OpenVAS Reporting Documentation

Release 1.4.4

TheGroundZero

Feb 08, 2023

Contents

A tool to convert OpenVAS XML into reports.

CHAPTER 1

Table of contents

1.1 Changelog

1.5.0 - New Features:

- Included option (–config-file) to define a .yaml file with all options but input and output filenames. if this option is used all other options are ignored. any options not present in this file will be set to default
- Included option (–report-type) to define the type of report and created two new types of reports: - a report summarizing the hosts with the highest number of vulnerabilities and sum of all its cvss severities and including a tab for each host listing each vulnerability - a csv report ordered by host with all vulnerabilities (same fields as the by vulnerability type)

  (I don't believe it's worth creating a .docx version of this report, so I'm not creating it)
- Included option (–network-exclude) to define a file with a list of ips or ipcidrs or range of ips (one by line) that will be excluded from the report
- Included option (–network-include) to define a file with a list of ips or ipcidrs or range of ips (one by line) that will be included in the report
- Included option (–regex-include) to define a file with a list of regex expressions to include in the report the regex expressions will be matched against the name of the vulnerability
- Included option (–regex-exclude) to define a file with a list of regex expressions to exclude in the report the regex expressions will be matched against the name of the vulnerability
- Included option (–cve-include) to define a file with a list of CVE numbers to include in the report
- Included option (–cve-exclude) to define a file with a list of CVE numbers to exclude from the report

-Fixes:

- Major code refactor to include the new reports and the new options
- Fix module packaging and shell script executions now run ok (import 'main' in top source __init__.py so the egg may be found)
- Converted module packaging to python3.6+ packaging using setup.cfg e pyproject.toml

- Removed package top dir setup.py and requirements.txt files that are not used anymore
- Updated README.md to reflect those changes

1.4.2 - Fixed "ValueError: Unknown format code 'f' for object of type 'str'"

1.4.1 - Small bugfixes and code refactoring

1.4.0 - Use Word template for report building

1.3.1 - Add charts to Word document using matplotlib. Some code clean-up and small lay-out changes in Excel.

1.3.0 - Fix retrieval of description and other useful info by parsing <tags> instead of <description>

1.2.3 - Implement https://github.com/cr0hn/openvas_to_report/pull/12

1.2.2 - Fix bug where port info was not correctly extracted

1.2.1 - Fix bug where affected hosts were added on wrong row in Excel export

1.2.0 - Functional export to Word document (.docx). Includes some formatting. TODO: graphs

1.1.0a - Support for exporting to Word document (.docx). Limited formatting, needs more testing

1.0.1a - Small updates, preparing for export to other formats

1.0.0 - First official release, supports export to Excel with graphs, ToC and worksheet per vulnerability

1.2 Usage

1.2.1 Installation

You can install this package directly from source by cloning the Git repository.

```
# Install pip
apt(-get) install python3 python3-pip # Debian, Ubuntu
yum -y install python3 python3-pip    # CentOS
dnf install python3 python3-pip       # Fedora

# Clone repo
git clone git@github.com:TheGroundZero/openvasreporting.git

# Install requirements
cd openvasreporting
pip3 install -r requirements.txt
pip3 install build --upgrade
pip3 install pip --upgrade
python -m build .
pip3 install dist/Openvas_Reporting[...].whl
```

Alternatively, you can install the package through the Python package installer 'pip'.

```
# Install pip
apt(-get) install python3 python3-pip # Debian, Ubuntu
yum -y install python3 python3-pip    # CentOS
dnf install python3 python3-pip       # Fedora

# Install the package
pip install OpenVAS-Reporting
```

1.2.2 Command line usage

```
# When working from the Git repo
python3 -m openvasreporting -i *.xml [-i ...] [-c config.yml] [-o openvas_report]␣
↪[-f xlsx] [-l none] [-t "openvasreporting/src/openvas-template.docx"] [-T␣
↪vulnerability] [-n included_networks] [-N excluded-networks] [-r included-regex]␣
↪[-R excluded-regex] [-e included-cve] [-E excluded-cve]
# When using the pip package
openvasreporting -i *.xml [-i ...] [-c config.yml] [-o openvas_report] [-f xlsx] [-
↪l none] [-t "openvasreporting/src/openvas-template.docx"] [-T vulnerability] [-n␣
↪included_networks] [-N excluded-networks] [-r included-regex] [-R excluded-
↪regex] [-e included-cve] [-E excluded-cve]
```

-i, –input

Mandatory
Selects the OpenVAS XML report file(s) to be used as input.
Accepts one or more inputs, including wildcards

-o, –output

Optional
Name of the output file, without extension.
Defaults to: openvas_report

-c, –config-file

Optional
Path to a .yml file containing the configuration (format, level, type, filters)
If this option is used all other options (but input and output files) will be ignored
Defaults to: None

-f, –format

Optional
Type of output file.
Valid values are: xlsx, docx, csv
Defaults to: xlsx

-l, –level

Optional
Minimal severity level of finding before it's included in the report.
Valid values are: c(ritical), h(igh), m(edium), l(low), n(one)
Defaults to: none

-t, –template

Optional, only used with '-f docx'
Template document for docx export. Document must contain formatting for styles used in export.
Valid values are: path to a docx file
Defaults to: openvasreporting/src/openvas-template.docx

-T, –report-type

Optional
Selects if will list hosts by vulnerability (v) or vulnerabilities by host (h)
Valid values are: v, h, vulnerabiity, host
Defaults to: vulnerability

-e, –network-include

Optional

path to a file containing a list of ips, ipcidrs or ipaddrs (one per line) that
will be included in the report
Defaults to: all hosts with appropriate level will be included

-E, –network-exclude

Optional
path to a file containing a list of ips, ipcidrs or ipaddrs (one per line) that
will be excluded from the report
Defaults to: no excluded hosts

-r, –regex-include

Optional
path to a file containing a list of regex expressions that will be matched against
the name of the vulnerability field to be filtered into the report
Defaults to: all vulnerabilities will be included

-R, –regex-exclude

Optional
path to a file containing a list of regex expressions that will be matched against
the name of the vulnerability field to be filtered out of the report
Defaults to: no excluded vulnerabilities

-e, –cve-include

Optional
path to a file containing a list of CVEs (format CVEYYYY-nnn...) that will be
filtered into the report
Defaults to: all vulnerabilities with -l level will be included

-C, –cve-exclude

Optional
path to a file containing a list of CVEs (format CVEYYYY-nnn...) that will be
filtered out of the report
Defaults to: no excluded hosts

Todo: [Feature] Export to other formats (PDF, [proper] CSV)

Export to Excel

By default (or when passing the –format xlsx parameter), the tool will export reports in Excel (xlsx) format.

This report contains a summary sheet, table of contents, and a sheet per vulnerability containing vulnerability details and a list of affected hosts.

Examples

Create Excel report from 1 OpenVAS XML report using default settings

```
openvasreporting -i openvasreport.xml
```

Create Excel report from multiple OpenVAS XML report using default settings

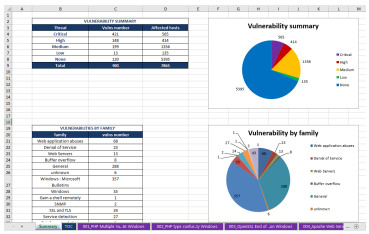
```
openvasreporting -i *.xml
# OR
openvasreporting -i openvasreport.xml -i openvasreport1.xml -i openvasreport2.xml␣
↪[-i ...]
```

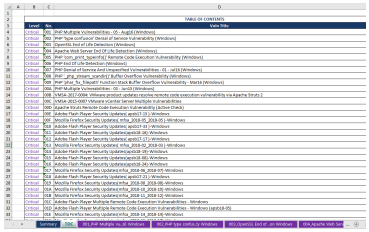
Create Excel report from 1 OpenVAS XML report, reporting only severity level high and up

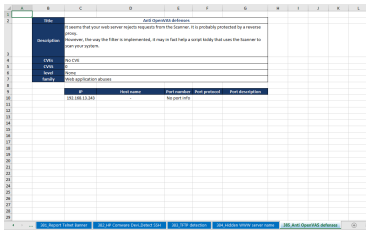
```
openvasreporting.py -i openvasreport.xml -o openvas_report -f xlsx -l h
```

Result

The final report will look similar to this:







Vulnerability detail worksheets are sorted according to CVSS score and are colored according to the threat level.

Export to Word

When passing the –format docx parameter, the tool will export reports in Word (docx) format.

This report contains a summary sheet, table of contents, and a sheet per vulnerability containing vulnerability details and a list of affected hosts.

Examples

Create Word report from 1 OpenVAS XML report using default settings

```
openvasreporting -i openvasreport.xml -f docx
```

Create Word report from multiple OpenVAS XML report using default settings

```
openvasreporting -i *.xml -f docx
# OR
openvasreporting -i openvasreport.xml -i openvasreport1.xml -i openvasreport2.xml␣
↪[-i ...] -f docx
```

Create Word report from 1 OpenVAS XML report, reporting only severity level high and up

```
openvasreporting -i openvasreport.xml -o openvas_report -f docx -l h
```

Create Word report using a different template

```
openvasreporting -i openvasreport.xml -o openvas_report -f docx -t /home/user/
↪myOpenvasTemplate.docx
```

The custom template document must contain a definition for the following styles:

- Title (default)
- Heading 1 (default)
- Heading 4 (default)
- OV-H1toc (custom format for Heading 1, included in Table of Contents)
- OV-H2toc (custom format for Heading 2, included in Table of Contents)
- OV-Finding (custom format for finding titles, included in Table of Contents)

To modify these styles, use the style selector in your prefered document editor (e.g. MS Office Word, LibreOffice, ...). See also this issue on GitHub.

Result

The final report will look similar to this:

Todo: [DOCS] Add screenshots of Word report

Vulnerabilities are sorted according to CVSS score (descending) and vulnerability name (ascending).

Export to Comma Separated Values

When passing the –format csv parameter, the tool will export reports in Comma Separated Values (CSV) format. The CSV format is optimized for import in Excel.

Examples

Create CSV report from 1 OpenVAS XML report using default settings

```
openvasreporting -i openvasreport.xml -f csv
```

Create CSV report from multiple OpenVAS XML report using default settings

```
openvasreporting -i *.xml -f csv
# OR
openvasreporting -i openvasreport.xml -i openvasreport1.xml -i openvasreport2.xml␣
↪[-i ...] -f csv
```

Create CSV report from 1 OpenVAS XML report, reporting only severity level high and up

```
openvasreporting -i openvasreport.xml -o openvas_report -f csv -l h
```

Result

The final report will look similar to this:

Todo: [DOCS] Add examples of CSV report

Vulnerabilities are sorted according to CVSS score (descending) and vulnerability name (ascending).

Export to Excel sorted by Host

By default (or when passing the –format xlsx parameter), the tool will export reports in Excel (xlsx) format sorted by vulnerability. If you add the **–report-type host** parameter, it will generate an Excel report sorted by Host.

This report contains a summary sheet, table of contents, and a sheet per Host containing vulnerability details.

Examples

Create Excel report from 1 OpenVAS XML report, sorted by host, using default settings

```
openvasreporting -i openvasreport.xml -T host
```

Create Excel report from multiple OpenVAS XML report, sorted by host, using default settings

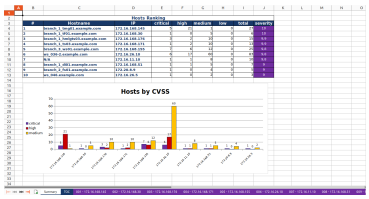
```
openvasreporting -i *.xml -T host
# OR
openvasreporting -i openvasreport.xml -i openvasreport1.xml -i openvasreport2.xml␣
↪[-i ...] -T HOST
```

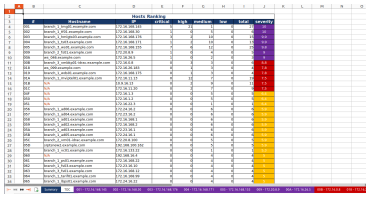
Create Excel report from 1 OpenVAS XML report, sorted by host, reporting only severity level high and up

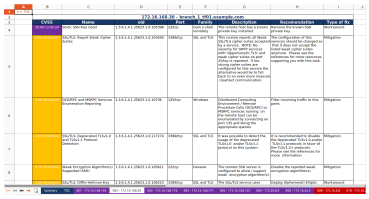
```
openvasreporting -i openvasreport.xml -o openvas_report -f xlsx -l h -T  HOST
```

Result

The final report will look similar to this:







Host Ranking and TOC is sorted according to the maximum CVSS score of a host followed by number of entries at each threat level.

Host worksheets are sorted by CVSS score followed by vulnerability name.

All worksheets are colored according to threat level.

Export to Comma Separated Values sorted by Host

When passing the –format csv parameter, the tool will export reports in Comma Separated Values (CSV) format. If you use [-T host] parameter, the list will be sorted by host.

The CSV format is optimized for import in Excel.

Examples

Create CSV report from 1 OpenVAS XML report, sorted by host, using default settings

```
openvasreporting -i openvasreport.xml -f csv -T host
```

Create CSV report from multiple OpenVAS XML report, sorted by host, using default settings

```
openvasreporting -i *.xml -f csv -T host
# OR
openvasreporting -i openvasreport.xml -i openvasreport1.xml -i openvasreport2.xml␣
↪[-i ...] -f csv -T host
```

Create CSV report from 1 OpenVAS XML report, sorted by host and reporting only severity level high and up

```
openvasreporting -i openvasreport.xml -o openvas_report -f csv -l h -T host
```

Result

The final report will look similar to this:

Todo: [DOCS] Add examples of CSV report

Hosts are sorted according to number of CVSS score in each level (descending)

1.2.3 Using Filters

You can filter the vulnerabilities that will be presented in you report using one of the filtering options. You can filter: - networks cidrs, ip ranges and any individual ip using the options **-n/–network-include** and **-N/–network-exclude**; - regex expressions that will be matched against the vulnerability names using the options **-r/–regex-include** and **-R/–regex-exclude** - The matches will be case insensitive; - CVEs numbers in the format CVEYYYY-nnn... using the options **-e/–cve-include** and **-E/–cve-exclude**; When passing the –format csv parameter, the tool will export reports in Comma Separated Values (CSV) format.

All these options receive the path to a .txt file containing one filtering option by line.

Examples

Create xlsx report from multiple OpenVAS XML Report filtering by network

```
openvasreporting -i *.xml -n ./branch_1.txt -N ./branch_1_ipaliases.txt
```

Contents of *branch_1.txt* could be:

```
172.16.168.0/24
172.16.0.1-172.16.0.3
172.16.1.1
```

Contents of *branch_1_ipaliases.txt* could be:

```
172.16.168.234
172.16.168.236-239
172.16.168.15
```

Create xlsx report, sorted by host, filtering by regex

```
openvasreporting -i *.xml -T host -R ./regex_defender.txt
```

Contents of *regex_defender.txt* could be:

```
defender
```

Create xlss report from 1 OpenVAS XML report, filtering by CVE

```
openvasreporting -i openvasreport.xml -e ./cisa_nov_2021.txt
```

Contents of *cisa_nov_2021.txt* could be

```
CVE-2021-27104
CVE-2021-27102
CVE-2021-27101
[...]
CVE-2020-10189
CVE-2019-8394
CVE-2020-29583
```

Of course, you can mix filtering options:

```
openvasreporting -i *.xml -r ./regex_defender.txt -e ./cisa_nov_2021.txt
```

1.2.4 .YML Configuration File

You can use the **-c/–config-file** option to define a .yml file to be loaded with all the configuration to execute **openvasreporting** but input and output filenames.

If you use this option, all other configuration options but input and output filenames will be ignored and if not defined in the .yml configuration file will be set to default.

Sample Configuration File:

```
level:
  medium

format:
  xlsx

reporttype:
  host

networks:
  includes:
    - 10.0.0.0/8
    - 172.16.0.0/12
    - 192.168.0.0/16
  excludes:
    - 172.16.168.234
    - 172.16.168.236-172.16.168.239
    - 172.20.16.120

regex:
  excludes:
    - defender

# I use this section to filter out recent ms patches not put in production yet. Or␣
↪to filter in CVEs from the CISA Active Exploit bulletin
cve:
  excludes:
    - CVE-2021-1971
```

Using this configuration file, the resulting report would be restricted to level medium and up, will be an Excel report, will be sorted by host, will filter in only rfc1918 local networks but will filter out some IP Ranges and IPs and will exclude CVE2021-1971.

Examples

Create report using above sample config:

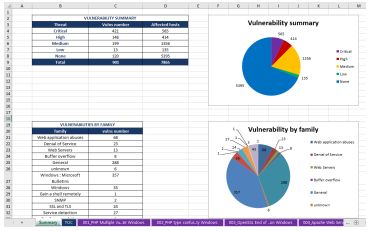
```
openvasreporting -i openvasreport.xml -c ./config_defender_out_by_host.yml
```

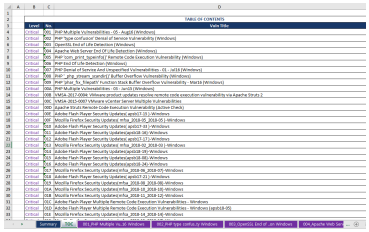
CHAPTER 2

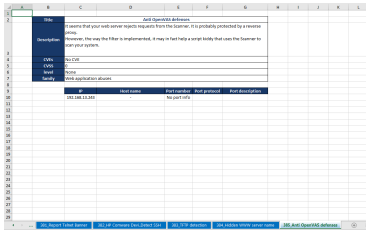
What's OpenVAS Reporting?

OpenVAS reporting allows you to create a report from one or more OpenVAS/Greenbone XML reports. This way, it's easy to create simple graphs for the compliance department, create pivot tables to collect statistics, or combine multiple scan reports into one.

The tool currently only supports exports to Excel, creating a workbook containing a summary sheet, table of contents and a worksheet per vulnerability (with info and list of hosts).







CHAPTER 3

Why create this tool?

OpenVAS is an awesome tool for many people and it's UI is nice but not always intuitive. It currently also lacks the ability to merge multiple task reports into one, especially when testing multiple environments. This tool allows you to merge multiple XML reports into one.

Working with CSV exports of the OpenVAS reports is just a pain, since they include newlines which cause Excel (or other importers) to view those lines as new inputs. This can be fixed with some find/replace voodoo, which can probably be automated as well (in tools like Notepad++), but it's just too much hassle to filter out all edge cases.

Or maybe you just prefer working in Excel because you're used to it (Excel is probably the #1 IT tool across all branches).

CHAPTER 4

Aren't there other tools to achieve this?

As a good netizen, I used some search engine terms to find tools that would fit my needs (merge reports, export to Excel, perhaps export to other formats) and found cr0hn's OpenVAS2Report. However, it appears that either the format of the XML reports has changed and the code doesn't handle this well, or my reports contain some weird voodoo, because some of my reports would convert (partially) while others wouldn't.

It was first thinking of tracking down the error and sending a pull request to fix it. However, at the time I needed a quick solution and I had no idea where to start tracing the error. So I made a fork of the project, but basically rewrote chunks of the code (copy/pasting) a lot of it as well.

CHAPTER 5

How can I help?

I am planning to maintain this tool in the future as to be able to support future changes to the OpenVAS report format. I also plan on adding more functionality as I feel the need for it, or receive requests from others.

If you feel like I'd need to implement a new feature, rewrite some code, fix a bug, ... hit me up on Twitter or file an issue on GitHub.

CHAPTER 6

TODO list

Todo: [Feature] Export to other formats (PDF, [proper] CSV)

original entry

Todo: [DOCS] Add examples of CSV report

original entry

Todo: [DOCS] Add examples of CSV report

original entry

Todo: [DOCS] Add screenshots of Word report

original entry