OpenVAS2Report Documentation

Release 1.0.0

Daniel Garcia (cr0hn) - @ggdaniel

Dec 27, 2017

Contents


OpenVAS
Open Vulnerability Assessment System
XML

CHAPTER 1

What's OpenVAS 2 Report

The idea is very simple:

1. Take an OpenVAS report, in it horrible XML format.
2. Convert it into an beautiful Excel, ready to give to your boss.

CHAPTER 2

Why?

I'm security auditor and I really hate to pass OpenVAS XML report into to and Excel document. This is a work for a monkey, not for a human! (Yes: security auditors are humans too. I know, I know. It's incredible)

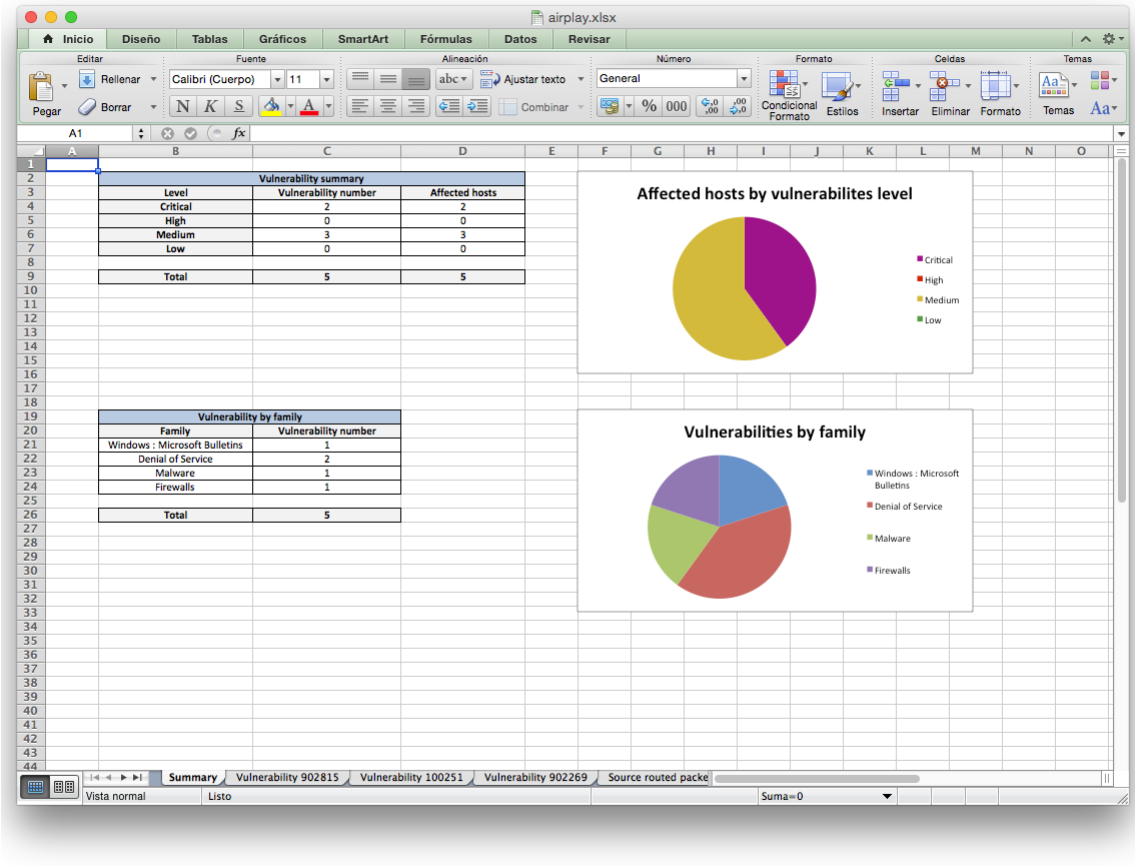
So I started to develop this project and I thought share it for help other auditors that also hate make a monkey's work.

2.1 Available tools

This package are composed by 2 tools:

- **openvas_to_document**: This is the main program. You can use it to generate the Excel file.
- **openvas_cutter**: This is a facility for filter and crop some information from OpenVAS XML report.

A picture is worth a 1000 words From XML. Using `openvas_to_document` you can obtain this Excel file:



2.2 As a library

Also, you can use the tool as a library and import them it in your own code. It has BSD license, Feel free to use!

2.3 Content Index

2.3.1 Quick start guide

The use of package is very simple. This document is a brief explanation.

Installation

Install the package is so easy. Simple type this: x .. code-block:: bash

> sudo pip install openvas-to-report

Note: Remember that you need **Python 3**!!

Generate Excel

To generate an Excel File you need to export the OpenVAS results as a XML format. If you can't a report by hand, you can find one in `example` folder.

Then, you need to use `openvas_to_document` tool:

```
> openvas_to_document -i my_openvas_report.xml -o generated_excel.xlsx
```

For further information go to the *Openvas to report manual*.

Filter results

If you want to filter the XML OpenVAS file, deleting some targets for example, and generate a new XML document without this information, you can user `openvas_cutter`

First we create a file with the targets that we want to remove.

```
> echo 10.0.1.1 > remove_targets.txt
```

Now launch the script:

```
> openvas_cutter -i my_openvas_report.xml -o my_openvas_report_filtered.xml --exclude-
↪hosts remove_targets.txt
```

For further information go to the *Openvas cutter manual*.

As a library

You also can use the library in your won code, importing as a usual lib. After install de library, using bellow instructions, you only must do:

```
from openvas_to_report.api import Config, convert

c = Config(input_files=["input_file.xml"], output_file="results.xlsx")
convert(c)
```

For further information go to the *Openvas as library manual*.

2.3.2 Openvas2Report Manual

What's this tool?

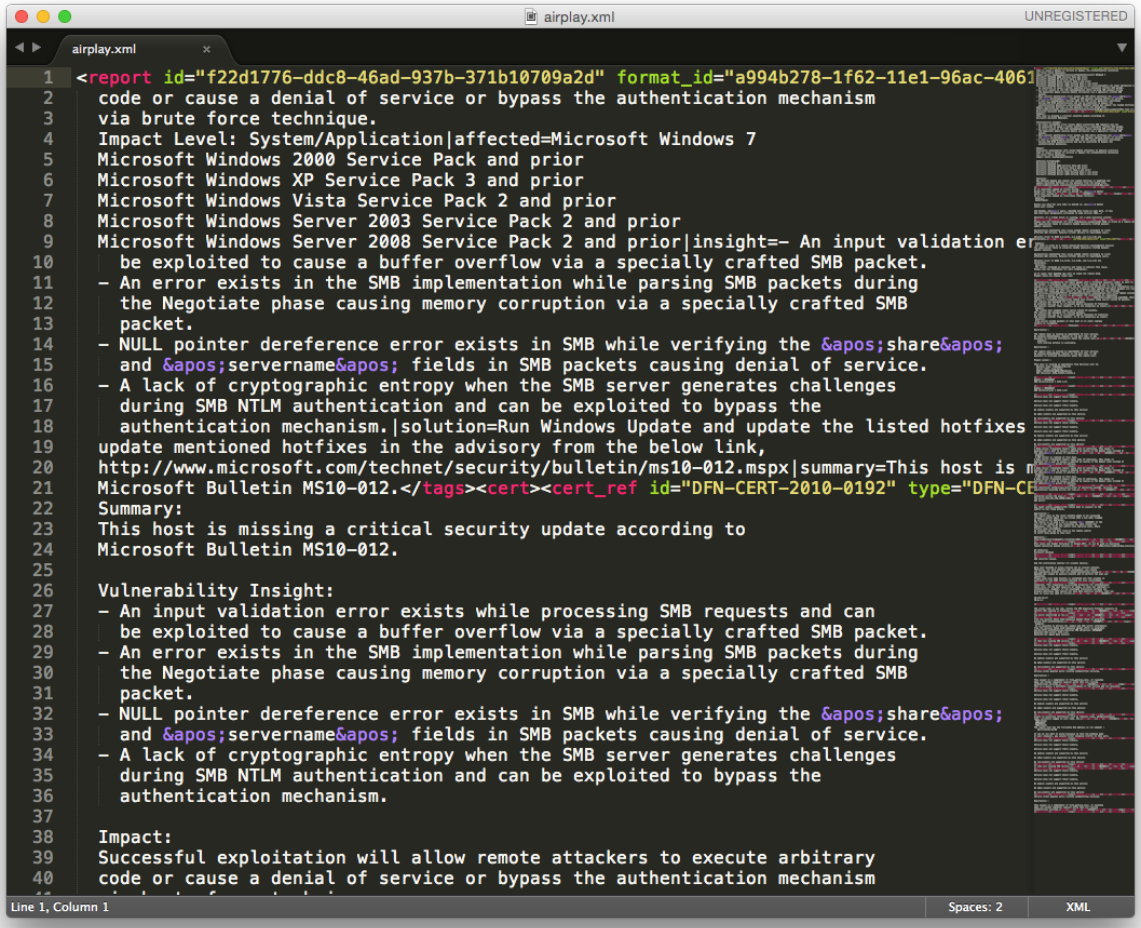
In few words: with this tools you can convert the OpenVAS XML report to beautiful Excel file.

Basic usage

The more simple usage is with only one file as input.

Note: If you haven't a XML as example, you can get one in folder `examples`.

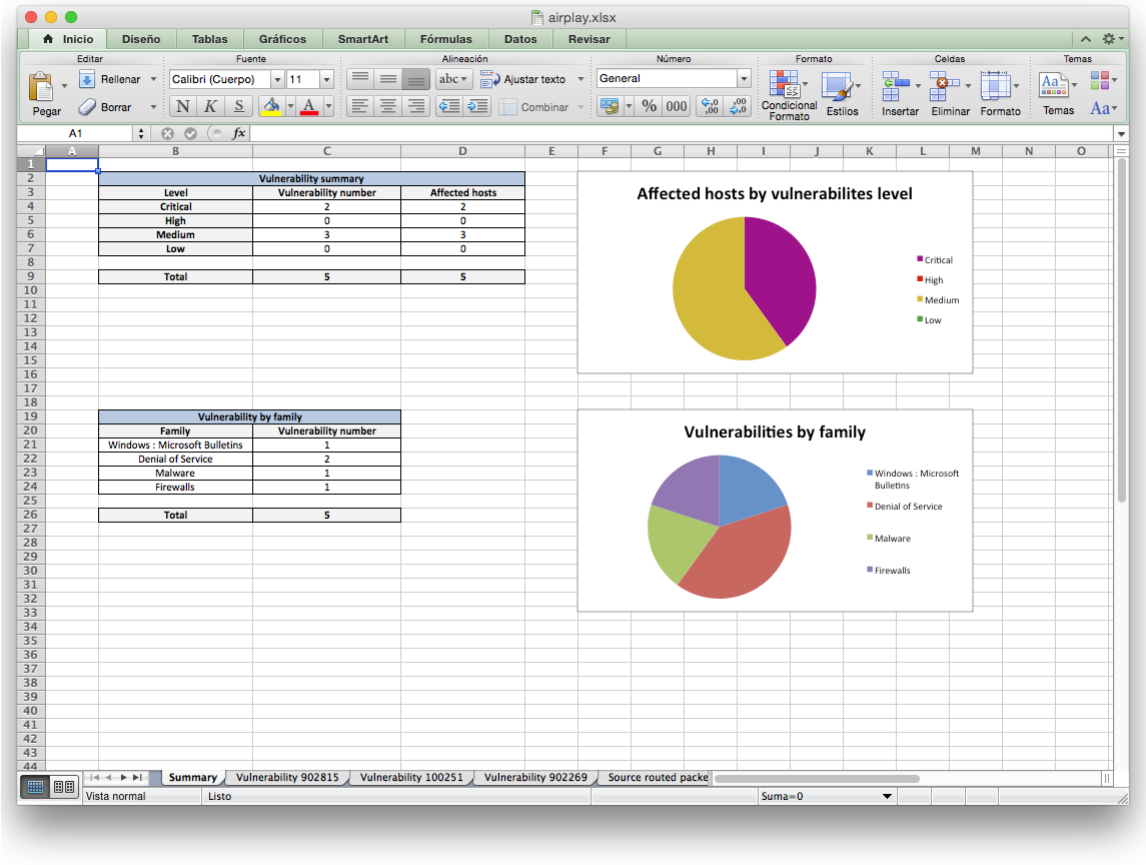
This pictures shows the example XML file:


airplay.xml
UNREGISTERED
airplay.xml
<report id="f22d1776-ddc8-46ad-937b-371b10709a2d" format_id="a994b278-1f62-11e1-96ac-4061
code or cause a denial of service or bypass the authentication mechanism
via brute force technique.
Impact Level: System/Application|affected=Microsoft Windows 7
Microsoft Windows 2000 Service Pack and prior
Microsoft Windows XP Service Pack 3 and prior
Microsoft Windows Vista Service Pack 2 and prior
Microsoft Windows Server 2003 Service Pack 2 and prior
Microsoft Windows Server 2008 Service Pack 2 and prior|insight=- An input validation e
be exploited to cause a buffer overflow via a specially crafted SMB packet.
- An error exists in the SMB implementation while parsing SMB packets during
the Negotiate phase causing memory corruption via a specially crafted SMB
packet.
- NULL pointer dereference error exists in SMB while verifying the 'share'
and 'servername' fields in SMB packets causing denial of service.
- A lack of cryptographic entropy when the SMB server generates challenges
during SMB NTLM authentication and can be exploited to bypass the
authentication mechanism.|solution=Run Windows Update and update the listed hotfixes
update mentioned hotfixes in the advisory from the below link,
http://www.microsoft.com/technet/security/bulletin/ms10-012.mspx|summary=This host is m
Microsoft Bulletin MS10-012.</tags><cert><cert_ref id="DFN-CERT-2010-0192" type="DFN-CE
Summary:
This host is missing a critical security update according to
Microsoft Bulletin MS10-012.
Vulnerability Insight:
- An input validation error exists while processing SMB requests and can
be exploited to cause a buffer overflow via a specially crafted SMB packet.
- An error exists in the SMB implementation while parsing SMB packets during
the Negotiate phase causing memory corruption via a specially crafted SMB
packet.
- NULL pointer dereference error exists in SMB while verifying the 'share'
and 'servername' fields in SMB packets causing denial of service.
- A lack of cryptographic entropy when the SMB server generates challenges
during SMB NTLM authentication and can be exploited to bypass the
authentication mechanism.
Impact:
Successful exploitation will allow remote attackers to execute arbitrary
code or cause a denial of service or bypass the authentication mechanism
Line 1, Column 1
Spaces: 2
XML

To run only write:

```
> openvas_to_document -i my_openvas_report.xml -o generated_excel.xslx
```

After running you got this Excel:



Also, you can specify more than one XML report as input:

```
> openvas_to_document-i my_openvas_report_1.xml -i my_openvas_report_2.xml -o␣
↪generated_excel.xslx
```

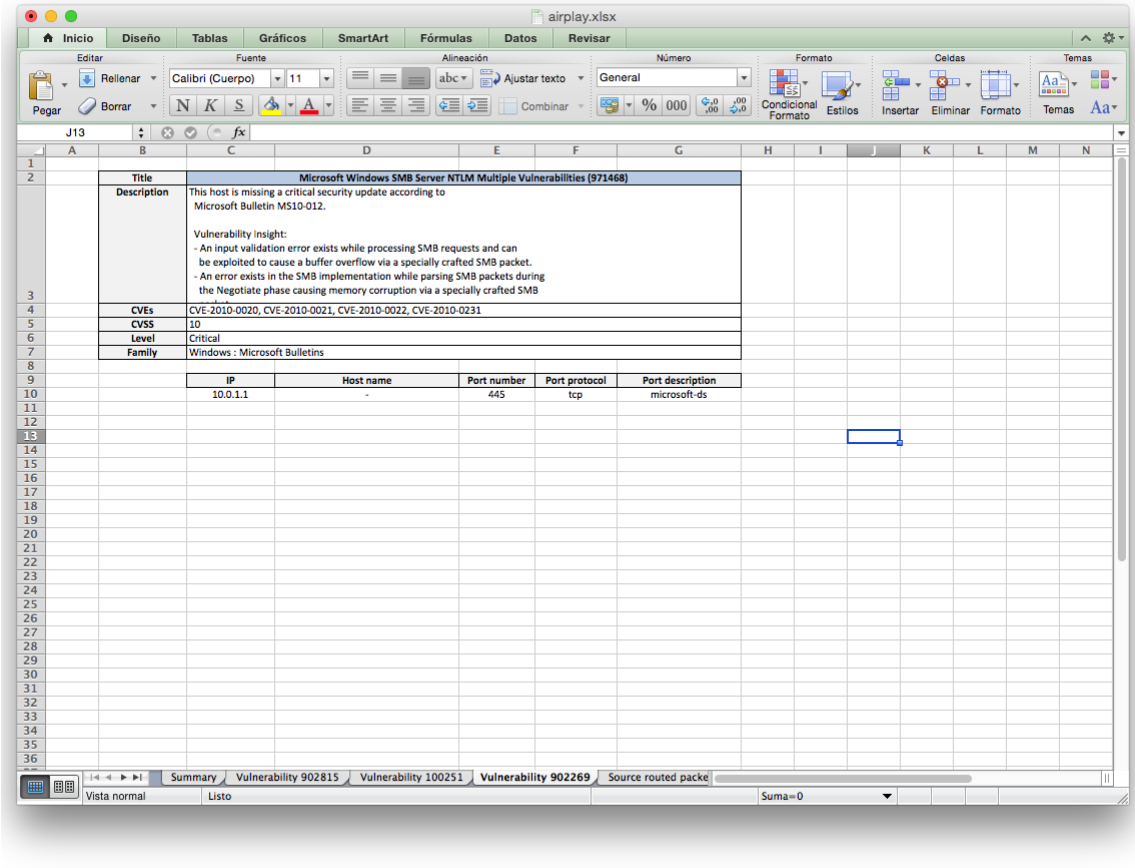
Advanced usage

If you want to exclude some hosts from report, you can use two aproaches:

1. Using a scope filter.
2. Specify a list of hosts to exclude.

Setting a filter

Is you only want to include certain hosts in your Excel report, you only must create a .txt file with your scope:



Title	Microsoft Windows SMB Server NTLM Multiple Vulnerabilities (971468)
Description	This host is missing a critical security update according to Microsoft Bulletin MS10-012. Vulnerability Insight: - An input validation error exists while processing SMB requests and can be exploited to cause a buffer overflow via a specially crafted SMB packet. - An error exists in the SMB implementation while parsing SMB packets during the Negotiate phase causing memory corruption via a specially crafted SMB
CVEs	CVE-2010-0020, CVE-2010-0021, CVE-2010-0022, CVE-2010-0231
CVSS	10
Level	Critical
Family	Windows : Microsoft Bulletins

IP	Host name	Port number	Port protocol	Port description
10.0.1.1	-	445	tcp	microsoft-ds

```
> echo 10.0.0.1 > my_scope.txt
> echo 10.0.1.23 >> my_scope.txt
```

Then use it as a parameter in the tools:

```
> openvas_to_document -i my_openvas_report.xml -o generated_excel.xslx --scope-hosts␣
↪my_scope.txt
```

Simple right? :)

Excluding hosts

The second approach is to create a black list. As in the previous case, we'll define our file:

```
> echo 127.0.0.1 > excluded.txt
> echo 192.168.0.3 >> excluded.txt
```

And then, use it:

```
> openvas_to_document -i my_openvas_report.xml -o generated_excel.xslx --exclude-
↪hosts excluded.txt
```

2.3.3 Openvas Cutter

What's this tool?

There are some times that you need to exclude some hosts from the XML. For example: If you must to deliver the XML report, but it contains some hosts that are out of scope.

This tools can help us. It remove host information from the original XML file, and generates a new XML file without this information.

Basic usage

The usage si very simple. Only need to specify the input file/s and the new output file:

Note: If you haven't a XML as example, you can get one in folder `examples`.

To run only write:

```
> openvas_cutter -i my_openvas_report.xml -o my_openvas_report_filtered.xml
```

Advanced usage

Advanced filer works as same as `openvas_to_document` tool. You can read it in: *Advanced usage*.

2.3.4 OpenVAS2Report as a library

You can user openvas2Report as a library. It's easy.

Configuration object

All the actions in package has a common configuration object called `Config`. We need to configure it before to run.

This code display the Config objects and mark the parameters accepted:

```
# --------------------------------------------------------------------------
class Config(object):
    """Program configuration"""

    # --------------------------------------------------------------------------
    def __init__(self, input_files, output_file, template=None, lang="en",␣
↪excluded=None, scope=None):
        """
        :param input_files: input file path
        :type input_files: list(str)

        :param output_file: output file path and name
        :type output_file: str

        :param template: path to template
        :type template: str

        :param lang: language
        :type lang: str

        :param excluded: path to file with excluded hosts.
        :type excluded: str

        :param scope: path to file with scope hosts
        :type scope: str

        :raises: TypeError, ValueError
```

The code is auto-explained. Then, we import them from `openvas_to_report.api`:

```
from openvas_to_report.api import Config

config = Config(["openvas_report1.xml", "openvas_report2.xml"],
                "results.xslx",
                "en",
                "excluded_hosts.txt",
                "scope_host.txt")
```

Run actions

I called action to these tasks or functions that you also can run in command line way.

After instance the config object, we can call actions:

```
from openvas_to_report.api import Config, convert, crop

# Convert to Excel
config_convert = Config(["openvas_report1.xml", "openvas_report2.xml"],
                        "results.xslx",
                        "en")

convert(config)
```

```
# Crop XML file
config_convert = Config(["openvas_report1.xml", "openvas_report2.xml"],
                        "results_filtered.xml")

crop(config)
```