Estudo de caso sobre a implementação de técnicas de blindagem em servidores Linux baseada na detecção de vulnerabilidades e tentativas de intrusão

Álisson Wegner Sousa[1], Gabriel Marvel Vaz[1],Vinícius Röpke[1], Lucas Vargas Dias[1], Tiago Antonio Rizzetti[1]

[1]Redes de Computadores – Universidade Federal de Santa Maria (UFSM)
Caixa Postal 5082 -– CEP: 97.105-900 – Santa Maria – RS – Brazil
{alissonwegner,gabrielmarvelvaz,viniropke,lucas_dias,rizzetti}
@redes.ufsm.br

Resumo. *Neste artigo é apresentado um estudo de caso baseado na detecção de vulnerabilidades e tentativas de intrusão em uma infraestrutura de rede composta por servidores Linux, seguido do emprego de técnicas de blindagem para mitigar as ameaças existentes. Para realizar as detecções foram utilizados o scanner OpenVAS e o Intrusion Detection System (IDS) Suricata. Com a implementação das técnicas de blindagem foi possível tornar a rede mais segura, mitigando as vulnerabilidades existentes e tentativas de intrusão. Ademais, concluiu-se que o processo de identificar, compreender, solucionar e testar as soluções dessas ameaças pode ser custoso em relação ao tempo empregado. Devido a isso, há necessidade de maior automação para que o tratamento de questões relacionadas à segurança das organizações e indivíduos se torne mais rápido e eficiente.*

1. Introdução

Em um ambiente amplamente conectado, novas vulnerabilidades são descobertas diariamente e os riscos aumentam de forma expressiva, tornando imprescindível e imediata a necessidade de empregar técnicas e ferramentas de segurança para mitigá-las. *Scanners* de vulnerabilidades e *Intrusion Detection Systems* (IDSs) podem auxiliar no processo de avaliação de riscos, permitindo identificar vulnerabilidades e ameaças, que são etapas importantes neste processo [William Stallings 2014]. *Hardening* trata-se de proteger um sistema através da redução de suas possíveis vulnerabilidades, por meio de configurações que implementam controles específicos [Melo 2014]. De acordo com as técnicas apresentadas por [Melo 2014], para se realizar o *hardening*, pode ser utilizada a abordagem de proteção de perímetro, que é executada diretamente no *firewall*, protegendo a rede em geral. Em conjunto, pode ser empregada uma abordagem individualizada, visando tratar especificidades de determinados equipamentos, aumentando sua segurança.

Este trabalho tem como objetivo apresentar um estudo de caso desenvolvido na rede de uma instituição de ensino, no qual foram utilizadas ferramentas para detecção de vulnerabilidades e intrusões. Com base nos dados obtidos, definiu-se um fluxo de avaliação e mitigação das vulnerabilidades, seguido do emprego de técnicas de filtragem e blindagem necessárias para tornar a rede mais segura. A partir das medidas implementadas, foi possível observar o impacto destas, e ainda, alguns desafios a serem enfrentados no âmbito de segurança da rede.

2. Trabalhos Relacionados

No trabalho proposto por [Chalvatzis et al. 2019] é realizado um comparativo entre as ferramentas de escaneamento de vulnerabilidades Nessus, OpenVAS e *Nmap Scripting Engine* (NSE). Os autores avaliam questões como capacidade de detecção de vulnerabilidades, velocidade e usabilidade. O Nessus apresenta pequena vantagem nessa comparação, principalmente devido à sua base de conhecimento, que abrange um número maior de *Common Vulnerabilities and Exposures* (CVE). Porém, considerando que algumas CVEs não são detectadas por serem muito antigas ou específicas, essa diferença pode ser muito menor do que parece.

Em [Huang et al. 2012] optou-se pelo IDS Snort devido à sua capacidade de monitorar e analisar o fluxo de dados em tempo real, gerando alertas conforme as regras configuradas [Snort 2020]. A partir da análise dos alertas obtidos, foram selecionadas técnicas de *hardening* para aumentar a segurança da rede através de configurações no *firewall*. Com isso, foi possível observar os fluxos de tráfego e resolver problemas como o volume de *big data* e a baixa velocidade de detecção de vulnerabilidades, consequentemente melhorando o desempenho da rede.

Já em [Mello 2017], o autor realiza a implementação de alguns mecanismos de blindagem em um servidor *web* de uma instituição de saúde, empregando técnicas baseadas no princípio de privilégio mínimo, além de configurações específicas nas aplicações Apache e MySQL. Após as blindagens, o autor utiliza a ferramenta Lynis para buscar vulnerabilidades, de modo a verificar a eficácia das técnicas empregadas.

3. Estudo de caso

A rede sob a qual foi realizado o estudo de caso possui uma faixa de endereços *Internet Protocol* versão 4 (IPv4) públicos, ou seja, acessíveis pela Internet. Portanto, os principais *hosts* da rede, entre eles servidores *Domain Name System* (DNS), *Dynamic Host Configuration Protocol* (DHCP) e servidores *Web*, são acessíveis externamente. Devido a isso, o trabalho foi focado nestes *hosts*. O fluxograma da Figura 1 apresenta a ordem lógica das ações tomadas a partir da detecção das ameaças, que serviu para otimizar o processo de blindagem e pode ser utilizada em outros ambientes para este propósito.

A estrutura desta Seção é composta pelo mapeamento das vulnerabilidades existentes através da ferramenta OpenVAS e correção das falhas encontradas, descrito na Subseção 3.1. Em sequência, foram tratadas a detecção de tentativas de intrusão com o IDS Suricata e mitigação das respectivas ameaças na Subseção 3.2. Por fim, na Subseção 3.3, foi realizada uma análise da eficácia das soluções implementadas através de um novo escaneamento e também de uma nova amostra de tentativas de intrusão.

3.1. Escaneamento de vulnerabilidades com OpenVAS e técnicas de blindagem empregadas

A ferramenta OpenVAS trata-se de um *software* livre para escaneamento de vulnerabilidades acompanhado por um *feed* da comunidade *Greenbone* que inclui mais de 50.000 testes de vulnerabilidade [OpenVAS 2020]. A varredura pode ser configurada e executada via interface *Web*, na qual também é possível visualizar os resultados e gerar relatórios a partir destes. Vale ressaltar que a faixa de endereços públicos utilizada possui uma

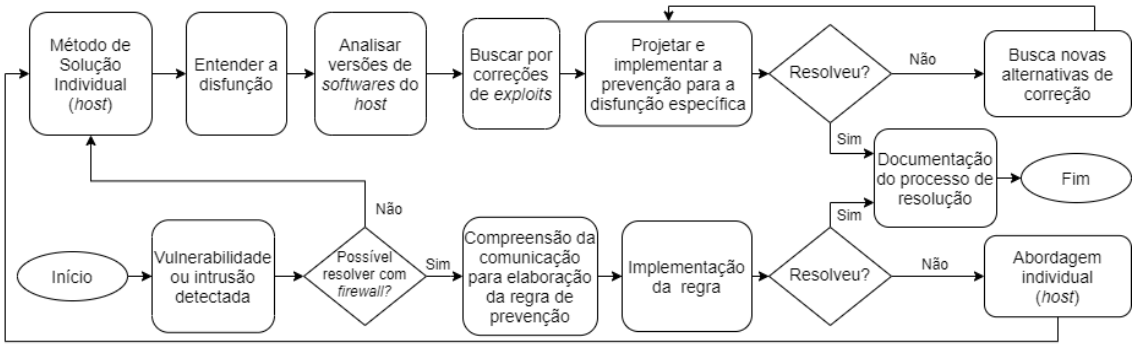

Figura 1. Metodologia adotada para mitigação das ameaças.

máscara de rede /25 e em torno de 10 *hosts* ativos. O escaneamento encontrou 27 vulnerabilidades. Foi encontrada apenas uma vulnerabilidade de severidade alta, 18 de nível médio e 8 de nível baixo. De acordo com as recomendações do NIST (*National Institute of Standards and Technology*) [Stoneburner et al. 2002], as probabilidades de uma possível vulnerabilidade ser explorada por uma determinada fonte de ameaça pode ser descrita como alta, média ou baixa.

Na Tabela 1 temos a listagem das vulnerabilidades encontradas, bem como seu nível de severidade e descrição. Foram encontradas 27 vulnerabilidades no total. Porém algumas detecções são redundantes, ou seja, as mesmas vulnerabilidades foram encontradas em *hosts* distintos. No total, foram encontradas 10 vulnerabilidades distintas.

Tabela 1. Resultados do escaneamento com o OpenVAS.

Nome da vulnerabilidade	Severidade	Descrição
OS End Of Life Detection	Alta	Sistema operacional desatualizado.
HTTP *Debbuging Methods (TRACE/TRACK) Enabled*	Média	Método *TRACE* ativado, vulnerável a ataques *Croos-Site-Tracing* (XST).
***Unprotected Web App Installers* (HTTP)**	Média	Páginas de configuração de aplicativos disponíveis para acesso no servidor *Web*.
Missing 'httpOnly' Cookie Attribute	Média	*Cookies* não estão usando o atributo de *cookie* seguro '*httponly*'.
Kill service with random data	Média	Um atacante pode executar código arbitrário e travar o sistema.
***Cleartext Transmission of Sensitive Information via* HTTP**	Média	Uso de HTTP, que não utiliza criptografia na comunicação.
Telnet *Unencrypted Cleartext Login*	Média	Uso de Telnet, que não utiliza criptografia na comunicação
SSH *Weak Encryption Algorithms Supported*	Média	Servidor SSH utilizando algoritmos de criptografia fracos, como: "3-des-cbc", "aes128-cbc" entre outros.
jQuery < 1.9.0 XSS *Vulnerability*	Média	Versão do jQuery vulnerável a ataques *Cross-site Scripting* (XSS).
TCP *Timestamps*	Baixa	Permite que um atacante obtenha o *uptime* do servidor remotamente.

Identificadas e compreendidas as vulnerabilidades, foram realizadas as configurações necessárias para sanar os problemas existentes. As técnicas empregadas consistiram

em configurações específicas nos servidores, que são descritas resumidamente na Tabela 2.

Tabela 2. Técnicas implementadas para mitigar as vulnerabilidades.

Vulnerabilidade	Solução implementada
OS End Of Life Detection	Atualização do sistema operacional.
Kill service with random data	
***Cleartext Transmission of Sensitive Information via* HTTP**	Migração das páginas *Web* de HTTP para HTTPS.
HTTP *Debugging Methods* (*TRACE/TRACK*) *Enabled*	Desativação do método *TRACE* no servidor *Web*.
***Unprotected Web App Installers* (HTTP)**	Modificação nas permissões de acesso às páginas de configuração e instalação do servidor *Web*.
Missing "httpOnly" Cookie Attribute	Ativação do atributo de *cookie* seguro "*httponly*" no servidor *Web*.
Relacionadas à Telnet e SSH	Notificação dos administradores responsáveis.
TCP *timestamps*	Desativação da extensão *"tcptimestamps"*.

3.2. Detecção de intrusões com o IDS Suricata e técnicas de blindagem empregadas

Optou-se por posicionar o IDS Suricata de forma passiva na rede, analisando os pacotes encaminhados e recebidos, objetivando detectar a existência de intrusos. Na figura 2 vemos a localização do IDS na rede, que recebe uma cópia do tráfego através do mecanismo de *port mirroring* disponível nos *switches*.

O Suricata foi escolhido pelo seu mecanismo *multithread*, que fornece maior velocidade na análise do tráfego, usando um poder maior de processamento fornecido pelos processadores multi-núcleo [Wong et al. 2017].

A ferramenta disponibiliza um conjunto de regras padrão da comunidade [Suricata 2019], o qual foi utilizado neste estudo de caso. Após 5 dias de execução (anteriores às blindagens), foram gerados 69.990 alertas. O gráfico apresentado na Figura 3 mostra a porcentagem equivalente aos principais tipos de assinaturas detectadas. A partir do relatório de detecção foi possível identificar que a maioria das tentativas de intrusão relacionavam-se à *scans*, tentativas de acesso remoto via SSH ou exploração de *exploits* em servidores *Web*.

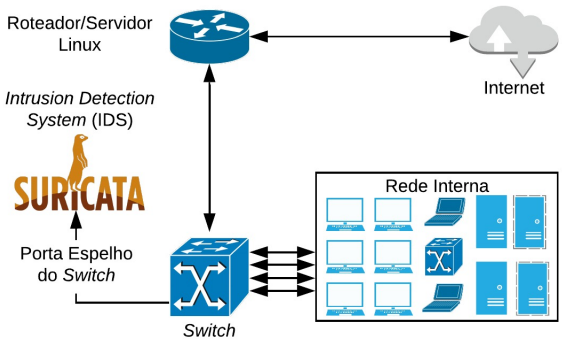

Figura 2. Localização do IDS.

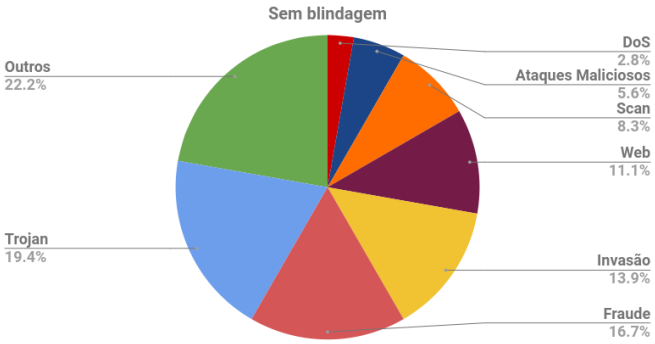

Figura 3. Porcentagem de assinaturas detectadas.

Com o objetivo de mitigar as ameaças detectadas, foram utilizadas as técnicas descritas na Tabela 3, baseando-se nos processos de *hardening* propostos em [Melo 2014] de acordo com cada ameaça. Após a implementação dos mecanismos de blindagem, foram empregados os mesmos processos de detecção, que serão descritos na Seção 3.3, desta vez com o objetivo de verificar a eficácia da mitigação das ameaças.

Tabela 3. Técnicas implementadas com base nas detecções do IDS.

Ameaça	**Medidas de Prevenção**
Scan	-
Tentativas SSH	Alteração da porta padrão SSH em todos os servidores que possuem IP público
Exploits Web	Bloqueio de portas e páginas *Web* que não necessitam ser acessadas publicamente; Implementação do HTTPS nos servidores *Web*.

3.3. Resultados

Conforme apresentado na Figura 4, o novo escaneamento do OpenVAS detectou apenas sete vulnerabilidades, sendo quatro delas referentes a *hosts* fora do domínio dos autores. Das outras três vulnerabilidades encontradas, uma delas é referente à versão do jQuery, que ainda não foi resolvida também por questões externas ao domínio dos autores e as outras duas são referentes aos servidores *Web*, nos quais serão feitas as configurações necessárias para mitigá-las. Com o novo escaneamento observou-se uma redução de 89,95% no número de vulnerabilidades encontradas.

Já com o Suricata, após 5 dias de análise posteriores às blindagens foram gerados 20.373 alertas, o que equivale a uma diminuição de 30% em relação às detecções iniciais. No gráfico da Figura 5 podemos observar que as tentativas de escaneamento predominaram neste período, seguidas de tentativas de ataques maliciosos e demais intrusões variadas.

4. Comparativo entre a proposta e os trabalhos relacionados

Esta Seção apresenta a comparação dos trabalhos relacionados com o estudo de caso descrito neste artigo, conforme ilustrado na Tabela 4. O comparativo é feito baseado nos dois pontos principais abordados neste estudo de caso, bem como os resultados obtidos.

Vulnerability	Severity ▾	QoD	Host IP	Host Name	Location	Created
mod_access_referer 1.0.2 NULL pointer dereference	5.0 (Medium)	99 %			443/tcp	Tue, Sep 15, 2020 6:25 AM UTC
SSL/TLS: Report Vulnerable Cipher Suites for HTTPS	5.0 (Medium)	98 %			443/tcp	Tue, Sep 15, 2020 4:52 AM UTC
Telnet Unencrypted Cleartext Login	4.8 (Medium)	70 %			2323/tcp	Tue, Sep 15, 2020 3:25 AM UTC
SSH Weak Encryption Algorithms Supported	4.3 (Medium)	95 %			2222/tcp	Tue, Sep 15, 2020 3:26 AM UTC
jQuery < 1.9.0 XSS Vulnerability	4.3 (Medium)	80 %			443/tcp	Tue, Sep 15, 2020 3:33 AM UTC
SSH Weak MAC Algorithms Supported	2.6 (Low)	95 %			2222/tcp	Tue, Sep 15, 2020 3:28 AM UTC
TCP timestamps	2.6 (Low)	80 %			general/tcp	Tue, Sep 15, 2020 3:27 AM UTC

Figura 4. Resultado do escaneamento com o OpenVAS após as blindagens.

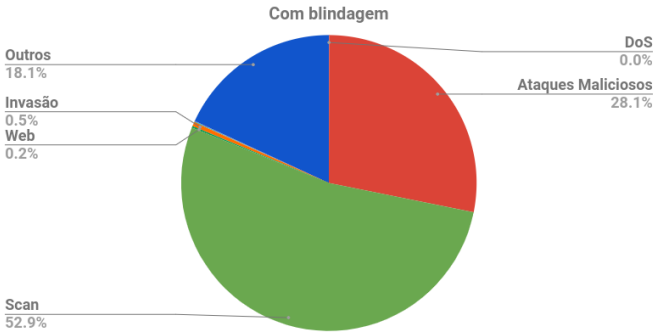

Figura 5. Porcentagem de assinaturas detectadas após a blindagem.

5. Conclusões e trabalhos futuros

Com a implementação do trabalho foi possível detectar ameaças e mitigá-las, tornando a rede consideravelmente mais segura e confiável para seus usuários. Além disso, os procedimentos de segurança realizados foram padronizados através da documentação, visando tornar a detecção e prevenção de ameaças mais usual aos administradores, possibilitando controlar a rede de uma forma segura.

Mesmo com a documentação e padronização dos passos a serem realizados, a tarefa de detecção de vulnerabilidades e intrusões, bem como a implementação de técnicas de blindagem, apresenta-se consideravelmente custosa em termos de complexidade e tempo demandado. Haja vista, se identifica a necessidade de automatizar estes processos através de um *software* capaz de realizar as blindagens automaticamente de acordo com a detecção de intrusão ou vulnerabilidade inserida. O desenvolvimento deste *software* será estudado para implementação em trabalhos futuros.

Referências

Chalvatzis, I., Karras, D. A., and Papademetriou, R. C. (2019). Evaluation of security vulnerability scanners for small and medium enterprises business networks resilience towards risk assessment. In *2019 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)*, pages 52–58.

Tabela 4. Comparativo com os trabalhos relacionados.

Trabalho	Detecção	Blindagem	Resultados
[Chalvatzis et al. 2019]	Os autores realizam um comparativo entre os *scanners* Nessus, NSE e OpenVAS.	O trabalho não abrange este ponto.	Nessus e OpenVAS apresentaram vantagem em relação ao NSE de acordo com as métricas utilizadas, com o Nessus levando pequena vantagem no geral.
[Huang et al. 2012]	Utiliza o IDS Snort para detectar intrusões.	Realiza configurações no *firewall* de acordo com as detecções.	Foram resolvidos problemas como volume de *big data* e baixa velocidade de detecção de intrusões na rede.
[Mello 2017]	Utiliza a ferramenta Lynis para detectar vulnerabilidades após implementar as técnicas de blindagem.	Implementa mecanismos como limitação de horário para acesso remoto, configuração do comando *sudo*, *suid bit*, entre outras técnicas baseadas no princípio do privilégio mínimo.	Com as técnicas implementadas foi possível elevar o nível de segurança do servidor *Web*, tornando-o menos propenso a ataques danosos.
[Autores 2020]	Uso do *scanner* OpenVAS para detecção de vulnerabilidades específicas nos servidores, em conjunto com o IDS Suricata para detectar tentativas de intrusão à rede.	Implementa individualmente as técnicas necessárias para solucionar as vulnerabilidades existentes e também configurações no *firewall* para tornar a rede mais segura.	Aumento da segurança dos principais servidores e da rede em geral a partir das técnicas implementadas. E além disso, diminuição no número de tentativas de intrusão.

Huang, C., Xiong, J., and Peng, Z. (2012). Applied research on snort intrusion detection model in the campus network. In *2012 IEEE Symposium on Robotics and Applications (ISRA)*, pages 596–599.

Mello, B. F. (2017). Estudo da aplicação da técnica de Hardening nos servidores web do Hospital de Clínicas de Porto Alegre. *Trabalho de Conclusão de Curso - Gestão da Segurança da Informação - Unisul Virtual*.

Melo, S. (2014). Hardening em Linux. *Escola Superior de Redes - RNP*.

OpenVAS (2020). OpenVAS - Open Vulnerability Assessment Scanner.

Snort (2020). Snort Overview.

Stoneburner, G., Goguen, A., and Alexis, F. (2002). Risk Management Guide for Information Technology Systems. *Recommendations of the National Institute of Standards and Technology , NIST Special Publication 800-30*.

Suricata (2019). Rule Management with Suricata-Update.

William Stallings, L. B. (2014). *Segurança de Computadores: Princípios e Práticas*. Rio de Janeiro.

Wong, K., Dillabaugh, C., Seddigh, N., and Nandy, B. (2017). Enhancing suricata intrusion detection system for cyber security in scada networks. In *2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE)*, pages 1–5.