Auditoria de Segurança
Trabalho Prático

Pipeline Automatizada de Pentesting

João Almeida, 2016225010
José Donato, 2016225043

Motivação

“Using many tools and techniques ... attempts to exploit ... systems” [PP de Pentesting]

- Processo caro e extenso
  - Grande variedade de ferramentas, com diversos propósitos e aplicadas em fases diferentes
- Comunicação dos resultados fundamental
  - expôr situação e motivar/guiar mudança
  - diferentes níveis de detalhe necessários para cada entidade envolvida

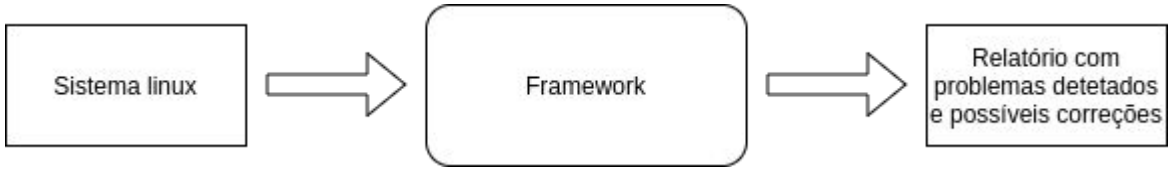
Ideia

Criar Pipeline Automatizada de Pentesting a partir de ferramentas existentes

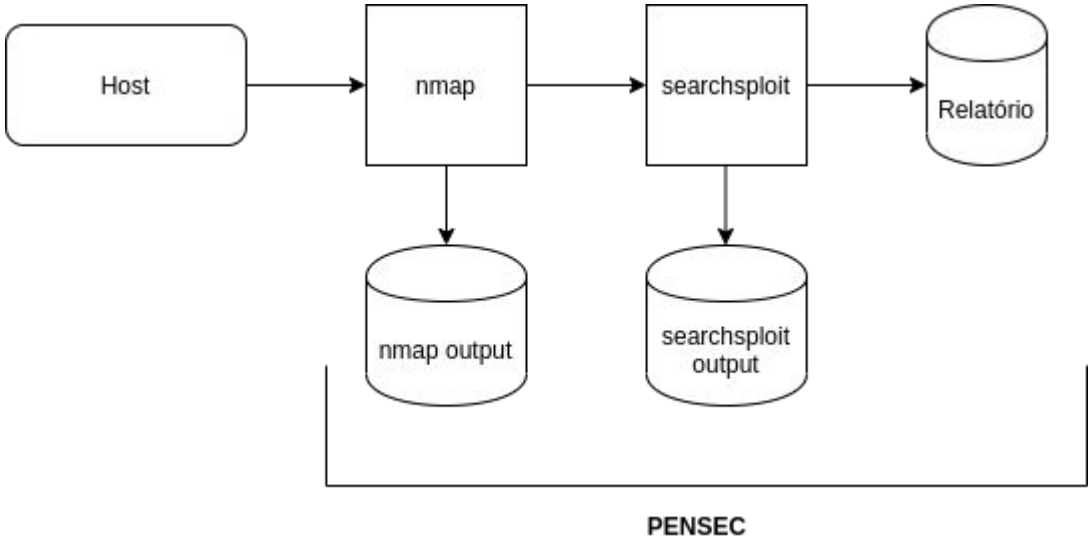
- *“Don’t reinvent the wheel”*

Plano de Trabalho:

1. Analisar ferramentas existentes
2. Explorar interdependências e sinergias (cruzar as ferramentas)
3. Criação da nova framework (Pipeline Automatizada)
    a. **Essencial:** processamento dos outputs para inclusão num relatório (com base em templates)



Proof of Concept



Configurações

- Especificam ferramentas e suas opções (*scope do pentest*)

- Ficheiros de texto reutilizáveis
- Modificáveis na aplicação
- Sincronizados para uso futuro

```
OUTPUT_DIRECTORY = 'output'
OUTPUT_TO_CONSOLE = True

# TOOL_iota = tool;options
TOOL_1 = 'Nmap;-sV'
TOOL_2 = 'Searchsploit;'
```

Pipeline

Cruzar as ferramentas identificadas.

Interdependências:

- Que informação é esperada como **input** por cada ferramenta?
- E quais a fornecem como **output?**
- **Essencial:** flexibilidade do pipeline
  - Saber se **correr** ferramenta (não instalada / dependências não satisfeitas)

```
f"Skipping tool {tool} in configuration (Missing)") for tool in missing

f"Tool {t.__class__.__name__} requires {r.value} (provided by: {', '.join(tools_providing)})"
```

  - Saber **quando** correr ferramenta (*ordenamento topológico*)

Arquitetura - Pipeline

```
for tool in sorted_tools:
    out, err = tool.run(outputs)
    for p in tool.PROVIDES:
        outputs[p] = out

    if err and not tool.IGNORE_STDERR:
        self.logger.error(err.decode('ascii'))
    else:
        self.logger.info("Output saved")
        report_obj = tool.report(reports)
        for p in tool.PROVIDES:
            reports[p] = report_obj
self.create_report(reports, sorted_tools)
```

```
def check_dependencies(self):

def missing_tool_dependencies(tools):

def sortby_dependencies(tools):
```

Arquitetura - Ferramentas

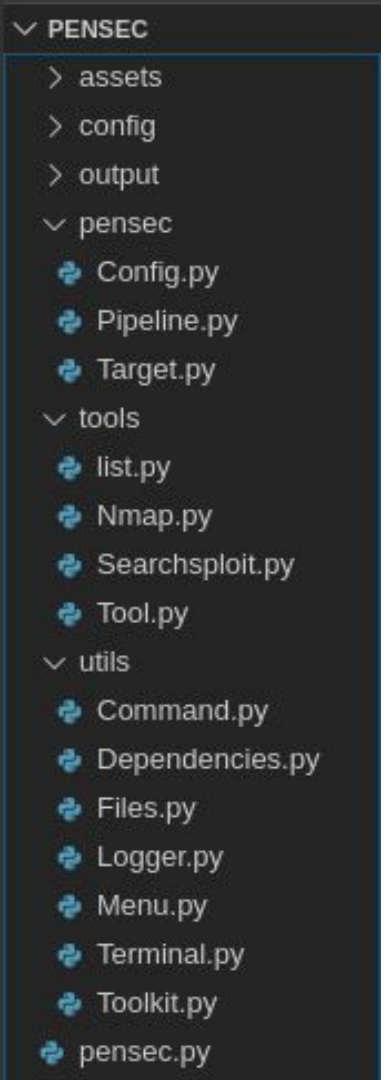
```
class Nmap(Tool):
    PROGRAM = "nmap"
    OPTIONS = Tool.Options([
        Tool.Option("Detect Versions (default)", "-sV"),
        Tool.Option("Detect Versions and Vulnerabilities",
                    "-sV --script nmap-vulners,vulscan --script-args vulscandb=scipvuldb.csv"),
    ])
    OPTIONS_PROMPT = OPTIONS.prompt()
    REQUIRES = []
    PROVIDES = [Tool.Dependencies.NMAP_SERVICES]


    def run(self, dependencies):

    def report(self, reports):

    def write_report_summary(self, reportfile, reports):

    def write_report(self, reportfile, reports):
```



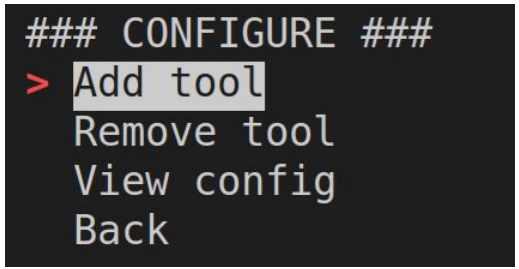
Estrutura

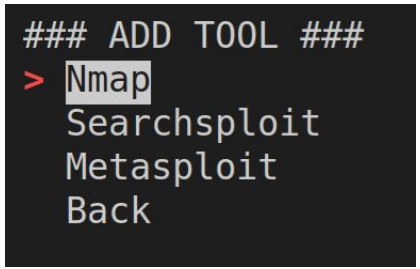
```
$ pipenv run sudo python pensec.py
```

```
### PENSEC ###
> Configure
  Run
  Save
  Exit
```

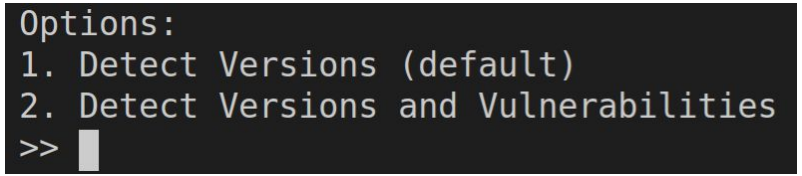
Gerir ferramentas

- Adicionar ferramentas (atualmente apenas nmap e searchsploit)
- Remover ferramentas
- Gravar configurações para carregarem na próxima execução

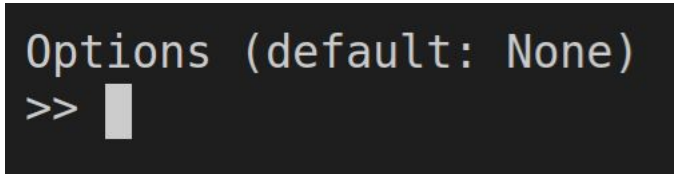




Adicionar Nmap



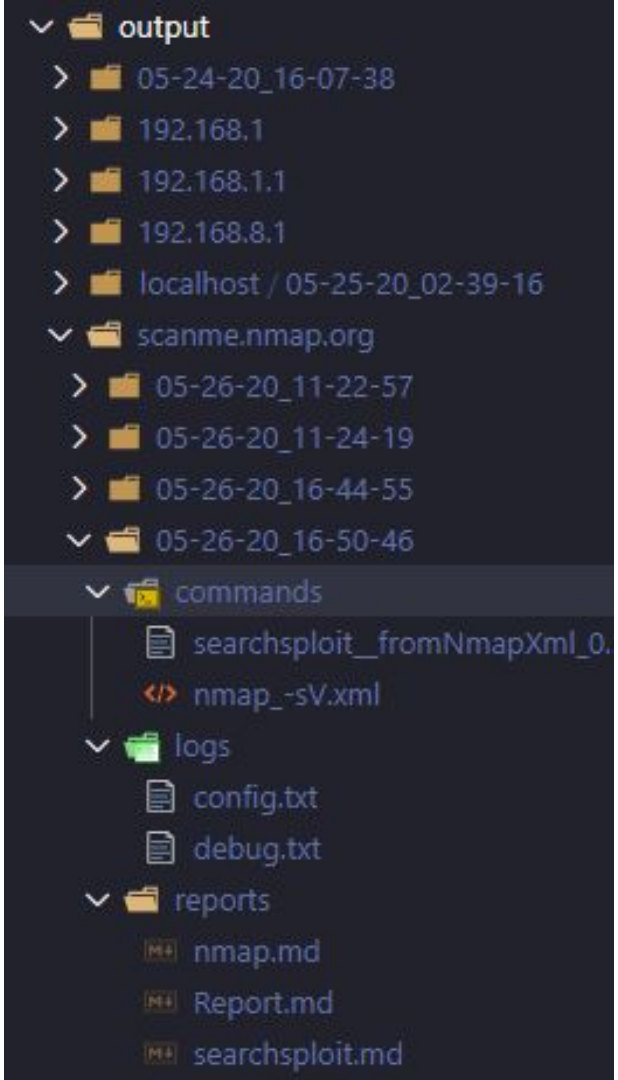
Adicionar Searchsploit



Outputs

Três diferentes outputs:

- Output dos comandos
- Logging da aplicação pensec
- **Relatórios**



Relatórios

- Dois tipos:
  - Sumário Executivo: expor situação aos superiores e motivar mudança
  - Detalhes técnicos: para corrigir problemas encontrados

- Formato: Markdown

```python
def create_report(self, reports, sorted_tools):
    outfile = f"{self.outdir}/reports/Report.md"
    title = f"PENSEC - Report of {self.target.hostname}"
    reportfile = MdUtils(file_name=outfile, title=title)

    # "Execute Summary"
    reportfile.new_header(level=3, title="Common Statistics")
    for tool in sorted_tools:
        tool.write_report_summary(reportfile, reports)
    # "Technical Details"
    for tool in sorted_tools:
        tool.write_report(reportfile, reports)

    reportfile.create_md_file()
    self.logger.info("Report saved in "+outfile)
```

PENSEC - Report of scanme.nmap.org

Common Statistics

- 5 open ports found
- 27 exploits found

Services

name	product	version	cpe	portid
ssh	OpenSSH	6.6.1p1 Ubuntu 2ubuntu2.13	cpe:/a:openbsd:openssh:6.6.1p1,cpe:/o:linux:linux_kernel	22
http	Apache httpd	2.4.7	cpe:/a:apache:http_server:2.4.7	80
nping-echo	Nping echo			9929
unknown				27355
tcpwrapped				31337

Exploits

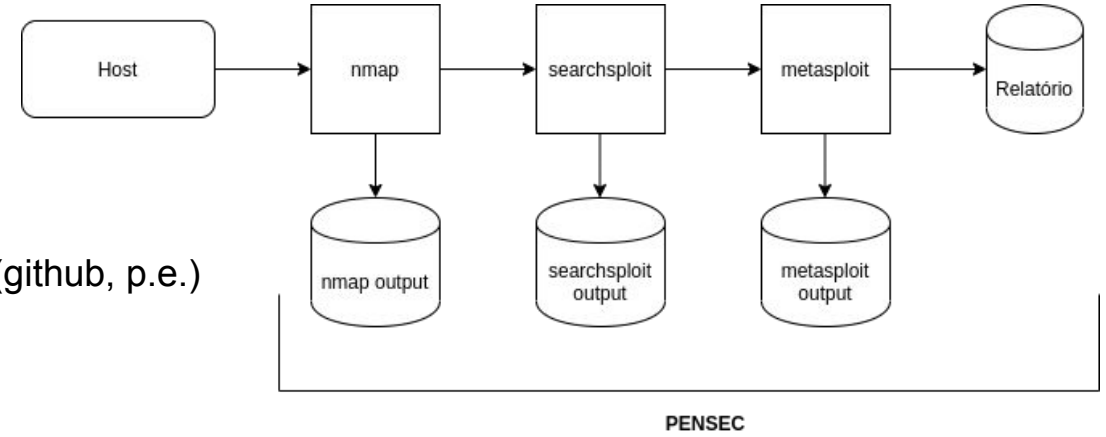
openssh

remote

- Debian OpenSSH - (Authenticated) Remote SELinux Privilege Escalation (/usr/share/exploitdb/exploits/linux/remote/6094.txt)
- FreeBSD OpenSSH 3.5p1 - Remote Command Execution (/usr/share/exploitdb/exploits/freebsd/remote/17462.txt)
- OpenSSH 1.2 - '.scp' File Create/Overwrite (/usr/share/exploitdb/exploits/linux/remote/20253.sh)
- OpenSSH 2.3 < 7.7 - Username Enumeration (PoC) (/usr/share/exploitdb/exploits/linux/remote/45210.py)

Future Work

- Adicionar novas ferramentas
- Melhorar o relatório
- Integrar com um sistema de versões (github, p.e.)
- Tornar o projeto open-source



Contribuições

- **Configurações** reutilizáveis permitem a alguém com pouco conhecimento realizar pentests
- **Composição** das ferramentas, considerando as suas interdependências, torna o processo flexível e escalável (*poderoso*)
- **Relatórios** fazem com que o processo seja realmente útil, permitindo extração de informação

Exemplo

- Criação de configuração que permitisse verificar suscetibilidade de dispositivos à botnet da Mirai
  - portátil, facilmente distribuída e verificada
  - à base de ferramentas existentes (*“Don’t reinvent the wheel”)*
  - relatório com resultados da análise e medidas a tomar pelo dono

Mapeamento com o PTES

- **Pre-engagement:** configuração específica por parte do utilizador
  - definir *scope* e objetivos (eg. scan activo/passivo)
  - setup da ferramenta e parametrização
- **Security Assessment**:
  - executado pelo nosso pipeline, correspondendo às restantes secções do PTES.
  - exploração do sistema (*Intelligence Gathering*)
  - procurar exploits (*Vulnerability Analysis* + *(Optional) Exploitation)*
- **Reporting:** criar template com resultados
  - *Executive summary*: alto nível, motivar a correção
  - *Technical Report:* baixo nível, guiar a correção

- Pre-engagement Interactions
- Intelligence Gathering
- Threat Modeling
- Vulnerability Analysis
- Exploitation
- Post Exploitation
- Reporting

Referências

- PTES: http://www.pentest-standard.org
- Nmap: https://nmap.org/book/man.html
- Searchsploit: https://www.exploit-db.com/searchsploit
- NSE scripts:
  - https://null-byte.wonderhowto.com/how-to/easily-detect-cves-with-nmap-scripts-0181925/
  - https://github.com/vulnersCom/nmap-vulners
  - https://github.com/scipag/vulscan

Organização

João Almeida	**José Donato**
Configuração	Ferramentas
Pipeline	Relatórios

Demo