Cibersegurança

M4 - PRÁTICA – 4

New Technology School

Tokio.

INDICE

- Introdução……..……pág. 3
- Cenário Alvo..…….....…….……pág. 4
- Tema 1 (OSINT)…….....…….....pág. 5 – 6
- Tema 2 (análise forense)……..…….pág. 7 – 8
- Tema 3 (cracking)……..pág. 9 – 11
- Tema 4 (auditoria)…….....….pág. 12 – 14
- Tema 5 (anonimização)…….……..….pág. 15 – 18
- Conclusão…….....pág. 19

Introdução

No âmbito do Módulo 4 do curso de especialização em Cibersegurança, foi proposto o desenvolvimento de uma prática baseada num cenário hipotético, com o objetivo de aplicar ferramentas do Kali Linux relacionadas com OSINT, Análise Forense, Cracking, Auditoria e Anonimização.

Este relatório descreve a atuação de um profissional de segurança cibernética com perfil *White Hat*, num contexto simulado de investigação e mitigação de um incidente de segurança informática.

Cenário Alvo

A empresa (fictícia) **Data4Safe**, prestadora de serviços de armazenamento em nuvem, registou atividade invulgar no seu servidor interno de testes. Um dos colaboradores reportou lentidão na rede durante o horário noturno, e o sistema de firewall sinalizou ligações externas suspeitas através da porta SSH (22).

O departamento de IT acionou o analista de cibersegurança para investigar o incidente.

De maneira a resolver o incidente, o analista seccionou a sua missão da seguinte forma:

- Recolher informações públicas relacionadas com a empresa e possíveis ameaças (OSINT);
- Analisar tráfego de rede capturado durante o período suspeito (Análise Forense);
- Verificar se existem contas com credenciais fracas ou comprometidas (Cracking);
- Auditar o servidor para identificar potenciais vulnerabilidades (Auditoria);
- Assegurar que o processo de investigação decorre de forma anónima e segura (Anonimização).

*-De salientar que para a simulação/resolução desta tarefa, será feito em ambiente de testes simulado utilizando o **Kali Linux** como máquina de ataque e o **Metasploitable2** como máquina alvo, respeitando as boas práticas de ética e legalidade em cibersegurança.-*

Tema 1: OSINT (com theHarvester)

Descrição:
A ferramenta **theHarvester** é usada para recolher informação pública (**OSINT** – Open Source Intelligence) disponível na internet sobre um domínio-alvo, como e-mails, sub-domínios, IPs e nomes de hosts. Esta etapa é fundamental em investigações de segurança, especialmente na fase inicial de reconhecimento.

Cenário:
A simulação foi realizada com o domínio data4safe.com, uma empresa fictícia da área financeira, no contexto de uma investigação de um possível ataque interno ou falha de segurança.

Comando utilizado:

theHarvester -d data4safe.com -b bing

- Inicialmente foi executado o comando com a referência *all* seguida do comando *-b*, no entanto surgiram alguns erros, e optou-se pelo motor de busca *bing,* por estar disponível sem necessidade de API key, ao contrário de outras fontes. –

Fig1. – Execução de comando

Resultados:

- Nenhum endereço IP ou e-mail foi encontrado.
- Foram identificados dois subdomínios:
  - facturador.data4safe.com
  - leonardo.data4safe.com

Estes subdomínios podem representar serviços internos expostos à internet e podem ser analisados noutras fases da investigação (como por exemplo, fases de scanning, finger-printing ou fuzzing).

Justificação da ferramenta:
A escolha do **theHarvester** justifica-se pelo seu uso amplamente aceite em processos de footprinting e reconhecimento passivo. Não compromete sistemas e fornece uma base segura para iniciar uma análise mais aprofundada.

Nota pessoal:
Achei interessante como a ferramenta permite reunir rapidamente informação útil sem interação ativa com os sistemas, evitando deteção. A sua simplicidade torna-a uma boa opção para o início de qualquer investigação.

Dificuldades encontradas:
Alguns motores exigem API keys, o que limitou o número de fontes possíveis nesta simulação. No entanto, isso reforça a importância de conhecer as alternativas gratuitas disponíveis no Kali Linux.

Tema 2: Análise Forense com tráfego SSH

Descrição:

Nesta etapa, foi realizada uma análise forense de tráfego de rede para investigar atividades suspeitas reportadas fora do horário laboral. O foco esteve na identificação de comunicações via protocolo SSH (porta 22), associadas a potenciais acessos não autorizados. Utilizaram-se as ferramentas *tcpdump* (para captura de tráfego) e *Wireshark* (para análise detalhada).

A análise forense de tráfego é uma componente essencial em investigações de incidentes, permitindo detetar comportamentos anómalos, possíveis ataques e identificar origens de comunicações suspeitas.

Cenário:

De acordo com o alerta inicial do departamento de IT da empresa **Data4Safe**, foram registadas ligações externas através da porta 22 (SSH) durante o período noturno, acompanhadas de queixas de lentidão na rede. A análise visou confirmar essas ligações e recolher evidências técnicas do tráfego capturado.

Captura de tráfego:

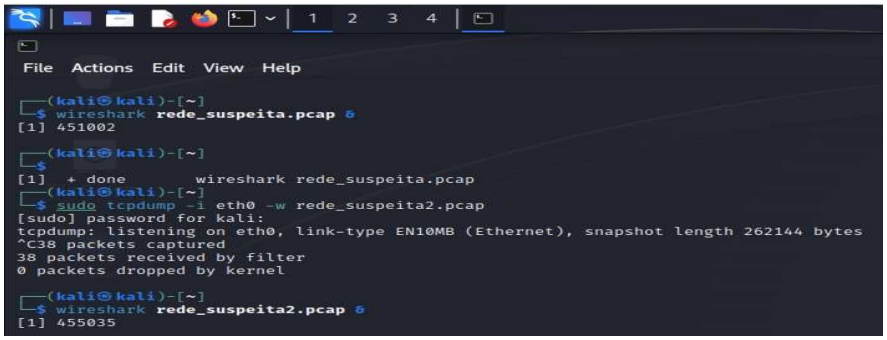

Fig.2 – Comando utilizado para captura de tráfego

Na **Fig.2,** o comando "sudo tcpdump -i eth0 -w rede_suspeita2.pcap" foi utilizado para capturar todo o tráfego da interface *eth0*, gerando um ficheiro .pcap com os pacotes recolhidos durante um curto período de tempo.

Geração de tráfego SSH (simulação):
Durante a captura, foi iniciada uma sessão SSH a partir da máquina Kali para a máquina de testes Metasploitable2, gerando comunicações reais na porta 22.

Análise no Wireshark:
O ficheiro rede_suspeita2.pcap foi aberto no Wireshark e filtrado com **tcp.port == 22,** conforme imagem **Fig.3.**

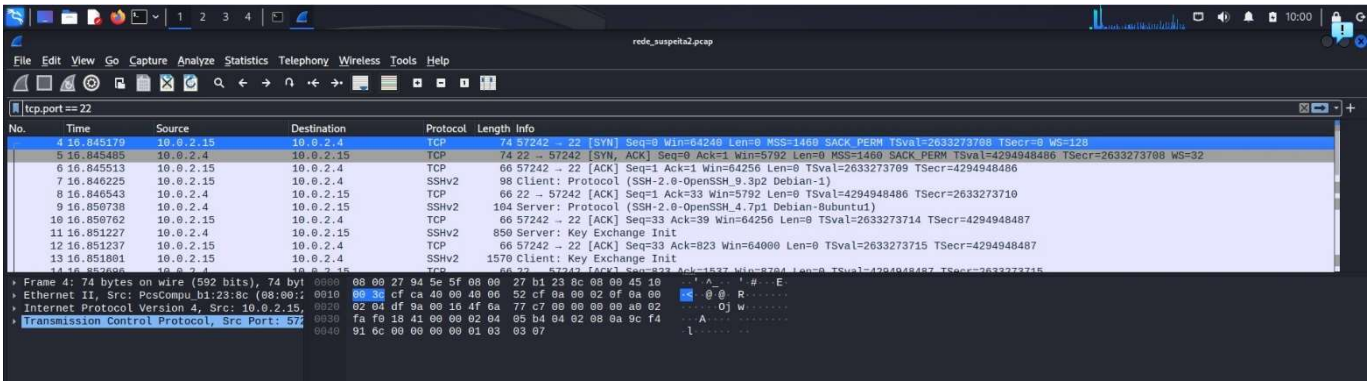

Fig.3 – Captura de trafego analisada com Wireshark.

Resultados:

- Foram capturados **32 pacotes** associados a comunicações SSH entre os IPs 10.0.2.15 (Kali) e 10.0.2.4 (Metasploitable).
- A comunicação ocorreu durante o horário fora do expediente, alinhando-se com a janela temporal descrita no alerta interno.
- O tráfego mostrou o início de uma sessão SSH com múltiplos pacotes de estabelecimento da ligação (SYN, ACK, etc.), confirmando a atividade remota.

Justificação da ferramenta:

- **Tcpdump** foi utilizado por ser uma ferramenta leve e eficiente para capturas rápidas e controladas, ideal para ambientes CLI.
- **Wireshark** oferece uma interface gráfica poderosa para inspeção detalhada de protocolos, útil para examinar sessões e comportamentos anómalos em camadas superiores.

Nota pessoal:

Gostei da complementaridade entre as ferramentas: o tcpdump simplifica a captura, enquanto o o Wireshark permite visualizar as comunicações com detalhe. Foi interessante perceber como pequenas sessões SSH geram vários pacotes — o que pode ser útil para detetar tentativas automatizadas ou acessos não autorizados com base apenas no volume de pacotes.

Dificuldades encontradas:

A primeira tentativa de captura resultou em apenas 4 pacotes, sem tráfego SSH visível. Foi necessário gerar tráfego manualmente (via SSH real), o que reflete um desafio comum em simulações — garantir que os dados representem uma situação realista.

Tema 3: Cracking de Credenciais SSH (com Hydra)

Descrição:

O objetivo desta fase foi testar a robustez das credenciais SSH do servidor comprometido, avaliando a existência de contas com palavras-passe fracas ou previsíveis. Para isso, recorreu-se à ferramenta **Hydra**, uma aplicação amplamente usada em auditorias de segurança para ataques de força bruta contra serviços autenticáveis. Foi utilizada também a ferramenta **Ncrack.**

Esta prática é comum quando há suspeitas de intrusão, para verificar se o atacante poderá ter obtido acesso devido a más práticas de segurança, como senhas fracas.

Cenário:

Dando seguimento à análise forense, e tendo em conta as tentativas de ligação identificadas à porta 22, decidiu-se testar a resistência das contas SSH do servidor 10.0.2.4 (Metasploitable), simulando o comportamento de um possível atacante.

1. Tentativa com Hydra

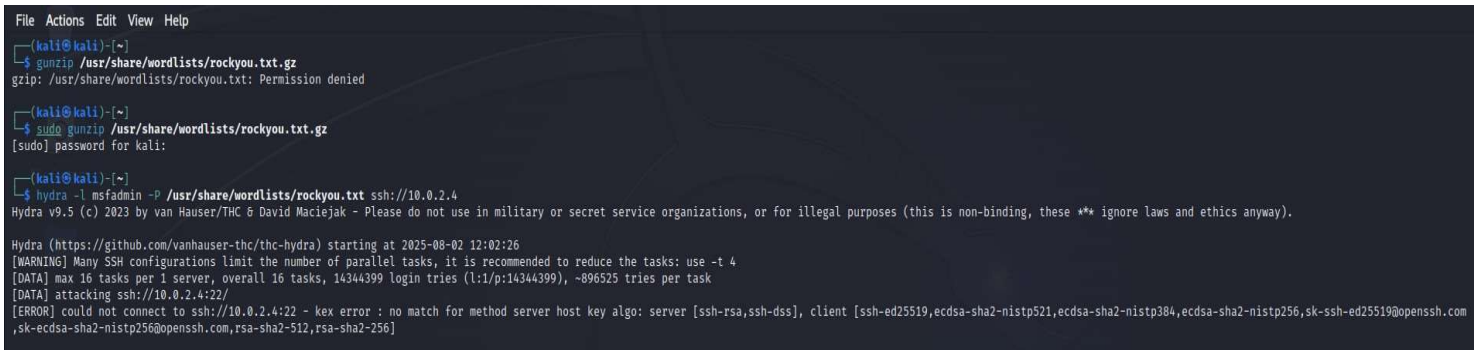

Fig.4 – Teste de penetração com Hydra

Resultado:

Após ter sido efetuada tentativa de descoberta com o Hydra, culminou em falha porque, por padrão, esta ferramenta utiliza algoritmos de encriptação modernos incompatíveis com o servidor SSH antigo do Metasploitable, impedindo o início da ligação.

2. Tentativa com Ncrack

A primeira tentativa foi feita com a wordlist extensa (rockyou.txt):

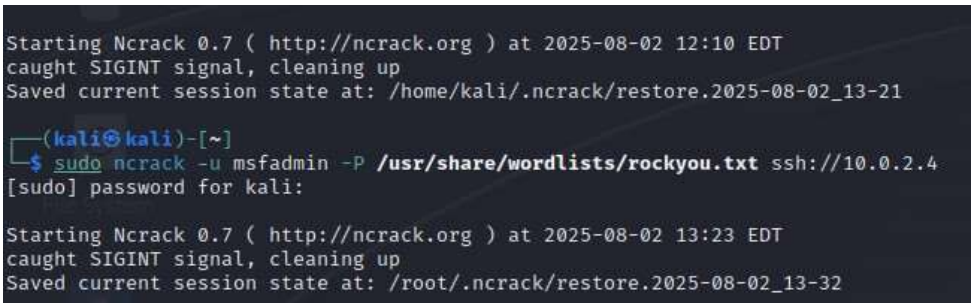

Fig. 5 – 1ª Tentativa com Ncrack

No comando utilizado, podemos fazer uma breve descrição:

-u msfadmin define o nome de utilizador a testar

-P /usr/share/wordlists/rockyou.txt é o caminho para o ficheiro de palavras-passe usado para testar combinações

Ssh://10.0.2.4 especifica o serviço (SSH) e o IP da máquina alvo

Esta tentativa não produziu resultados visíveis de imediato devido ao elevado número de combinações a testar (milhões), o que provocou lentidão extrema na execução sem feedback.

Assim foi criada um ficheiro **test.txt** com apenas a palavra msfadmin (através do comando **echo**), que servirá de wordlist personalizada e reduzida. Permitindo assim testar diretamente uma credencial especifica, agilizando o processo.

Fig. 6 – 2ª tentativa com Ncrack, com sucesso (wordlist reduzida)

Resultados:

Confirmou-se que o utilizador **msfadmin** estava acessível via SSH com a palavra-passe padrão.

Este tipo de configuração representa uma **falha crítica de segurança**, comum em ambientes de teste mal configurados ou esquecidos.

Justificação das ferramentas:

- O **Hydra** é normalmente indicado para brute force por ser altamente configurável, mas foi incapaz de estabelecer ligação devido à incompatibilidade de algoritmos com o servidor legado.
- O **Ncrack**, por sua vez, revelou-se mais adaptável. Embora inicialmente tenha demonstrado lentidão com uma wordlist extensa, funcionou eficazmente quando ajustado com uma lista reduzida e objetiva.

Nota pessoal:

Foi uma experiência interessante perceber como o tamanho da wordlist pode afetar o desempenho e o feedback de ferramentas como o ncrack. A necessidade de ajustar a abordagem durante o processo refletiu um cenário real de investigação, onde o analista precisa de se adaptar a limitações técnicas e tomar decisões práticas.

Dificuldades encontradas:

A incompatibilidade do Hydra com o servidor SSH antigo impediu a ligação.

O uso de uma wordlist pesada com o Ncrack gerou atrasos e ausência de resposta.

A criação de uma wordlist reduzida permitiu contornar o problema de forma eficaz.

Tema 4: Auditoria e Vulnerabilidades

(com Nmap e Searchsploit)

Descrição:

O objetivo desta fase foi realizar uma auditoria ao servidor identificado como vulnerável (IP: 10.0.2.4), de forma a detetar serviços expostos, versões desatualizadas e vulnerabilidades conhecidas associadas. Foram utilizadas as ferramentas **Nmap** (para deteção de serviços e versões) e **Searchsploit** (para cruzar os dados com vulnerabilidades conhecidas).

Cenário:

Após a análise forense e confirmação da presença de contas vulneráveis via SSH, tornou-se necessário identificar outros vetores de ataque que pudessem estar a ser explorados no sistema alvo, especialmente tendo em conta que o Metasploitable simula um ambiente com múltiplas vulnerabilidades propositadamente deixadas ativas.

Atendendo a continuação de cenário, foi feita uma varredura com Nmap, para deteção de serviços e versões:

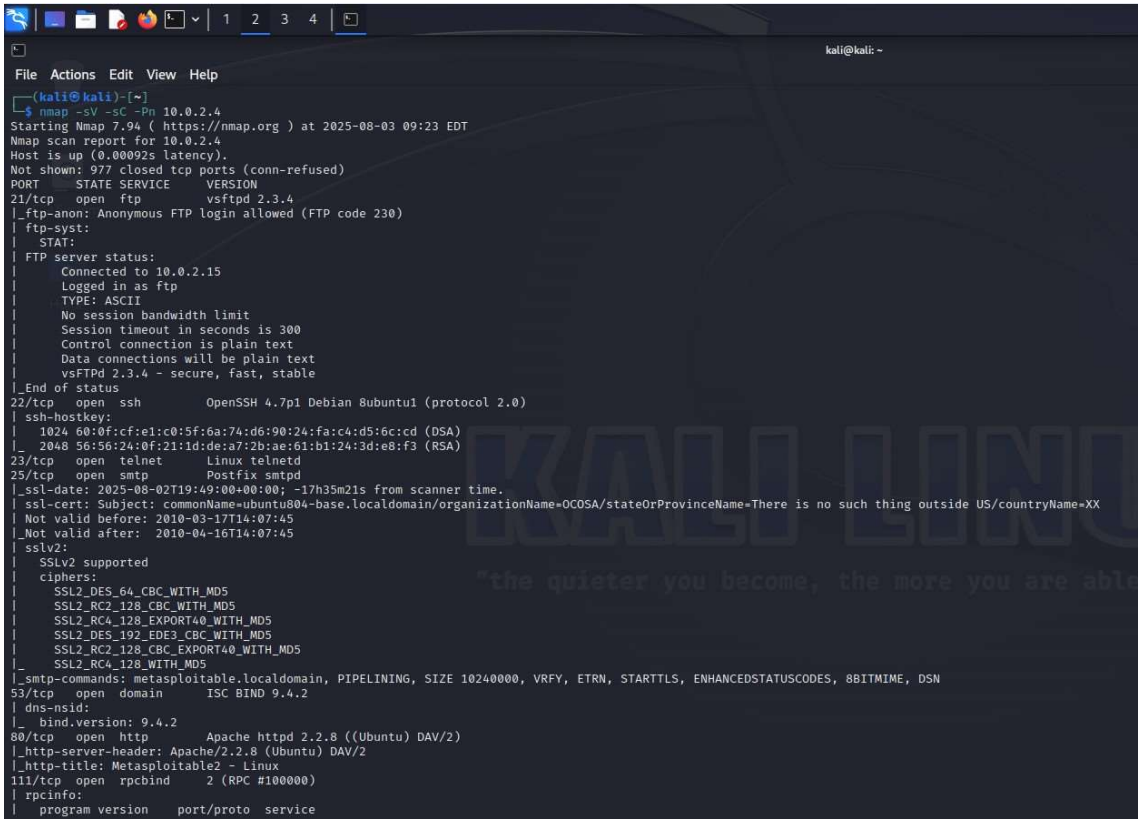

Fig. 7 – Pesquisa com Nmap de serviços e versões vulneráveis.

Atendendo aos resultados obtidos, foi feita uma pesquisa na internet sobre versões e vulnerabilidades de conhecimento geral, e obteve-se a seguinte informação:

FTP **(21/tcp)** – *vsFTPd 2.3.4*

- Permite login anónimo (código 230).
- Esta versão é conhecida por conter uma *backdoor remota* (exploração disponível via metasploit e exploit-db).

SSH (22/tcp) – *OpenSSH 4.7p1 Debian 8ubuntu1*

- Versão bastante antiga, potencialmente vulnerável a ataques de brute force (confirmado em fase anterior).

SMTP (25/tcp) – *Exim smtpd*

- Serviço de email com suporte SSL, versão não detalhada.
- Pode ser explorado dependendo da configuração.

DNS (53/tcp) – *ISC BIND 9.4.2*

- Versão com histórico de vulnerabilidades, como DoS e execução remota.

HTTP (80/tcp) – *Apache httpd 2.2.8 (Ubuntu) DAV/2*

- Versão desatualizada, com múltiplas falhas conhecidas (ex: *directory traversal*, buffer overflows, etc.).

Assim, com esta informação podemos iniciar o nosso **Searchsploit,** da seguinte maneira:

Comando: searchsploit vsftpd 2.3.4

- **Objetivo:** Verificar se a versão do servidor FTP (vsftpd 2.3.4) contém vulnerabilidades conhecidas.
- **Resultado esperado:** É uma versão notoriamente vulnerável, com uma *backdoor remota*.
- **Impacto:** A exploração desta falha pode permitir a obtenção de uma shell remota sem autenticação.

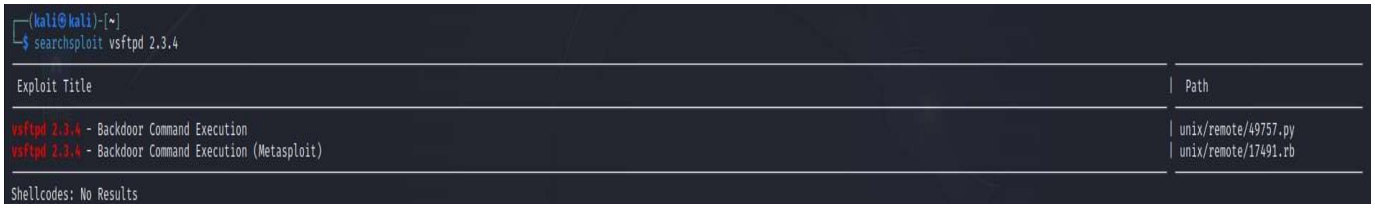

Fig. 8 – Execução de comando **searchsploit vsftpd**

Comando: searchsploit apache 2.2.8

- **Objetivo:** Avaliar vulnerabilidades no servidor web Apache encontrado.
- **Resultado esperado:** Diversos exploits associados, incluindo *directory traversal*, *DoS* e *execução arbitrária de código*.
- **Impacto:** Permite acesso não autorizado a ficheiros do sistema ou execução remota de comandos, dependendo da configuração.

Fig. 9 – Execução de comando **searchsploit apache**

Justificação da ferramenta:

O **Nmap** é uma ferramenta essencial para reconhecimento de rede, amplamente usada em auditorias técnicas. Permite uma primeira abordagem à superfície de ataque da máquina. Combinado com o **Searchsploit**, é possível verificar rapidamente se essas versões estão ligadas a vulnerabilidades públicas.

Nota pessoal:

Foi interessante observar como o Metasploitable apresenta um leque de serviços vulneráveis à vista de todos. Usar o nmap com -sC e -sV foi suficiente para descobrir múltiplas versões inseguras, que podem explicar a origem do tráfego suspeito identificado anteriormente.

Dificuldades encontradas:

A grande quantidade de serviços requer análise manual adicional para validação de riscos reais.

Algumas versões podem ser detetadas de forma genérica, exigindo técnicas complementares para confirmação (ex: banner grabbing manual).

Tema 5: Anonimização

Descrição:

A anonimização tem como objetivo garantir que a investigação seja realizada sem expor a identidade do analista ou o IP da máquina de análise. Em contextos reais, especialmente durante análises externas ou simulações de ataque (red team), o uso de ferramentas de anonimização protege contra deteção, retaliação ou responsabilização jurídica indevida.

Cenário:

Durante a investigação de incidentes de segurança, como o tráfego SSH suspeito identificado na rede da empresa fictícia **Data4Safe**, podem ser processados dados pessoais e sensíveis, como:

- IPs internos associados a colaboradores
- Nomes de utilizadores (usernames)
- Registos de data e hora de atividade

É essencial garantir que a análise e partilha de resultados não comprometa a identidade dos indivíduos envolvidos, em conformidade com o Regulamento Geral sobre a Proteção de Dados (RGPD).

Métodos de Anonimização Considerados:

Com base no manual de formação, foram avaliadas três técnicas principais de anonimização:

1. **Generalização** – Reduz a precisão dos dados substituindo-os por categorias mais amplas (ex: 25 anos → 20–30 anos).
2. **Agregação de dados** – Combina registos individuais num conjunto agregado (ex: total de acessos por hora).
3. **Mascaramento de dados (Data Masking)** – Substitui valores sensíveis por caracteres fictícios, preservando o formato (ex: joaosilva → user****).

Técnica Escolhida: Mascaramento de Dados

Optámos por aplicar a técnica de **mascaramento**, por ser prática, rápida de implementar com ferramentas nativas do **Kali Linux**, e adequada para ocultar dados diretamente em ficheiros de logs.

Ferramentas e comandos utilizados:

O objetivo principal vai ser mascarar usernames e IP's internos em ficheiros de log SSH (/var/log/auth.log ou logs simulados).

Vamos em primeiro lugar criar um ficheiro de log chamado log_ssh.txt, com uma entrada típica de log:

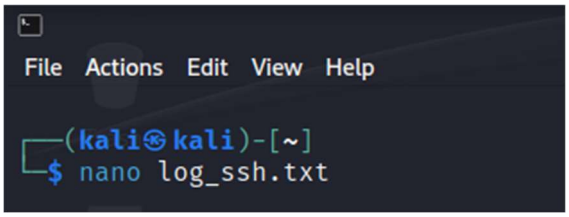

Fig. 10 – Execução de comando **nano,** para criar ficheiro

Dentro deste ficheiro de log, vamos inserir os registos (logs), que normalmente seriam gerados por um servidos SSH, geralmente encontrados em ficheiros de log do sistema. Estes nossos registos vão documentar as tentativas de autenticação realizadas no servidor:

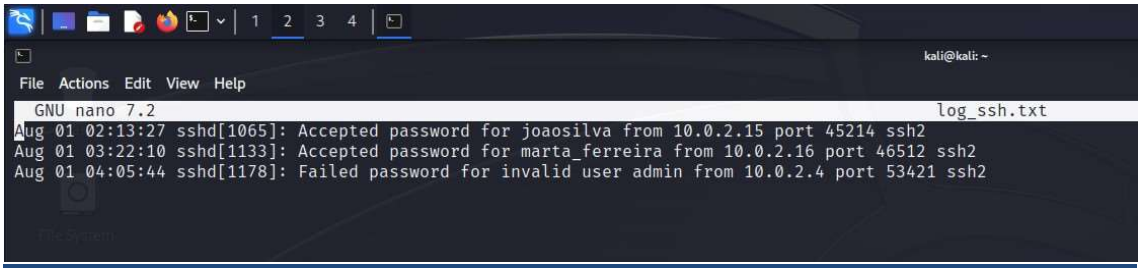

Fig. 11 – Dentro do ficheiro **log_ssh.txt** inserimos 3 registos

Na **Figura 11,** temos 3 registos inseridos, o primeiro e segundo registos mostram acessos bem sucedidos ao servidor, por parte dos utilizadores **joaosilva** e **marta_ferreira,** e a ultima linha indica uma tentativa falhada de autenticação por parte de um utilizador inválido (**admin**).

Todos os registos contem data e hora do evento registado, o processo do servidor que está a gerir a conexão, o nome do utilizador com que foi feita a tentativa, e se este conseguiu ou não autenticar-se, o endereço de IP de origem da conexão, a porta especifica que o cliente SSH utilizou na máquina de origem para estabelecer a comunicação, e a versão do protocolo SSH utilizada nessa conexão.

Após inserir os registos no ficheiro de log, fazemos **CTRL+O** para gravar, e **CTRL+X** para sair do ficheiro .txt.

De seguida vamos aplicar o mascaramento com o comando **sed**, que +e um editor de fluxo que permite realizar substituições, filtros e modificações em texto, onde vamos substituir partes de texto ou números no ficheiro **log_ssh.txt,** e gravar o resultado num novo ficheiro chamado **log_anonimizado.txt.**

Para auxiliar o comando **sed,** usamos a opção **-E,** que ativa o modo de expressões regulares estendidas, que nos permite usar padrões mais avançados e simplificados no comando **sed.**

Fig. 12 – Aqui estabelecemos os parametros para a anonimização

Descrição do código:

“ **s/for [a-zA-Z0-9_]+/for user****/ ”

Este é o primeiro comando de substituição:

- **s/.../.../** é a sintaxe básica para substituição no **sed**. O texto entre o primeiro e o segundo / é o padrão a ser encontrado, e o texto entre o segundo e o terceiro / é o que será usado para substituir.
- **Padrão a procurar:**
  - **for [a-zA-Z0-9_]+**
    - Procura pela palavra "**for**" seguida de um espaço e depois por uma sequência de caracteres alfanuméricos (letras maiúsculas ou minúsculas, dígitos) ou sublinhados (_).
    - O símbolo + indica "uma ou mais ocorrências" dos caracteres especificados.
    - Exemplo de correspondência: "**for user123**", "**for admin**", "**for user_name**".
- **Substituição:**
  - Substitui qualquer correspondência por "**for user****".
  - Exemplo: "**for user123**" torna-se "**for user****".

Exemplo:

- Entrada: **for admin**
- Saída: **for user****

“ **; s/from 10\.0\.2\.[0-9]+/from 10.0.2.XX/** “

- Este é outro comando de substituição, separado por um ponto e vírgula (**;**) do anterior.
- **Padrão de pesquisa: from 10\.0\.2\.[0-9]+**
  - Procura por uma palavra que começa com "**from**", seguida de um endereço IP no formato **10.0.2.X**, onde **X** é qualquer número.
  - O carácter **\.** é utilizado para escapar o ponto (**.**), que em expressões regulares representa "qualquer carácter". Aqui queremos que o ponto seja interpretado literalmente como parte do endereço IP.
  - **[0-9]+** significa "um ou mais dígitos".
- **Substituição:** from 10.0.2.XX

- Substitui o endereço IP encontrado por um formato anonimizado, onde o último octeto é substituído por "**XX**".

Exemplo:

- Entrada: **from 10.0.2.15**
- Saída: **from 10.0.2.XX**

“ **log_ssh.txt > log_anonimizado.txt** “

- O ficheiro de entrada é **log_ssh.txt**, ou seja, este é o ficheiro que será processado pelo comando.
- A saída do comando (>) é redirecionada para um novo ficheiro chamado **log_anonimizado.txt**. Isto significa que o conteúdo modificado será guardado no ficheiro de destino.

Depois de executarmos o comando de anonimização podemos verificar os resultados, com o comando **cat** ao ficheiro criado **log_anonimizado.txt:**

Fig. 13 – Resultado do ficheiro de logs anonimizado.

Justificação da escolha da ferramenta

A escolha do **sed** foi baseada na sua:

- Disponibilidade nativa no Kali Linux
- Capacidade de fazer substituições rápidas em texto com expressões regulares
- Eficiência e simplicidade para tarefas pontuais como mascaramento de logs

Apesar de existirem ferramentas mais avançadas para anonimização de dados em grande escala, o **sed** é extremamente útil para **scripts de resposta a incidentes**, onde é necessário anonimizar rapidamente dados antes de partilhar ou documentar.

Nota pessoal:

Durante este exercício, percebi que, embora o mascaramento com **sed** pareça simples, o maior desafio está em construir expressões regulares corretas **e** prever todos os formatos possíveis dos dados sensíveis.

Por exemplo, foi necessário ajustar o padrão para incluir underscores em usernames (marta_ferreira) e garantir que os IPs internos fossem corretamente tratados sem afetar outros números.

Este tipo de tarefa também levanta uma reflexão importante: **anonimizar não é apenas apagar dados — é saber como protegê-los mantendo a utilidade da informação**.

No contexto do meu cenário, esta técnica permitiu-me anonimizar logs sem perder a lógica do ataque SSH que estava a ser investigado. O resultado é um ficheiro que posso incluir num relatório, sem expor identidades reais.

Conclusão

Este trabalho teve como objetivo simular uma investigação a um possível incidente de segurança na empresa fictícia **Data4Safe**. Ao longo das várias etapas — desde a recolha de informações públicas até à aplicação de técnicas de anonimização — foi possível aplicar diversas ferramentas e metodologias aprendidas durante o módulo, bem como aprofundar conhecimentos através de pesquisa complementar.

Na fase de **OSINT**, utilizou-se o **theHarvester** para recolher dados disponíveis online, o que ajudou a entender melhor o "alvo". Posteriormente, a **análise de tráfego** efetuada com o tcpdump e visualizada no **Wireshark** permitiu identificar ligações SSH anormais, validando assim o alerta inicial da empresa. Foram também testadas credenciais com ferramentas como **Hydra** e **Ncrack**, demonstrando como senhas fracas podem comprometer a segurança de um sistema. Na etapa de **auditoria**, o **Nmap** revelou os serviços ativos e as suas versões, enquanto o **Searchsploit** facilitou a correlação desses dados com vulnerabilidades conhecidas.

Na etapa final, dedicada à **anonimização**, aplicou-se a técnica de **mascaramento de dados** para anonimizar usernames e endereços IP internos num ficheiro de log SSH fictício, preservando a utilidade dos dados para análise, mas garantindo que não permitiam a identificação dos titulares.

Para efeitos deste trabalho, o ficheiro de log utilizado foi criado manualmente no Kali Linux com dados fictícios. O processo de anonimização foi realizado sobre este ficheiro no próprio editor e terminal, sem qualquer tipo de intrusão ou acesso não autorizado a sistemas reais. Embora não tenha havido consulta a dados reais para confirmar a anonimização, este é precisamente o objetivo desta técnica: garantir que, em caso de conter dados pessoais ou sensíveis, estes possam ser partilhados e analisados de forma segura, sem risco de identificação dos titulares.

Durante o desenvolvimento desta simulação, foi necessário realizar pesquisa adicional para consolidar conhecimentos teóricos e práticos, especialmente para dominar comandos e parâmetros específicos que inicialmente desconhecia. Esse processo de aprendizagem foi fundamental para executar corretamente as ferramentas e obter resultados confiáveis.

Esta experiência demonstrou a importância de seguir um processo estruturado e utilizar as ferramentas adequadas em cada fase da investigação. Além disso, reforçou a necessidade de manter boas práticas de segurança — como a utilização de senhas fortes e a atualização constante dos serviços — e evidenciou o valor do conhecimento das técnicas ofensivas para fortalecer a defesa dos sistemas.