splunk, Inc.

Discovered License(s)

None

Notice File(s)

None

splunktaucclib 8.2.0

pip+splunktaucclib$8.2.0 • Direct

Description

None

Package Info

Package Author(s)	addonfactory@splunk.com
Package Manager	Pip
Project URL	https://pypi.org/project/splunktaucclib/
Package Download URL	https://files.pythonhosted.org/packages/a1/c5/7dbb204b45a8c5dbb999119fbe64ae643e9fa22be8f4f21f63dbc5345862/splunktaucclib-8.2.0.tar.gz
Dependency Path	splunktaucclib

Declared License(s)

Apache-2.0

Copyright(s)	2021 Splunk Inc.

Discovered License(s)

None

Notice File(s)

None

underscore 1.13.8

npm+underscore$1.13.8 • Direct

Description

JavaScript's functional programming helper library.

Package Info

Package Author(s)	dev@juliangonggrijp.com, jashkenas@gmail.com
Package Manager	NPM
Project URL	https://underscorejs.org/
Package Download URL	https://registry.npmjs.org/underscore/-/underscore-1.13.8.tgz
Dependency Path	underscore

Declared License(s)

MIT

Copyright(s)	2009-2022 Jeremy Ashkenas, Julian Gonggrijp, and DocumentCloud and Investigative Reporters & Editors

Discovered License(s)

None

Notice File(s)

None

certifi 2024.12.14

pip+certifi$2024.12.14 • Transitive

Description

Python package for providing Mozilla's CA Bundle.

Package Info

Package Author(s)	me@kennethreitz.com
Package Manager	Pip
Project URL	https://github.com/certifi/python-certifi
Package Download URL	https://files.pythonhosted.org/packages/0f/bd/1d41ee578ce09523c81a15426705dd20969f5abf006d1afe8aeff0dd776a/certifi-2024.12.14.tar.gz
Dependency Path	requests > certifi

Declared License(s)

MPL-2.0

Copyright(s)	None

Discovered License(s)

None

Notice File(s)

None

charset-normalizer 3.3.2

pip+charset-normalizer$3.3.2 • Transitive

Description

The Real First Universal Charset Detector. Open, modern and actively maintained alternative to Chardet.

Package Info

Package Author(s)	"Ahmed R. TAHRI" <tahri.ahmed@proton.me>
Package Manager	Pip
Project URL	https://pypi.org/project/charset-normalizer/
Package Download URL	https://files.pythonhosted.org/packages/63/09/c1bc53dab74b1816a00d8d030de5bf98f724c52c1635e07681d312f20be8/charset-normalizer-3.3.2.tar.gz
Dependency Path	requests > charset-normalizer

Declared License(s)

MIT

Copyright(s)	2019 TAHRI Ahmed R.

Discovered License(s)

None

Notice File(s)

None

defusedxml 0.7.1

pip+defusedxml$0.7.1 • Transitive

Description

==========
defusedxml -- defusing XML bombs and other exploits
========== .. image:: https://img.shields.io/pypi/v/defusedxml.svg :target: https://pypi.org/project/defusedxml/ :alt: Latest Version .. image:: https://img.shields.io/pypi/pyversions/defusedxml.svg :target: https://pypi.org/project/defusedxml/ :alt: Supported Python versions .. image:: https://travis-ci.org/tiran/defusedxml.svg?branch=master :target: https://travis-ci.org/tiran/defusedxml :alt: Travis CI .. image:: https://codecov.io/github/tiran/defusedxml/coverage.svg?

branch=master :target: https://codecov.io/github/tiran/defusedxml?branch=master :alt: codecov .. image:: https://img.shields.io/pypi/dm/defusedxml.svg :target: https://pypistats.org/packages/defusedxml :alt: PyPI downloads .. image:: https://img.shields.io/badge/code%20style-black-000000.svg :target: https://github.com/psf/black :alt: Code style: black .. "It's just XML, what could probably go wrong?" Christian Heimes <christian@python.org> Synopsis ======== The results of an attack on a vulnerable XML library can be fairly dramatic. With just a few hundred **Bytes** of XML data an attacker can occupy several **Gigabytes** of memory within **seconds**. An attacker can also keep CPUs busy for a long time with a small to medium size request. Under some circumstances it is even possible to access local files on your server, to circumvent a firewall, or to abuse services to rebound attacks to third parties. The attacks use and abuse less common features of XML and its parsers. The majority of developers are unacquainted with features such as processing instructions and entity expansions that XML inherited from SGML. At best they know about ≤!DOCTYPE>`from experience with HTML but they are not aware that a document type definition (DTD) can generate an HTTP request or load a file from the file system. None of the issues is new. They have been known for a long time. Billion laughs was first reported in 2003. Nevertheless some XML libraries and applications are still vulnerable and even heavy users of XML are surprised by these features. It's hard to say whom to blame for the situation. It's too short sighted to shift all blame on XML parsers and XML libraries for using insecure default settings. After all they properly implement XML specifications. Application developers must not rely that a library is always configured for security and potential harmful data by default. .. contents:: Table of Contents :depth: 2 Attack vectors ============== billion laughs / exponential entity expansion ---------- The Billion Laughs`_ attack -- also known as exponential entity expansion -- uses multiple levels of nested entities. The original example uses 9 levels of 10 expansions in each level to expand the string `lol`to a string of 3 * 10 :sup:9`bytes, hence the name "billion laughs". The resulting string occupies 3 GB (2.79 GiB) of memory; intermediate strings require additional memory. Because most parsers don't cache the intermediate step for every expansion it is repeated over and over again. It increases the CPU load even more. An XML document of just a few hundred bytes can disrupt all services on a machine within seconds. Example XML:: <!DOCTYPE xmlbomb [<!ENTITY a "1234567890" > <!ENTITY b "&a;&a;&a;&a;&a;&a;&a;&a;&a;"> <!ENTITY c "&b;&b;&b;&b;&b;&b;&b;&b;&b;"> <!ENTITY d "&c;&c;&c;&c;&c;&c;&c;&c;&c;">]> <bomb>&d;</bomb> quadratic blowup entity expansion ---------- A quadratic blowup attack is similar to a Billion Laughs`_ attack; it abuses entity expansion, too. Instead of nested entities it repeats one large entity with a couple of thousand chars over and over again. The

attack isn't as efficient as the exponential case but it avoids triggering countermeasures of parsers against heavily nested entities. Some parsers limit the depth and breadth of a single entity but not the total amount of expanded text throughout an entire XML document. A medium-sized XML document with a couple of hundred kilobytes can require a couple of hundred MB to several GB of memory. When the attack is combined with some level of nested expansion an attacker is able to achieve a higher ratio of success. :: <!DOCTYPE bomb [<!ENTITY a "xxxxxxx... a couple of ten thousand chars">]> <bomb>&a;&a;&a;... repeat</bomb> external entity expansion (remote) ---------- Entity declarations can contain more than just text for replacement. They can also point to external resources by public identifiers or system identifiers. System identifiers are standard URIs. When the URI is a URL (e.g. a ħttp://`locator) some parsers download the resource from the remote location and embed them into the XML document verbatim. Simple example of a parsed external entity:: <!DOCTYPE external [<!ENTITY ee SYSTEM "http://www.python.org/some.xml">]> <root>ⅇ </root> The case of parsed external entities works only for valid XML content. The XML standard also supports unparsed external entities with a NData declaration. External entity expansion opens the door to plenty of exploits. An attacker can abuse a vulnerable XML library and application to rebound and forward network requests with the IP address of the server. It highly depends on the parser and the application what kind of exploit is possible. For example: * An attacker can circumvent firewalls and gain access to restricted resources as all the requests are made from an internal and trustworthy IP address, not from the outside. * An attacker can abuse a service to attack, spy on or DoS your servers but also third party services. The attack is disguised with the IP address of the server and the attacker is able to utilize the high bandwidth of a big machine. * An attacker can exhaust additional resources on the machine, e.g. with requests to a service that doesn't respond or responds with very large files. * An attacker may gain knowledge, when, how often and from which IP address an XML document is accessed. * An attacker could send mail from inside your network if the URL handler supports s̀mtp://` URIs. external entity expansion (local file) ---------- External entities with references to local files are a sub-case of external entity expansion. It's listed as an extra attack because it deserves extra attention. Some XML libraries such as lxml disable network access by default but still allow entity expansion with local file access by default. Local files are either referenced with a f̀ile:///`URL or by a file path (either relative or absolute). An attacker may be able to access and download all files that can be read by the application process. This may include critical configuration files, too. :: <!DOCTYPE external [<!ENTITY ee SYSTEM "file:///PATH/TO/simple.xml">]> <root>ⅇ</root> DTD retrieval ---------- This case is similar to external entity expansion, too. Some XML libraries like Python's xml.dom.pulldom retrieve document type definitions from

remote or local locations. Several attack scenarios from the external entity case apply to this issue as well. :: <?xml version="1.0" encoding="utf-8"?> <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"> <html> <head/> <body>text</body> </html> Python XML Libraries ========== .. csv-table:: vulnerabilities and features :header: "kind", "sax", "etree", "minidom", "pulldom", "xmlrpc", "lxml", "genshi" :widths: 24, 7, 8, 8, 7, 8, 8, 8 :stub-columns: 0 "billion laughs", "**True**", "**True**", "**True**", "**True**", "**True**", "False (1)", "False (5)" "quadratic blowup", "**True**", "**True**", "**True**", "**True**", "**True**", "**True**", "False (5)" "external entity expansion (remote)", "**True**", "False (3)", "False (4)", "**True**", "false", "False (1)", "False (5)" "external entity expansion (local file)", "**True**", "False (3)", "False (4)", "**True**", "false", "**True**", "False (5)" "DTD retrieval", "**True**", "False", "False", "**True**", "false", "False (1)", "False" "gzip bomb", "False", "False", "False", "False", "**True**", "**partly** (2)", "False" "xpath support (7)", "False", "False", "False", "False", "False", "**True**", "False" "xsl(t) support (7)", "False", "False", "False", "False", "False", "**True**", "False" "xinclude support (7)", "False", "**True** (6)", "False", "False", "False", "**True** (6)", "**True**" "C library", "expat", "expat", "expat", "expat", "expat", "libxml2", "expat" 1. Lxml is protected against billion laughs attacks and doesn't do network lookups by default. 2. libxml2 and lxml are not directly vulnerable to gzip decompression bombs but they don't protect you against them either. 3. xml.etree doesn't expand entities and raises a ParserError when an entity occurs. 4. minidom doesn't expand entities and simply returns the unexpanded entity verbatim. 5. genshi.input of genshi 0.6 doesn't support entity expansion and raises a ParserError when an entity occurs. 6. Library has (limited) XInclude support but requires an additional step to process inclusion. 7. These are features but they may introduce exploitable holes, see `Other things to consider`_ Settings in standard library ---------- xml.sax.handler Features ..... feature_external_ges (http://xml.org/sax/features/external-general-entities) disables external entity expansion feature_external_pes (http://xml.org/sax/features/external-parameter-entities) the option is ignored and doesn't modify any functionality DOM xml.dom.xmlbuilder.Options ..... external_parameter_entities ignored external_general_entities ignored external_dtd_subset ignored entities unsure defusedxml ========== The `defusedxml package`_ (`defusedxml on PyPI`_) contains several Python-only workarounds and fixes for denial of service and other vulnerabilities in Python's XML libraries. In order to benefit from the protection you just have to import and use the listed functions / classes from the right defusedxml module instead of the original module. Merely `defusedxml.xmlrpc`_ is implemented as monkey patch. Instead of:: >>> from xml.etree.ElementTree import parse >>> et = parse(xmlfile) alter code to:: >>> from defusedxml.ElementTree import parse >>> et = parse(xmlfile)

Additionally the package has an **untested** function to monkey patch all stdlib modules with `defusedxml.defuse_stdlib()`. All functions and parser classes accept three additional keyword arguments. They return either the same objects as the original functions or compatible subclasses. forbid_dtd (default: False) disallow XML with a `<!DOCTYPE>` processing instruction and raise a *DTDForbidden* exception when a DTD processing instruction is found. forbid_entities (default: True) disallow XML with `<!ENTITY>`declarations inside the DTD and raise an *EntitiesForbidden* exception when an entity is declared. forbid_external (default: True) disallow any access to remote or local resources in external entities or DTD and raising an *ExternalReferenceForbidden* exception when a DTD or entity references an external resource. defusedxml (package) ---------- DefusedXmlException, DTDForbidden, EntitiesForbidden, ExternalReferenceForbidden, NotSupportedError defuse_stdlib() (*experimental*) defusedxml.cElementTree ---------- **NOTE** `defusedxml.cElementTree`is deprecated and will be removed in a future release. Import from `defusedxml.ElementTree`instead. parse(), iterparse(), fromstring(), XMLParser defusedxml.ElementTree ---------- parse(), iterparse(), fromstring(), XMLParser defusedxml.expatreader ---------- create_parser(), DefusedExpatParser defusedxml.sax ---------- parse(), parseString(), make_parser() defusedxml.expatbuilder ---------- parse(), parseString(), DefusedExpatBuilder, DefusedExpatBuilderNS defusedxml.minidom ---------- parse(), parseString() defusedxml.pulldom ---------- parse(), parseString() defusedxml.xmlrpc ---------- The fix is implemented as monkey patch for the stdlib's xmlrpc package (3.x) or xmlrpclib module (2.x). The function `monkey_patch()`enables the fixes, `unmonkey_patch()`removes the patch and puts the code in its former state. The monkey patch protects against XML related attacks as well as decompression bombs and excessively large requests or responses. The default setting is 30 MB for requests, responses and gzip decompression. You can modify the default by changing the module variable MAX_DATA. A value of `-1` disables the limit. defusedxml.lxml ---------- **DEPRECATED** The module is deprecated and will be removed in a future release. The module acts as an *example* how you could protect code that uses lxml.etree. It implements a custom Element class that filters out Entity instances, a custom parser factory and a thread local storage for parser instances. It also has a check_docinfo() function which inspects a tree for internal or external DTDs and entity declarations. In order to check for entities lxml > 3.0 is required. parse(), fromstring() RestrictedElement, GlobalParserTLS, getDefaultParser(), check_docinfo() defusedexpat ========== The `defusedexpat package`_ (`defusedexpat on PyPI`_) comes with binary extensions and a `modified expat`_ library instead of the standard `expat parser`_. It's basically a stand-alone version of the patches for Python's standard library C extensions. Modifications in expat ----------

---------- new definitions:: XML_BOMB_PROTECTION XML_DEFAULT_MAX_ENTITY_INDIRECTIONS XML_DEFAULT_MAX_ENTITY_EXPANSIONS XML_DEFAULT_RESET_DTD new XML_FeatureEnum members:: XML_FEATURE_MAX_ENTITY_INDIRECTIONS XML_FEATURE_MAX_ENTITY_EXPANSIONS XML_FEATURE_IGNORE_DTD new XML_Error members:: XML_ERROR_ENTITY_INDIRECTIONS XML_ERROR_ENTITY_EXPANSION new API functions:: int XML_GetFeature(XML_Parser parser, enum XML_FeatureEnum feature, long *value); int XML_SetFeature(XML_Parser parser, enum XML_FeatureEnum feature, long value); int XML_GetFeatureDefault(enum XML_FeatureEnum feature, long *value); int XML_SetFeatureDefault(enum XML_FeatureEnum feature, long value); XML_FEATURE_MAX_ENTITY_INDIRECTIONS Limit the amount of indirections that are allowed to occur during the expansion of a nested entity. A counter starts when an entity reference is encountered. It resets after the entity is fully expanded. The limit protects the parser against exponential entity expansion attacks (aka billion laughs attack). When the limit is exceeded the parser stops and fails with XML_ERROR_ENTITY_INDIRECTIONS`. A value of 0 disables the protection. Supported range 0 .. UINT_MAX Default 40 XML_FEATURE_MAX_ENTITY_EXPANSIONS Limit the total length of all entity expansions throughout the entire document. The lengths of all entities are accumulated in a parser variable. The setting protects against quadratic blowup attacks (lots of expansions of a large entity declaration). When the sum of all entities exceeds the limit, the parser stops and fails with XML_ERROR_ENTITY_EXPANSION`. A value of 0 disables the protection. Supported range 0 .. UINT_MAX Default 8 MiB XML_FEATURE_RESET_DTD Reset all DTD information after the <!DOCTYPE> block has been parsed. When the flag is set (default: false) all DTD information after the endDoctypeDeclHandler has been called. The flag can be set inside the endDoctypeDeclHandler. Without DTD information any entity reference in the document body leads to XML_ERROR_UNDEFINED_ENTITY`. Supported range 0, 1 Default 0 How to avoid XML vulnerabilities ================================ Best practices ---------- * Don't allow DTDs * Don't expand entities * Don't resolve externals * Limit parse depth * Limit total input size * Limit parse time * Favor a SAX or iterparse-like parser for potential large data * Validate and properly quote arguments to XSL transformations and XPath queries * Don't use XPath expression from untrusted sources * Don't apply XSL transformations that come untrusted sources (based on Brad Hill's `Attacking XML Security`_) Other things to consider ======================== XML, XML parsers and processing libraries have more features and possible issue that could lead to DoS vulnerabilities or security exploits in applications. I have compiled an incomplete list of theoretical issues that need further research and more

attention. The list is deliberately pessimistic and a bit paranoid, too. It contains things that might go wrong under daffy circumstances. attribute blowup / hash collision attack ---------- XML parsers may use an algorithm with quadratic runtime O(n :sup:2) to handle attributes and namespaces. If it uses hash tables (dictionaries) to store attributes and namespaces the implementation may be vulnerable to hash collision attacks, thus reducing the performance to O(n :sup:2) again. In either case an attacker is able to forge a denial of service attack with an XML document that contains thousands upon thousands of attributes in a single node. I haven't researched yet if expat, pyexpat or libxml2 are vulnerable. decompression bomb ---------- The issue of decompression bombs (aka ZIP bomb`_) apply to all XML libraries that can parse compressed XML stream like gzipped HTTP streams or LZMA-ed files. For an attacker it can reduce the amount of transmitted data by three magnitudes or more. Gzip is able to compress 1 GiB zeros to roughly 1 MB, lzma is even better:: $ dd if=/dev/zero bs=1M count=1024 | gzip > zeros.gz $ dd if=/dev/zero bs=1M count=1024 | lzma -z > zeros.xy $ ls -sh zeros.* 1020K zeros.gz 148K zeros.xy None of Python's standard XML libraries decompress streams except for `xmlrpclib`. The module is vulnerable <https://bugs.python.org/issue16043> to decompression bombs. lxml can load and process compressed data through libxml2 transparently. libxml2 can handle even very large blobs of compressed data efficiently without using too much memory. But it doesn't protect applications from decompression bombs. A carefully written SAX or iterparse-like approach can be safe. Processing Instruction ---------- PI`_'s like:: <?xml-stylesheet type="text/xsl" href="style.xsl"?> may impose more threats for XML processing. It depends if and how a processor handles processing instructions. The issue of URL retrieval with network or local file access apply to processing instructions, too. Other DTD features ---------- DTD`_ has more features like `<!NOTATION>`. I haven't researched how these features may be a security threat. XPath ----- XPath statements may introduce DoS vulnerabilities. Code should never execute queries from untrusted sources. An attacker may also be able to create an XML document that makes certain XPath queries costly or resource hungry. XPath injection attacks ---------- XPath injeciton attacks pretty much work like SQL injection attacks. Arguments to XPath queries must be quoted and validated properly, especially when they are taken from the user. The page `Avoid the dangers of XPath injection`_ list some ramifications of XPath injections. Python's standard library doesn't have XPath support. Lxml supports parameterized XPath queries which does proper quoting. You just have to use its xpath() method correctly:: # DON'T >>> tree.xpath("/tag[@id='%s']" % value) # instead do >>> tree.xpath("/tag[@id=$tagid]", tagid=name) XInclude -------- XML Inclusion`_ is another way to load and include external files:: <root xmlns:xi="http://www.w3.org/2001/XInclude"> <xi:include

href="filename.txt" parse="text" /> </root> This feature should be disabled when XML files from an untrusted source are processed. Some Python XML libraries and libxml2 support XInclude but don't have an option to sandbox inclusion and limit it to allowed directories. XMLSchema location ---------- A validating XML parser may download schema files from the information in a `xsi:schemaLocation` attribute. :: <ead xmlns="urn:isbn:1-931666-22-9" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="urn:isbn:1-931666-22-9 http://www.loc.gov/ead/ead.xsd"> </ead> XSL Transformation ---------- You should keep in mind that XSLT is a Turing complete language. Never process XSLT code from unknown or untrusted source! XSLT processors may allow you to interact with external resources in ways you can't even imagine. Some processors even support extensions that allow read/write access to file system, access to JRE objects or scripting with Jython. Example from `Attacking XML Security`_ for Xalan-J:: <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform" xmlns:rt="http://xml.apache.org/xalan/java/java.lang.Runtime" xmlns:ob="http://xml.apache.org/xalan/java/java.lang.Object" exclude-result-prefixes= "rt ob"> <xsl:template match="/"> <xsl:variable name="runtimeObject" select="rt:getRuntime()"/> <xsl:variable name="command" select="rt:exec($runtimeObject, 'c:\Windows\system32\cmd.exe')"/> <xsl:variable name="commandAsString" select="ob:toString($command)"/> <xsl:value-of select="$commandAsString"/> </xsl:template> </xsl:stylesheet> Related CVEs ========== CVE-2013-1664 Unrestricted entity expansion induces DoS vulnerabilities in Python XML libraries (XML bomb) CVE-2013-1665 External entity expansion in Python XML libraries inflicts potential security flaws and DoS vulnerabilities Other languages / frameworks ========== Several other programming languages and frameworks are vulnerable as well. A couple of them are affected by the fact that libxml2 up to 2.9.0 has no protection against quadratic blowup attacks. Most of them have potential dangerous default settings for entity expansion and external entities, too. Perl ---- Perl's XML::Simple is vulnerable to quadratic entity expansion and external entity expansion (both local and remote). Ruby ---- Ruby's REXML document parser is vulnerable to entity expansion attacks (both quadratic and exponential) but it doesn't do external entity expansion by default. In order to counteract entity expansion you have to disable the feature:: REXML::Document.entity_expansion_limit = 0 libxml-ruby and hpricot don't expand entities in their default configuration. PHP --- PHP's SimpleXML API is vulnerable to quadratic entity expansion and loads entities from local and remote resources. The option `LIBXML_NONET` disables network access but still allows local file access. `LIBXML_NOENT` seems to have no effect on entity expansion in PHP 5.4.6. C# / .NET / Mono ---------- Information in XML DoS and Defenses (MSDN)`_

suggest that .NET is vulnerable with its default settings. The article contains code snippets how to create a secure XML reader:: XmlReaderSettings settings = new XmlReaderSettings(); settings.ProhibitDtd = false; settings.MaxCharactersFromEntities = 1024; settings.XmlResolver = null; XmlReader reader = XmlReader.Create(stream, settings); Java ---- Untested. The documentation of Xerces and its Xerces SecurityMananger`_ sounds like Xerces is also vulnerable to billion laugh attacks with its default settings. It also does entity resolving when an `org.xml.sax.EntityResolver`is configured. I'm not yet sure about the default setting here. Java specialists suggest to have a custom builder factory:: DocumentBuilderFactory builderFactory = DocumentBuilderFactory.newInstance(); builderFactory.setXIncludeAware(False); builderFactory.setExpandEntityReferences(False); builderFactory.setFeature(XMLConstants.FEATURE_SECURE_PROCESSING, True); # either builderFactory.setFeature("http://apache.org/xml/features/disallow-doctype-decl", True); # or if you need DTDs builderFactory.setFeature("http://xml.org/sax/features/external-general-entities", False); builderFactory.setFeature("http://xml.org/sax/features/external-parameter-entities", False); builderFactory.setFeature("http://apache.org/xml/features/nonvalidating/load-external-dtd", False); builderFactory.setFeature("http://apache.org/xml/features/nonvalidating/load-dtd-grammar", False); TODO ==== * DOM: Use xml.dom.xmlbuilder options for entity handling * SAX: take feature_external_ges and feature_external_pes (?) into account * test experimental monkey patching of stdlib modules * improve documentation License ======= Copyright (c) 2013-2017 by Christian Heimes <christian@python.org> Licensed to PSF under a Contributor Agreement. See https://www.python.org/psf/license for licensing details. Acknowledgements ================ Brett Cannon (Python Core developer) review and code cleanup Antoine Pitrou (Python Core developer) code review Aaron Patterson, Ben Murphy and Michael Koziarski (Ruby community) Many thanks to Aaron, Ben and Michael from the Ruby community for their report and assistance. Thierry Carrez (OpenStack) Many thanks to Thierry for his report to the Python Security Response Team on behalf of the OpenStack security team. Carl Meyer (Django) Many thanks to Carl for his report to PSRT on behalf of the Django security team. Daniel Veillard (libxml2) Many thanks to Daniel for his insight and assistance with libxml2. semantics GmbH (https://www.semantics.de/) Many thanks to my employer semantics for letting me work on the issue during working hours as part of semantics's open source initiative. References ========== * XML DoS and Defenses

(MSDN)`_ * Billion Laughs`_ on Wikipedia * ZIP bomb`_ on Wikipedia * `Configure SAX parsers for secure processing`_ * Testing for XML Injection`_ .. _defusedxml package: https://github.com/tiran/defusedxml .. _defusedxml on PyPI: https://pypi.python.org/pypi/defusedxml .. _defusedexpat package: https://github.com/tiran/defusedexpat .. _defusedexpat on PyPI: https://pypi.python.org/pypi/defusedexpat .. _modified expat: https://github.com/tiran/expat .. _expat parser: http://expat.sourceforge.net/ .. _Attacking XML Security: https://www.isecpartners.com/media/12976/iSEC-HILL-Attacking-XML-Security-bh07.pdf .. _Billion Laughs: https://en.wikipedia.org/wiki/Billion_laughs .. _XML DoS and Defenses (MSDN): https://msdn.microsoft.com/en-us/magazine/ee335713.aspx .. _ZIP bomb: https://en.wikipedia.org/wiki/Zip_bomb .. _DTD: https://en.wikipedia.org/wiki/Document_Type_Definition .. _PI: https://en.wikipedia.org/wiki/Processing_Instruction .. _Avoid the dangers of XPath injection: http://www.ibm.com/developerworks/xml/library/x-xpathinjection/index.html .. _Configure SAX parsers for secure processing: http://www.ibm.com/developerworks/xml/library/x-tipcfsx/index.html .. _Testing for XML Injection: https://www.owasp.org/index.php/Testing_for_XML_Injection_(OWASP-DV-008) .. _Xerces SecurityMananger: https://xerces.apache.org/xerces2-j/javadocs/xerces2/org/apache/xerces/util/SecurityManager.html .. _XML Inclusion: https://www.w3.org/TR/xinclude/#include_element Changelog ========= defusedxml 0.7.1 ---------- *Release date: 08-Mar-2021* - Fix regression `defusedxml.ElementTree.ParseError`(#63) The ParseError`exception is now the same class object as `xml.etree.ElementTree.ParseError`again. defusedxml 0.7.0 ---------- *Release date: 4-Mar-2021* - No changes defusedxml 0.7.0rc2 ---------- *Release date: 12-Jan-2021* - Re-add and deprecate `defusedxml.cElementTree`- Use GitHub Actions instead of TravisCI - Restore ElementTree`attribute of `xml.etree`module after patching defusedxml 0.7.0rc1 ---------- *Release date: 04-May-2020* - Add support for Python 3.9 - `defusedxml.cElementTree`is not available with Python 3.9. - Python 2 is deprecate. Support for Python 2 will be removed in 0.8.0. defusedxml 0.6.0 ---------- *Release date: 17-Apr-2019* - Increase test coverage. - Add badges to README. defusedxml 0.6.0rc1 ---------- *Release date: 14-Apr-2019* - Test on Python 3.7 stable and 3.8-dev - Drop support for Python 3.4 - No longer pass *html* argument to XMLParse. It has been deprecated and ignored for a long time. The DefusedXMLParser still takes a html argument. A deprecation warning is issued when the argument is False and a TypeError when it's True. - defusedxml now fails early when pyexpat stdlib module is not available or broken. - defusedxml.ElementTree.__all__ now lists ParseError as public attribute. - The defusedxml.ElementTree and defusedxml.cElementTree modules had a typo and used XMLParse instead of XMLParser as an alias

for DefusedXMLParser. Both the old and fixed name are now available. defusedxml 0.5.0 ---------- *Release date: 07-Feb-2017* - No changes defusedxml 0.5.0.rc1 ---------- *Release date: 28-Jan-2017* - Add compatibility with Python 3.6 - Drop support for Python 2.6, 3.1, 3.2, 3.3 - Fix lxml tests (XMLSyntaxError: Detected an entity reference loop) defusedxml 0.4.1 ---------- *Release date: 28-Mar-2013* - Add more demo exploits, e.g. python_external.py and Xalan XSLT demos. - Improved documentation. defusedxml 0.4 ---------- *Release date: 25-Feb-2013* - As per http://seclists.org/oss-sec/2013/q1/340 please REJECT CVE-2013-0278, CVE-2013-0279 and CVE-2013-0280 and use CVE-2013-1664, CVE-2013-1665 for OpenStack/etc. - Add missing parser_list argument to sax.make_parser(). The argument is ignored, though. (thanks to Florian Apolloner) - Add demo exploit for external entity attack on Python's SAX parser, XML-RPC and WebDAV. defusedxml 0.3 ---------- *Release date: 19-Feb-2013* - Improve documentation defusedxml 0.2 ---------- *Release date: 15-Feb-2013* - Rename ExternalEntitiesForbidden to ExternalReferenceForbidden - Rename defusedxml.lxml.check_dtd() to check_docinfo() - Unify argument names in callbacks - Add arguments and formatted representation to exceptions - Add forbid_external argument to all functions and classes - More tests - LOTS of documentation - Add example code for other languages (Ruby, Perl, PHP) and parsers (Genshi) - Add protection against XML and gzip attacks to xmlrpclib defusedxml 0.1 ---------- *Release date: 08-Feb-2013* - Initial and internal release for PSRT review

Package Info

Package Author(s)	christian@python.org
Package Manager	Pip
Project URL	https://github.com/tiran/defusedxml
Package Download URL	https://files.pythonhosted.org/packages/0f/d5/c66da9b79e5bdb124974bfe172b4daf3c984ebd9c2a06e2b8a4dc7331c72/defusedxml-0.7.1.tar.gz
Dependency Path	solnlib > defusedxml, splunktaucclib > defusedxml, splunktaucclib > solnlib > defusedxml

Declared License(s)

None

Discovered License(s)

PSF-2.0

Copyright(s) i.e., "Copyright (c)

Notice File(s)

None

deprecation 2.1.0 pip+deprecation$2.1.0 • Transitive

Description

deprecation ========== .. image:: https://readthedocs.org/projects/deprecation/badge/?version=latest :target: http://deprecation.readthedocs.io/en/latest/ :alt: Documentation Status .. image:: https://travis-ci.org/briancurtin/deprecation.svg?branch=master :target: https://travis-ci.org/briancurtin/deprecation .. image:: https://codecov.io/gh/briancurtin/deprecation/branch/master/graph/badge.svg :target: https://codecov.io/gh/briancurtin/deprecation The `deprecation` library provides a `deprecated`decorator and a `fail_if_not_removed` decorator for your tests. Together, the two enable the automation of several things: 1. The docstring of a deprecated method gets the deprecation details appended to the end of it. If you generate your API docs direct from your source, you don't need to worry about writing your own notification. You also don't need to worry about forgetting to write it. It's done for you. 2. Rather than having code live on forever because you only deprecated it but never actually moved on from it, you can have your tests tell you when it's time to remove the code. The `@deprecated` decorator can be told when it's time to entirely remove the code, which causes `@fail_if_not_removed`to raise an `AssertionError`, causing either your unittest or py.test tests to fail. See http://deprecation.readthedocs.io/ for the full documentation. Installation ============ :: pip install deprecation Usage ===== :: import deprecation @deprecation.deprecated(deprecated_in="1.0", removed_in="2.0", current_version=__version__, details="Use the bar function instead") def foo(): """Do some stuff""" return 1 ...but doesn't Python ignore `DeprecationWarning`? == Yes, by default since 2.7—and for good reason [#]_ —and this works fine with that. 1. It often makes sense for you to run your tests with a `-W`flag

or the PYTHONWARNINGS`environment variable so you catch warnings in development and handle them appropriately. The warnings raised by this library show up there, as they're subclasses of the built-in DeprecationWarning`. See the Command Line <https://docs.python.org/2/using/cmdline.html#cmdoption-W>`_ and Environment Variable <https://docs.python.org/2/using/cmdline.html#envvar-PYTHONWARNINGS>`_ documentation for more details. 2. Even if you don't enable those things, the behavior of this library remains the same. The docstrings will still be updated and the tests will still fail when they need to. You'll get the benefits regardless of what Python cares about DeprecationWarning`. ---- .. [#] Exposing application users to DeprecationWarning\s that are emitted by lower-level code needlessly involves end-users in "how things are done." It often leads to users raising issues about warnings they're presented, which on one hand is done rightfully so, as it's been presented to them as some sort of issue to resolve. However, at the same time, the warning could be well known and planned for. From either side, loud DeprecationWarning\s can be seen as noise that isn't necessary outside of development.

Package Info

Package Author(s)	brian@python.org
Package Manager	Pip
Project URL	http://deprecation.readthedocs.io/
Package Download URL	https://files.pythonhosted.org/packages/5a/d3/8ae2869247df154b64c1884d7346d412fed0c49df84db635aab2d1c40e62/deprecation-2.1.0.tar.gz
Dependency Path	solnlib > splunk-sdk > deprecation, splunk-sdk > deprecation, splunktaucclib > solnlib > splunk-sdk > deprecation, splunktaucclib > splunk-sdk > deprecation

Declared License(s)

Apache-2.0

Copyright(s)	yyyy} {name of copyright owner}

Discovered License(s)

None

Notice File(s)

None

idna 3.18 pip+idna$3.18 • Transitive

Description

Internationalized Domain Names in Applications (IDNA)

Package Info

Package Author(s)	Kim Davies <kim+pypi@gumleaf.org>
Package Manager	Pip
Project URL	https://pypi.org/project/idna/
Package Download URL	https://files.pythonhosted.org/packages/cd/63/9496c57188a2ee585e0f1db071d75089a11e98aa86eb99d9d7618fc1edce/idna-3.18.tar.gz
Dependency Path	requests > idna

Declared License(s)

BSD-3-Clause

Copyright(s) 2013-2026 Kim Davies and contributors.

Discovered License(s)

None

Notice File(s)

None

PySocks 1.7.1

pip+pysocks$1.7.1 • Transitive

Description

A Python SOCKS client module. See https://github.com/Anorov/PySocks for more information.

Package Info

Package Author(s)	anorov.vorona@gmail.com
Package Manager	Pip
Project URL	https://github.com/Anorov/PySocks
Package Download URL	https://files.pythonhosted.org/packages/bd/11/293dd436aea955d45fc4e8a35b6ae7270f5b8e00b53cf6c024c83b657a11/PySocks-1.7.1.tar.gz
Dependency Path	splunktaucclib > pysocks

Declared License(s)

BSD-3-Clause

Copyright(s) 2006 Dan-Haim. All rights reserved.

Discovered License(s)

Apache-2.0

Copyright(s) None

Notice File(s)

None

sortedcontainers 2.4.0

pip+sortedcontainers$2.4.0 • Transitive

Description

Sorted Containers -- Sorted List, Sorted Dict, Sorted Set

Package Info

Package Author(s)	contact@grantjenks.com
Package Manager	Pip
Project URL	http://www.grantjenks.com/docs/sortedcontainers/
Package Download URL	https://files.pythonhosted.org/packages/e8/c4/ba2f8066cceb6f23394729afe52f3bf7adec04bf9ed2c820b39e19299111/sortedcontainers-2.4.0.tar.gz
Dependency Path	solnlib > sortedcontainers, splunktaucclib > solnlib > sortedcontainers

Declared License(s)

Apache-2.0

Copyright(s)	2014-2019 Grant Jenks

Discovered License(s)

None

Notice File(s)

None

urllib3 1.26.20

pip+urllib3$1.26.20 • Transitive

Description

<h1 align="center"> ![urllib3] (https://github.com/urllib3/urllib3/raw/main/docs/_static/banner_github.svg) </h1> <p align="center">   <img alt="Join our Discord"

src="https://img.shields.io/discord/756342717725933608?color=%237289da&label=discord" />
    </p> urllib3 is a powerful, *user-friendly* HTTP client for Python. Much of the Python ecosystem already uses urllib3 and you should too. urllib3 brings many critical features that are missing from the Python standard libraries: - Thread safety. - Connection pooling. - Client-side SSL/TLS verification. - File uploads with multipart encoding. - Helpers for retrying requests and dealing with HTTP redirects. - Support for gzip, deflate, brotli, and zstd encoding. - Proxy support for HTTP and SOCKS. - 100% test coverage. urllib3 is powerful and easy to use: þython3 >>> import urllib3 >>> resp = urllib3.request("GET", "http://httpbin.org/robots.txt") >>> resp.status 200 >>> resp.data b"User-agent: *\nDisallow: /deny\n" `## Installing urllib3 can be installed with [pip](https://pip.pypa.io): þash $ python -m pip install urllib3 `Alternatively, you can grab the latest source code from [GitHub](https://github.com/urllib3/urllib3): þash $ git clone https://github.com/urllib3/urllib3.git $ cd urllib3 $ pip install . `## Documentation urllib3 has usage and reference documentation at urllib3.readthedocs.io. ## Community urllib3 has a [community Discord channel](https://discord.gg/urllib3) for asking questions and collaborating with other contributors. Drop by and say hello □ ## Contributing urllib3 happily accepts contributions. Please see our [contributing documentation](https://urllib3.readthedocs.io/en/latest/contributing.html) for some tips on getting started. ## Security Disclosures To report a security vulnerability, please use the [Tidelift security contact](https://tidelift.com/security). Tidelift will coordinate the fix and disclosure with maintainers. ## Maintainers - [@sethmlarson](https://github.com/sethmlarson) (Seth M. Larson) - [@pquentin](https://github.com/pquentin) (Quentin Pradet) - [@illia-v]

(https://github.com/illia-v) (Illia Volochii) - [@theacodes] (https://github.com/theacodes) (Thea Flowers) - [@haikuginger] (https://github.com/haikuginger) (Jess Shapiro) - [@lukasa] (https://github.com/lukasa) (Cory Benfield) - [@sigmavirus24] (https://github.com/sigmavirus24) (Ian Stapleton Cordasco) - [@shazow] (https://github.com/shazow) (Andrey Petrov) □ ## Sponsorship If your company benefits from this library, please consider [sponsoring its development](https://urllib3.readthedocs.io/en/latest/sponsors.html). ## For Enterprise Professional support for urllib3 is available as part of the [Tidelift Subscription][1]. Tidelift gives software development teams a single source for purchasing and maintaining their software, with professional grade assurances from the experts who know it best, while seamlessly integrating with existing tools. [1]: https://tidelift.com/subscription/pkg/pypi-urllib3?utm_source=pypi-urllib3&utm_medium=referral&utm_campaign=readme

Package Info

Package Author(s)	Andrey Petrov <andrey.petrov@shazow.net>
Package Manager	Pip
Project URL	https://pypi.org/project/urllib3/
Package Download URL	https://files.pythonhosted.org/packages/e4/e8/6ff5e6bc22095cfc59b6ea711b687e2b7ed4bdb373f7eeec370a97d7392f/urllib3-1.26.20.tar.gz
Dependency Path	requests > urllib3, splunktaucclib > urllib3

Declared License(s)

MIT

Copyright(s)	2008-2020 Andrey Petrov and contributors (see CONTRIBUTORS.txt)

Discovered License(s)

None

Notice File(s)

None

packaging 24.0 pip+packaging$24.0 • Transitive

Description

packaging ========= .. start-intro Reusable core utilities for various Python Packaging ìnteroperability specifications <https://packaging.python.org/specifications/>`_. This library provides utilities that implement the interoperability specifications which have clearly one correct behaviour (eg: :pep:`440) or benefit greatly from having a single shared implementation (eg: :pep:`425). .. end-intro The `packaging`project includes the following: version handling, specifiers, markers, requirements, tags, utilities. Documentation ---------- The `documentation`_ provides information and the API for the following: - Version Handling - Specifiers - Markers - Requirements - Tags - Utilities Installation ------------ Use `pip`to install these utilities:: pip install packaging The `packaging`library uses calendar-based versioning (YY.N). Discussion ---------- If you run into bugs, you can file them in our `issue tracker`_. You can also join `#pypa`on Freenode to ask questions or get involved. .. _`documentation`: https://packaging.pypa.io/ .. _`issue tracker`: https://github.com/pypa/packaging/issues Code of Conduct ---------- Everyone interacting in the packaging project's codebases, issue trackers, chat rooms, and mailing lists is expected to follow the PSF Code of Conduct`_. .. _PSF Code of Conduct: https://github.com/pypa/.github/blob/main/CODE_OF_CONDUCT.md Contributing ---------- The `CONTRIBUTING.rst`file outlines how to contribute to this project as well as how to report a potential security issue. The documentation for this project also covers information about `project development`_ and `security`_. .. _`project development`: https://packaging.pypa.io/en/latest/development/ .. _`security`: https://packaging.pypa.io/en/latest/security/ Project History ---------- Please review the `CHANGELOG.rst`file or the `Changelog documentation`_ for recent changes and project history. .. _`Changelog documentation`: https://packaging.pypa.io/en/latest/changelog/

Package Info

Package Author(s)	Donald Stufft <donald@stufft.io>
Package Manager	Pip
Project URL	https://pypi.org/project/packaging/
Package Download URL	https://files.pythonhosted.org/packages/ee/b5/b43a27ac7472e1818c4bafd44430e69605baefe1f34440593e0332ec8b4d/packaging-24.0.tar.g

z

Dependency Path	solnlib > splunk-sdk > deprecation > packaging, splunk-sdk > deprecation > packaging, splunktaucclib > solnlib > splunk-sdk > deprecation > packaging, splunktaucclib > splunk-sdk > deprecation > packaging

Declared License(s)

Apache-2.0

Copyright(s)	None

Discovered License(s)

None

Notice File(s)

None