

Package ‘oysteR’

October 9, 2025

Title Scans R Projects for Vulnerable Third Party Dependencies

Version 0.1.4

Maintainer Colin Gillespie <csgillespie@gmail.com>

Description Collects a list of your third party R packages, and scans them with the 'OSS' Index provided by 'Sonatype', reporting back on any vulnerabilities that are found in the third party packages you use.

License Apache License 2.0 | file LICENSE

URL <https://github.com/sonatype-nexus-community/oysteR>

BugReports <https://github.com/sonatype-nexus-community/oysteR/issues>

Depends R (>= 4.0.0)

Imports cli, dplyr, glue, httr, jsonlite, purrr, rjson, rlang, stringr, tibble, tidyr, utils, yaml

Suggests covr, httptest, knitr, rmarkdown, testthat (>= 2.1.0)

VignetteBuilder knitr

Encoding UTF-8

RoxygenNote 7.3.3

NeedsCompilation no

Author Jeffry Hesse [aut],
Brittany Belle [aut],
Colin Gillespie [aut, cre] (ORCID:
<<https://orcid.org/0000-0003-1787-0275>>),
Dan Rollo [aut],
Josiah Parry [aut] (ORCID: <<https://orcid.org/0000-0001-9910-865X>>),
Sonatype [cph]

Repository CRAN

Date/Publication 2025-10-09 15:00:02 UTC

Contents

audit	2
audit_conda	3
audit_description	3
audit_installed_r_pkgs	4
audit_renv_lock	5
audit_req_txt	5
expect_secure	6
get_vulnerabilities	7
remove_cache	7
Index	8

audit	<i>Search for Package Vulnerabilities</i>
-------	---

Description

Search the OSS Index for known package vulnerabilities in any of the supported ecosystems— e.g. CRAN, PyPI, Conda, NPM, Maven, etc. see <https://ossindex.sonatype.org/ecosystems> for full list.

Usage

```
audit(pkg, version, type, verbose = TRUE, token = NULL)
```

Arguments

pkg	A vector of package names to search in the OSS Index.
version	The specific package version to search for. By default it will search all known versions. If not *, must be the same length as pkg.
type	The package management environment. For R packages, set equal to "cran". This defaults to "cran". See https://ossindex.sonatype.org/ecosystems .
verbose	Default TRUE.
token	If NULL, looks at OSSINDEX_USER & OSSINDEX_TOKEN, env variables. If those aren't available, try "~/.ossindex/.oss-index-config"

Examples

```
pkg = c("abind", "acepack")
version = c("1.4-5", "1.4.1")
audit(pkg, version, type = "cran")
```

audit_conda	<i>Audit a conda environment file</i>
-------------	---------------------------------------

Description

This function searches the OSS index for vulnerabilities recorded for packages listed in a Conda environment file typically called `environment.yml` but are subject to varied names. Conda environment can contain packages from both Conda and PyPI. All packages will be audited.

Usage

```
audit_conda(dir = ".", fname = "environment.yml", verbose = TRUE)
```

Arguments

<code>dir</code>	The directory containing a Conda environment yaml file.
<code>fname</code>	The file name of conda environment yaml file.
<code>verbose</code>	Default TRUE.

Examples

```
## Not run:  
# Looks for a environment.yml file in dir  
audit_conda(dir = ".")  
  
## End(Not run)
```

audit_description	<i>Audits Packages Listed in a DESCRIPTION file</i>
-------------------	---

Description

Looks for a `DESCRIPTION` file in `dir`, then extract the packages in the fields & calculates the dependency tree.

Usage

```
audit_description(  
  dir = ".",  
  fields = c("Depends", "Imports", "Suggests"),  
  verbose = TRUE  
)
```

Arguments

dir	The file path of an renv.lock file.
fields	The DESCRIPTION field to parse. Default is Depends, Import, & Suggests.
verbose	Default TRUE.

Examples

```
## Not run:
# Looks for a DESCRIPTION file in dir
audit_description(dir = ".")

## End(Not run)
```

```
audit_installed_r_pkgs
```

```
Audit Installed Packages
```

Description

Audits all installed packages by calling `installed.packages()` and checking them against the OSS Index.

Usage

```
audit_installed_r_pkgs(verbose = TRUE, token = NULL)
```

Arguments

verbose	Default TRUE.
token	If NULL, looks at OSSINDEX_USER & OSSINDEX_TOKEN, env variables. If those aren't available, try " <code>~/ .ossindex/.oss-index-config</code> "

Value

A tibble/data.frame.

Examples

```
## Not run:
# Audit installed packages
# This calls installed.packages()
pkgs = audit_installed_r_pkgs()

## End(Not run)
```

audit_renv_lock	<i>Audit an renv.lock File</i>
-----------------	--------------------------------

Description

This function searches the OSS index for vulnerabilities recorded for packages listed in an `renv.lock` file. An `renv.lock` file is created by the `{renv}` package which is used for project level package management in R.

Usage

```
audit_renv_lock(dir = ".", verbose = TRUE)
```

Arguments

<code>dir</code>	The file path of an <code>renv.lock</code> file.
<code>verbose</code>	Default TRUE.

Examples

```
## Not run:  
# Looks for renv.lock file in dir  
audit_renv_lock(dir = ".")  
  
## End(Not run)
```

audit_req_txt	<i>Audit a requirements.txt File</i>
---------------	--------------------------------------

Description

This function searches the OSS index for vulnerabilities recorded for packages listed in a `requirements.txt` file based on PyPi.

Usage

```
audit_req_txt(dir = ".", verbose = TRUE)
```

Arguments

<code>dir</code>	The file path of a <code>requirements.txt</code> file.
<code>verbose</code>	Default TRUE.

Details

pip is a standard of python package management based on the Python Package Index (PyPI). pip uses a requirements.txt file to manage of Python libraries. The requirements.txt file contains package names and versions (often used to manage a virtual environment).

Examples

```
## Not run:
# Looks for a requirements.txt file in dir
audit_description(dir = ".")

## End(Not run)
```

expect_secure

Vulnerability Detection via Testthat

Description

A testthat version for detecting vulnerabilities. This function is used within the testthat framework. As testthat strips out the repositories from options, we have to set the value locally in the function, i.e. the value you have in getOption("repos") is not used.

Usage

```
expect_secure(pkg, repo = "https://cran.rstudio.com", verbose = FALSE)
```

Arguments

pkg	The pkg to check
repo	The CRAN repository, used to get version numbers
verbose	Default TRUE.

Details

An important proviso is that we are only testing packages for specific versions. By default, this will be the latest version on CRAN. This may differ for users or if you are using a CRAN snapshot. For the latter, simply change the repo parameter.

Examples

```
## Not run:
# Typically used inside testthat
oysteR::expect_secure("oysteR")

## End(Not run)
```

get_vulnerabilities	<i>Extract vulnerabilities</i>
---------------------	--------------------------------

Description

Parse the audit data frame (obtained via audit_deps), and extract the vulnerabilities.

Usage

```
get_vulnerabilities(audit)
```

Arguments

audit	Output from audit_deps.
-------	-------------------------

Examples

```
## Not run:
# Audit installed packages
# This calls installed.packages()
# pkgs = audit_deps()

# Or pass your own packages
pkgs = data.frame(package = c("abind", "acepack"),
                  version = c("1.4-5", "1.4.1"))
#deps = audit_deps(pkgs)
#get_vulnerabilities(deps)

## End(Not run)
```

remove_cache	<i>Remove cache</i>
--------------	---------------------

Description

The OSS cache is located at `tools::R_user_dir("oyster", which = "cache")`. The function `R_user_dir()` is only available for `R >= 4.0.0`. Packages are cached for 12 hours, then refreshed at the next audit

Usage

```
remove_cache()
```

Index

`audit`, [2](#)
`audit_conda`, [3](#)
`audit_description`, [3](#)
`audit_installed_r_pkgs`, [4](#)
`audit_renv_lock`, [5](#)
`audit_req_txt`, [5](#)

`expect_secure`, [6](#)

`get_vulnerabilities`, [7](#)

`remove_cache`, [7](#)