README MIT license

Linuxprivchecker.py

A Linux Privilege Escalation Check Script

Orginal Author: Mike Czumak (T_v3rn1x) -- @SecuritySift

Current Maintainer: Michael Contino (@Sleventyeleven)

This script is intended to be executed locally on a Linux box to enumerate basic system info and search for common privilege escalation vectors such as world writable files, misconfigurations, clear-text passwords and applicable exploits.

Linuxprivchecker is designed to identify potential areas to investigate further, not provide direct action or exploitation. This is to help users further learn how these privilege escalations work and keep it in line with the rules, for self directed exploitation, laid out for the OSCP, HTB, and other CTFs/exams.

We will try our best to addtional information and reference where possible. As the current Maintainer, I also plan to accompany new feature adds, with a post on my blog (hackersvanguard.com) to further explain each potential area for privilege escalation and what criteria may be required.

Running on Legacy Python 2.6/2.7 System

To run on legacy python >2.6 systems just get the all in one python script and run it.

```
wget https://raw.githubusercontent.com/sleventyeleven/linuxprivchecker/master/linuxprivchecker.py
```

```
python linuxprivchecker.py -w -o linuxprivchecker.log
```

Running on Current Python 3.X System (Beta)


sleventyeleven Michael Contino


ankh2054 Charles Holtzkampf


n3k00n3 n3k00n3


jtpereyda Joshua Pereyda

Languages

- **Python** 100.0%

Right now Linuxprivchecker for python 3.X should be considered a stable beta versions. Issues can happen with the script and it certainly can miss possible vulnerabilities (open an issue or PR).

To run the python 3 version, just utilize pip.

```
pip install linuxprivchecker
```

Then just run via commandline if runpy is available.

```
linuxprivchecker -w -o linuxprivchecker.log
```

or if runpy fails to add the script to your path

```
python3 -m linuxprivchecker -w -o linuxprivchecker.log
```

Command Options and arguments

If the system your testing has Python 2.6 or high and/or argparser installed, you can utilize the following options. If importing argparser does not work, all checks will be run and no log file will be written. However, you can still use terminal redirection to create a log, such as 'python linuxprivchecker.py > linuxprivchecker.log.'

usage: linuxprivchecker.py [-h] [-s] [-w] [-o OUTFILE]

Try to gather system information and find likely exploits

optional arguments: -h, --help show this help message and exit

-s, --searches Skip time consumming or resource intensive searches

-w, --write Wether to write a log file, can be used with -0 to specify name/location

-o OUTFILE, --outfile OUTFILE The file to write results (needs to be writable for current user)

Warning

This script comes as-is with no promise of functionality or accuracy. I have no plans to maintain updates, I did not write it to be efficient and in some cases you may find the functions may not produce the desired results. For example, the function that links packages to running processes is based on keywords and will not always be accurate. Also, the exploit list included in this function will need to be updated over time. Feel free to change or improve it any way you see fit.

Modification, Distribution, and Attribution

You are free to modify and/or distribute this script as you wish. I only ask that you maintain original author attribution and not attempt to sell it or incorporate it into any commercial offering (as if it's worth anything anyway :)