Segurança de Sistemas – 2024-2025

Relatório do Trabalho de grupo D4

Nome Iuri Carrasqueiro (2222059)
Nome João Tendeiro (2222047)
Nome Miguel Lopes (2222397)

Introdução

Na era digital moderna, a necessidade de armazenar e salvaguardar dados em formatos digitais tornou-se um desafio significativo para empresas e instituições de vários setores. A vulnerabilidade de dispositivos como servidores e computadores a acessos não autorizados tem facilitado a manipulação, o roubo e até a destruição de dados críticos. Neste projeto pretendemos mostrar como podemos deixar os sistemas mais seguros, para isso utilizámos um Raspberry Pi Zero 2 W como ferramenta central (equipado com um processador ARM Cortex-A53 de 64-bit, quad-core a 1GHz, e 512MB de SDRAM).

Para a criação do domínio, foi utilizado o serviço da dominios.pt, que oferece, de forma gratuita, a obtenção de um domínio “.pt” por um ano. O domínio obtido pelo nosso grupo foi "ijm.pt".

Com a criação do domínio, conseguimos obter, também de forma gratuita, um certificado TLS assinado e verificado pela entidade "Let's Encrypt", o que nos possibilitou criar um site utilizando o protocolo HTTPS. Para implementar o site, foi utilizado o Apache2 como servidor web.

Para aumentar a segurança no acesso à máquina, implementámos o serviço fail2ban, que bloqueia temporariamente endereços IP após 4 tentativas falhadas de acesso via SSH. Adicionalmente foram criadas chaves assimétricas EdDSA que permitiram a partir de uma chave pública e privada aceder à máquina via SSH sem a necessidade de utilizar uma password.

A segurança do servidor foi aumentada com a implementação de uma firewall utilizando o serviço iptables. Para evitar a aplicação manual das regras uma a uma pela linha de comandos, foram escritas num script Bash para trazer mais organização, flexibilidade e agilidade nas alterações que foram surgindo.

A eficácia das regras de firewall foi verificada utilizando a ferramenta Nmap, juntamente com outras ferramentas complementares, que nos ajudaram a assegurar que as configurações estavam a funcionar conforme o esperado, garantindo a proteção do servidor contra possíveis ataques externos.

Para reforçar ainda mais a proteção do servidor, implementámos o Port Knocking, uma técnica que permite abrir portas de forma dinâmica e segura, exigindo uma sequência específica para o acesso ao serviço SSH (22), configurámos uma VPN WireGuard para garantir um acesso remoto seguro à rede interna do servidor e o Bitwarden como gestor de passwords.

Desenvolvimento

Registo do Domínio

De modo a registar um domínio para o nosso site, recorremos ao domínios.pt que trabalha na área dos registos de domínios, alojamentos de Websites, e hosting de servidores. Graças à sua oferta de um ano de domínio grátis foi possível criar o domínio "ijm.pt" de forma gratuita.

Para registar um domínio com esta empresa, é necessário criar uma conta e escolher o nome para o nosso domínio.

O serviço domínios.pt disponibiliza uma dashboard interativa onde o utilizador pode fazer a gestão dos seus nomes e endereços IP. Neste dashboard configurámos os servidores DNS para o serviço da Cloudflare para a gestão do domínio e subdomínios, como também para a criação do túnel usado no serviço Bitwarden.

Criação do Web Server

No nosso projeto, utilizamos o Apache 2 como servidor web para hospedar e gerir os recursos do site. O Apache 2 foi configurado para servir um site estático, no qual o arquivo HTML é entregue diretamente aos utilizadores, sem necessidade de processamento dinâmico. Para isso, criamos um diretório específico para o arquivo do site e ajustamos a configuração do Apache para apontar para esse local, associando-o ao domínio do projeto. Após realizar a configuração dos arquivos de host virtual e ativar o site no Apache, o servidor foi reiniciado para garantir que todas as modificações fossem aplicadas corretamente. O Apache 2 oferece uma solução robusta e flexível para garantir a entrega eficiente e segura dos recursos do site.

Criação do certificado TLS

Para garantir que o nosso site funcione de maneira segura e criptografada, implementámos um certificado TLS (SSL), utilizando o Certbot, uma ferramenta automatizada que facilita a criação e instalação de certificados SSL gratuitos da Let's Encrypt. O Certbot foi configurado no servidor Apache 2 para solicitar e instalar o certificado, o que permite que o site seja acedido por HTTPS, proporcionando uma comunicação segura entre os utilizadores e o servidor.

Durante o processo, o Certbot verificou automaticamente o domínio e configurou o Apache para redirecionar todas as requisições HTTP para HTTPS, garantindo que todas as conexões sejam criptografadas. Além disso, o Certbot também configurou a renovação automática do certificado, o que significa que o certificado será renovado sem intervenção manual, garantindo a continuidade da segurança.

Com a implementação do TLS, conseguimos proteger os dados dos utilizadores, prevenir ataques como o man-in-the-middle e oferecer uma navegação mais confiável e segura.

Configuração da firewall

Para proteger o servidor contra-ataques, implementámos uma firewall que limita o tráfego de entrada e saída baseando-se nos portos específicos que estão abertos**.** Para fazermos essa configuração usámos o IPTABLES, que filtra todos os pacotes que entram e saem da máquina, descartando aqueles que não cumprem as regras estabelecidas.

No início do script de configuração da firewall, inserimos comandos para limpar todas as regras existentes do iptables e eliminar listas personalizadas. Estabelecemos uma política por omissão que automaticamente descarta pacotes que não estão em conformidade com as regras especificadas.

A firewall foi configurada para:

- Permitir a comunicação via interface loopback, essencial para o funcionamento interno do sistema.
- Permitir apenas o tráfego de entrada e saída de pacotes associados a conexões previamente estabelecidas.
- Registar e bloquear pacotes inválidos que tentem entrar ou sair da máquina.
- Proteger contra-ataques por flood (inundações), limitando a quantidade de pacotes ICMP, TCP e UDP para evitar que haja sobrecargas.
- Monitorizar através de logs e autorizar pacotes de entrada para protocolos específicos como ICMP, HTTP, HTTPS e SSH.
- Aceitar pacotes de saída em estado “new” para diversos protocolos, incluindo ICMP, DNS, DNS over TLS, SSH, Docker, whois, git, HTTP e HTTPS.

Configuração do sshttp

O sshttp é uma ferramenta de multiplexação que permite a comunicação de dois protocolos distintos, HTTPS e SSH, através de um único porto (443). O sshttp analisa o início de cada conexão para identificar se o pedido é de HTTPS ou SSH, encaminhando-o para o serviço apropriado: a porta 44044 para HTTPS e a porta 22022 para SSH, aceitando apenas pedidos provenientes do serviço sshttp (443). Esta abordagem permite manter um único porto (443) para ambos os protocolos, simplificando a configuração da rede e permitindo bypass de restrições, enquanto mantém a segurança e a funcionalidade de cada protocolo.

Configuração do ssh

Na configuração do SSH fizemos algumas alterações no ficheiro de configuração. O ssh apenas comunica por IPv4, não é possível fazer login com a conta root, nem com o uso de uma palavra-passe.

Para realizar login só podemos usar a conta de utilizador ijm que tem as chaves assimétricas criadas. As chaves assimétricas foram criadas para permitir um login mais seguro, mas também como um login mais rápido.

Além desta configuração no SSH, foi instalado o serviço fail2ban que permite bloquear um endereço IP de um dispositivo que se tente conectar por SSH ao nosso servidor e houver erro na troca de chaves mais que 4 vezes, fica bloqueado por 10 minutos.

Serviços Extras

Port Knocking

O Port Knocking é uma técnica de segurança que permite abrir portas de um servidor de forma dinâmica, sem expô-las diretamente à rede. Para utilizá-la, é necessário configurar a ferramenta knockd, que analisa pacotes enviados para portas específicas em uma sequência predefinida. No nosso caso, configuramos o servidor para monitorar as portas 22059, 22047 (UDP) e 22397, que, quando transmitidas na ordem correta, permitem que o porto 22 (SSH) seja aberto temporariamente, proporcionando acesso ao servidor.

WireGuard

O WireGuard é uma VPN simples e rápida. Escolhemos esta solução para permitir o acesso seguro à rede interna do nosso servidor a partir dos nossos computadores. Instalámos o WireGuard nos nossos dispositivos e na configuração do servidor, incluímos os nossos IPs e as respetivas chaves públicas, garantindo que só esses dispositivos têm acesso. Esta configuração é fiável e segura, sendo ideal para o nosso projeto, onde a privacidade e a integridade da informação são cruciais.

Bitwarden

Implementámos o Bitwarden, que é um serviço de gestão de passwords usando o nosso próprio domínio via cloudflare. Este corre diretamente no nosso Raspberry Pi, e configurámos o Cloudflare para criar um túnel seguro na porta 7844. Esse túnel permite acesso remoto ao Bitwarden por meio de um subdomínio personalizado, mantendo a segurança e a criptografia do tráfego sem necessidade de expor diretamente o Raspberry Pi à Internet.

Testes

Para garantir a segurança e o controlo de tráfego em rede, foram realizados testes com ferramentas como iptables, nmap, ping, wget, ssh, git, whois, dig e nslookup. Esses testes validaram o acesso às portas e o funcionamento de serviços críticos. Além disso, métodos como Port Knocking e WireGuard foram utilizados para validar o controle de acesso seguro.

Controle de tráfego HTTP e SSH de INPUT com wget, nmap , ssh e ping:

Usamos o wget para simular pedidos HTTP, o nmap para inspecionar a acessibilidade das portas, e o ssh para testar conexões remotas via SSH, confirmando que as regras de firewall permitem apenas o tráfego autorizado nessas portas (22, 80 e 433). O ping foi utilizado para testar o ICMP.

Controle de limite de pacotes ICMP (ping), UDP e TCP de INPUT e com hping3:

O hping3 foi utilizado para simular pacotes ICMP com uma taxa de 5 pacotes por segundo, pacotes UDP com taxa de 10 pacotes por segundo e pacotes TCP com um taxa de 50 pacotes por segundo, garantindo que as regras de firewall restrinjam o tráfego conforme configurado.

Controle de tráfego UDP e TCP de OUTPUT com nmap, wget, ssh, git, whois, dig e nslookup:

O nmap foi utilizado para verificar se as portas estavam abertas para DNS, HTTP, HTTPS, SSH, Git, Whois e DNS sobre TLS. O wget testou a comunicação HTTP (80) e HTTPS (443), enquanto o ssh validou a conectividade remota na porta SSH (22). O git simulou a comunicação com o serviço de Git na porta 9418, o whois foi utilizado para consultar informações de domínio na porta Whois (43), e o dig validou consultas DNS seguras através da porta 853. O nslookup foi utilizado para realizar consultas DNS, testando a conectividade na porta 53. Esses testes garantem que as regras de firewall permitam o tráfego necessário e mantenham a segurança da rede.

Controle de logs:

Para testar os logs gerados durante esses testes, utilizamos o comando journalctl -f, que permite visualizar os logs em tempo real do sistema e verificar se as regras da firewall estão aplicadas corretamente, garantindo que o tráfego necessário seja permitido e a segurança da rede seja mantida.

Controle de Acesso Seguro via Port Knocking com knock:

A ferramenta knock é utilizada para inserir e apagar regras na firewall como parte do serviço de port knocking. Enviando uma sequência específica de pacotes TCP e UDP, o knockd abre temporariamente a porta SSH e uma nova sequência apaga a regra, fechando o acesso. Podendo ser verificado com nmap.

Controle de Acesso Seguro via WireGuard com wg-quick:

A ferramenta wg-quick adiciona ou remove regras no iptables para gerir o acesso à rede via WireGuard. Essas regras podem ser testadas realizando uma conexão VPN via WireGuard, garantindo que o tráfego seja corretamente encaminhado de acordo com as configurações aplicadas.

Gestor de passwords Bitwarden:

Para testar o serviço Bitwarden criámos 3 contas através do subdomínio bitwarden.ijm.pt. Instalámos a extensão do browser para estabelecer comunicação com o serviço e acrescentamos credenciais de acesso de vários websites, inserindo automaticamente, ao iniciar sessão, as credenciais nesses websites.

Limitações

O uso de um Raspberry Pi como plataforma de desenvolvimento apresentou algumas limitações devido aos recursos limitados do dispositivo. A memória RAM, em particular, não foi suficiente para permitir a execução de vários serviços simultaneamente, o que restringiu a implementação de serviços adicionais. Além disso, o processador ARM de 64 bits trouxe dificuldades devido à falta de compatibilidade com alguns serviços, por exemplo o snort.

Anexos

- firewall-D4.sh – ficheiro com a script para carregar todas as regras da firewall IPTables.
- test-D4.xls – Excel que contem os comandos da firewall, os respetivos comandos de teste e as explicações.