E-fólio A | Folha de resolução para E-fólio

UNIDADE CURRICULAR: Segurança em Redes e Computadores

CÓDIGO: 21181

DOCENTE: Henrique S. Mamede

A preencher pelo estudante

NOME: João Paulo Alves Correia Mendes Pires

N.º DE ESTUDANTE: 2203810

CURSO: Licenciatura em Engenharia Informática

DATA DE ENTREGA: 18/11/2024

TRABALHO / RESOLUÇÃO:

1. Introdução

O objetivo deste trabalho é aplicar os princípios de segurança de computadores a uma arquitetura de rede especifica.

Para atingir o objetivo proposto identifiquei e analisei as principais superfícies de ataque e ameaças da referida rede segundo o modelo da tríade da CIA, e proponho controlos de segurança devidamente justificados e fundamentados nos princípios orientadores da garantia da confidencialidade, integridade e disponibilidade do sistema.

Pretende-se, também, implementar proteções criptográficas básicas com base em encriptação simétrica e assimétrica.

2. Análise da rede e das ameaças

Análise do diagrama de rede e principais componentes:

A rede desta organização apresenta uma arquitetura que começa com uma ligação à Internet através de um router, seguido por uma firewall que atua como primeira linha de defesa.
Seguidamente, o tráfego é direcionado para um switch central que interliga todos os componentes internos da rede, onde se incluí um sistema IDS/IPS responsável pela deteção e prevenção de intrusões e uma Zona Desmilitarizada (DMZ) que aloja os três servidores críticos da organização: um Database Server para armazenamento de dados, um App/Web Server para aplicações e serviços web, e um servidor de Email para gestão da comunicação via Email.

A separação dos três servidores críticos da organização é crucial pra proteger serviços que precisam ser acessíveis externamente.
Esta configuração da rede permite análise contínua de todo tráfego que passa na rede por parte dos sistemas IDS/IPS.

O acesso dos clientes é fornecido através de duas redes distintas, ligadas também ao switch:

- Uma LAN (Local Area Network), constituída por equipamentos fixos ligados através de cablagem a um router que por sua vez se liga ao switch, destinada a fornecer acesso a equipamentos como computadores desktop, scanners e impressoras;
- Uma WLAN (Wireless Local Area Network) constituída por equipamentos que se ligam sem fios a um router wireless que por sua vez se liga ao switch, destinada a fornecer acesso a dispositivos móveis ligados sem fios como smartphones, tablets e computadores laptop.

  A segmentação de redes nesta arquitetura permite a implementação de diversos pontos de controlo.

Possíveis vulnerabilidades:

Ameaças à Confidencialidade:

- Privacidade

  Rastreamento não autorizado de atividades dos utilizadores

Exposição de padrões de navegação dos utilizadores
Monitorização básica de tráfego de email não encriptado
Exposição de metadados dos utilizadores por configurações incorretas

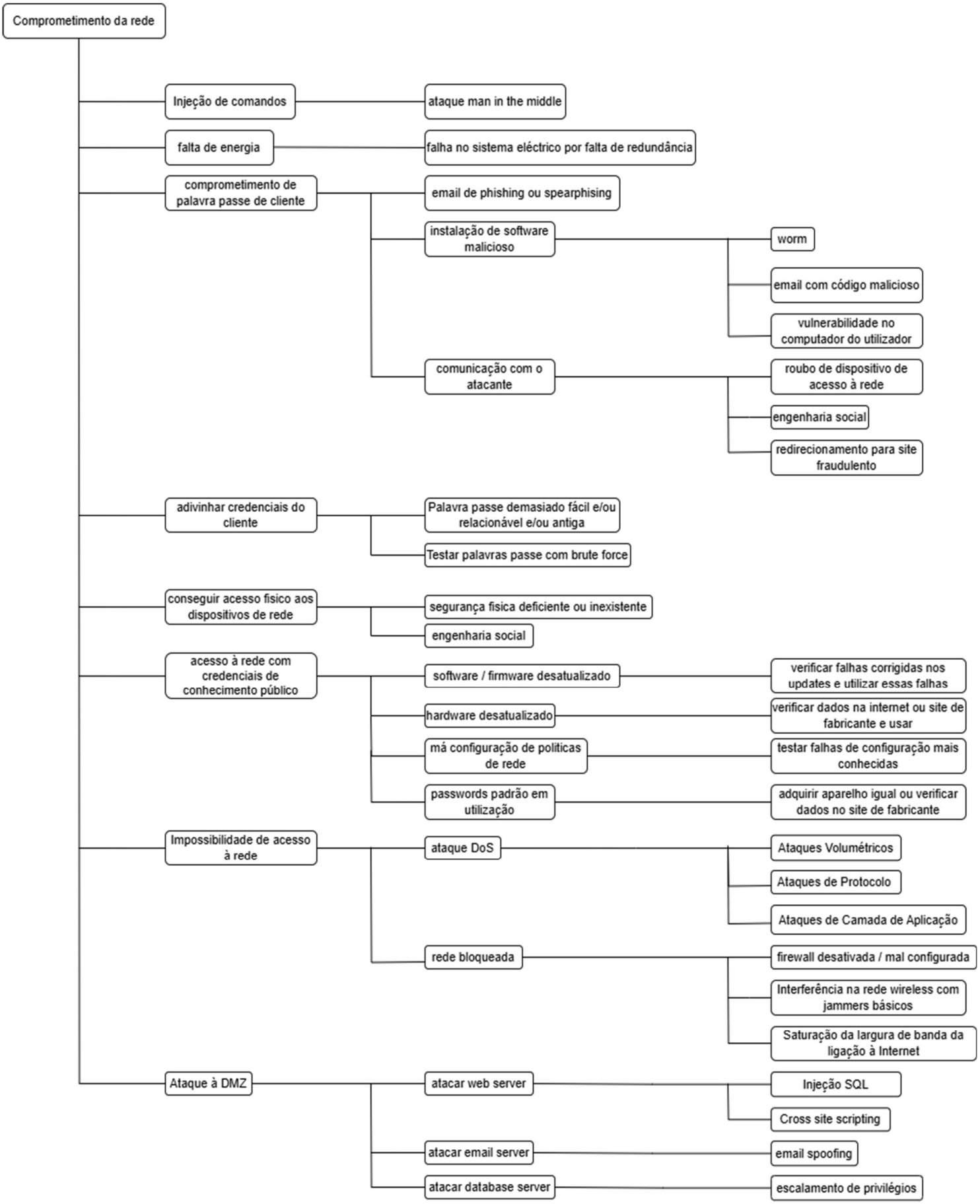
- Confidencialidade dos Dados:
  Interceção de tráfego na rede WLAN com ferramentas comuns
  Captura básica de pacotes na rede local (LAN)
  Acesso não autorizado aos servidores por passwords fracas
  Exploração de sessões não encriptadas
  Acesso físico não autorizado por falha de controlo de acessos
  Roubo de dispositivos móveis não protegidos

Ameaças à Integridade:

- Integridade dos Dados:
  Modificação não autorizada de registos na base de dados via password fraca
  Alteração de conteúdo no servidor web/aplicacional por falhas de autenticação
  Manipulação de emails por configurações incorretas
  Corrupção acidental de ficheiros durante a transmissão
  Alteração não autorizada de documentos por falta de controlo de versões
- Integridade do Sistema:
  Modificação não autorizada de configurações de rede devido a passwords padrão
  Comprometimento das regras simples da firewall
  Alteração de configurações do sistema operativo
  Instalação de software malicioso por falta de antivírus
  Manipulação das políticas de segurança

- Ameaças à Disponibilidade:
  Ataques DDoS direcionados ao router ou firewall
  Sobrecarga dos servidores
  Interferência na rede wireless com jammers básicos
  Saturação da largura de banda da ligação à Internet
  Falha no sistema de energia elétrica por falta de redundância

Diagrama de árvore de ataque



3. Proposta de controlo de segurança

Controlos propostos:

- Utilização de palavras passe fortes não padrão e renovação periódica das mesmas de modo a prevenir a adivinhação de credenciais, utilização de bruteforce ou simples pesquisa das mesmas por serem públicas.
- Antivírus atualizado de modo a prevenir a instalação de programas maliciosos, abertura de emails maliciosos ou simples exploração de vulnerabilidade do aparelho.
- Utilização de uninterruptible power supply (UPS) de modo a prevenir falhas de energia.
- Utilização de encriptação forte para prevenir intromissão nas comunicações por parte do atacante em ataques man in the middle, WEP/WPA cracking, desautenticação, pontos de acesso duplicados, etc.
- Atualização periódica de software e hardware de modo a prevenir vulnerabilidades conhecidas e/ou dados de acesso amplamente divulgados fora da organização.
- Contratar seguranças certificados e prevenir o acesso físico não autorizado aos dispositivos críticos da rede.
- Formação a todos os clientes e funcionários sobre engenharia social de modo a prevenir ataques de engenharia social mais comuns.
- Configurar corretamente as políticas de acesso à rede, especialmente na firewall, de modo a prevenir ataques à DMZ, ataques DoS e controlar os acessos.
- Ter um plano de contingência e de resposta a incidentes para o caso de algum ataque à rede ser bem sucedido ou existir outro problema.

4. Implementação da solução criptográfica

- **Encriptação simétrica:**

```
from cryptography.hazmat.primitives.ciphers import Cipher, algorithms, modes
from cryptography.hazmat.primitives import padding
from cryptography.hazmat.backends import default_backend
import os
#Encripta a mensagem com Advanced Encryption Standard em modo Cipher-block chaining.
# Função para encriptar
def encriptar_simetrico(chave, texto_original):
    vetor_inicializacao = os.urandom(16)  # vetor de inicialização bloco de 16 bytes para AES
    cifrador = Cipher(algorithms.AES(chave), modes.CBC(vetor_inicializacao),
    backend=default_backend())              # Criar o cifrador
    encriptador = cifrador.encryptor()
    gerador_padding = padding.PKCS7(128).padder()      # Adicionar padding ao texto original
    dados_preenchidos = gerador_padding.update(texto_original) + gerador_padding.finalize()
```

```
    texto_cifrado = encriptador.update(dados_preenchidos) + encriptador.finalize()
# Encriptar os dados
    return vetor_inicializacao, texto_cifrado

def desencriptar_simetrico(chave, vetor_inicializacao, texto_cifrado):      # Função para desencriptar
    cifrador = Cipher(algorithms.AES(chave), modes.CBC(vetor_inicializacao),
    backend=default_backend())    # Criar o decifrador
    desencriptador = cifrador.decryptor()
    dados_preenchidos = desencriptador.update(texto_cifrado) + desencriptador.finalize()
        #Desencriptar os dados
    removedor_padding = padding.PKCS7(128).unpadder()    # Remover o padding
    texto_original = removedor_padding.update(dados_preenchidos) + removedor_padding.finalize()
    return texto_original

# teste
chave = b'chaveSecreta1234'              # Chave de 16 bytes (128 bits) para AES
texto_original = b'texto secreto'
vetor_inicializacao, texto_cifrado = encriptar_simetrico(chave, texto_original)
print(f"Texto encriptado: {texto_cifrado}")
texto_desencriptado = desencriptar_simetrico(chave, vetor_inicializacao, texto_cifrado)
print(f"Texto desencriptado: {texto_desencriptado}")
```

A implementação consiste em gerar uma chave de valor aleatório com 128 bits (16 bytes), exigido para a configuração do algoritmo AES no modo CBC e esta terá que ser mantida em sigilo, pois a segurança da encriptação depende dela.

É, também, gerado um vetor de Inicialização de valor inicial aleatório de 16 bytes para introduzir aleatoriedade na encriptação, o que garante que o mesmo texto com a mesma chave produza resultados diferentes.

Na encriptação, o texto original é preenchido com o padrão PKCS7 para ser um múltiplo do tamanho do bloco 16 bytes, e o objeto de encriptação é Advanced Encryption Standard em modo Cipher-block chaining, gerando desta forma a encriptação em blocos num texto cifrado.

O texto original é definido como uma string de bytes e o seu comprimento será ajustado através da padding para um múltiplo do tamanho do bloco AES (16 bytes) por razões de compatibilidade com o algoritmo.

Na decriptação o objeto de desencriptação AES é configurado usando a mesma chave e o mesmo vetor de inicialização, o texto cifrado é processado para retornar ao estado preenchido, e no fim o padding é removido, tendo como resultado o texto original.

Vantagens da encriptação simétrica:

- Desempenho: Execução muito rápida comparada com criptografia assimétrica porque requer menos poder computacional.
- Simplicidade: Algoritmos mais simples de implementar, visto que tem menor complexidade computacional.
- Segurança: Chaves mais curtas que na assimétrica para mesmo nível de segurança

Desvantagens da encriptação simétrica:

- Gestão de chaves: Necessidade de compartilhar a chave secreta de forma segura e pode ser difícil gerir muitas chaves entre muitos utilizadores.
- Limitações de Uso: Requer canal seguro inicial para troca de chaves.
- Riscos de Segurança: Mais vulnerável a ataques de força bruta que assimétrica

- **Encriptação assimétrica:**

```
from cryptography.hazmat.primitives.asymmetric import rsa, padding
from cryptography.hazmat.primitives import serialization, hashes

def gerar_par_chaves_rsa():                    # Geração do par de chaves RSA
    chave_priv = rsa.generate_private_key(
        public_exponent=65537,  # Expoente público padrão
        key_size=4096,          # Tamanho da chave em bits
    )
    chave_pub = chave_priv.public_key()
    return chave_priv, chave_pub

def encriptar_mensagem(mensagem, chave_pub):   # Encriptação com OAEP padding
    mensagem_cifrada = chave_pub.encrypt(
        mensagem,
        padding.OAEP(
            mgf=padding.MGF1(algorithm=hashes.SHA256()),
            algorithm=hashes.SHA256()
        )
```

```
    )
    return mensagem_cifrada

def desencriptar_mensagem(mensagem_cifrada, chave_priv): # Desencriptação com OAEP padding
    mensagem_original = chave_priv.decrypt(
        mensagem_cifrada,
        padding.OAEP(
            mgf=padding.MGF1(algorithm=hashes.SHA256()),
            algorithm=hashes.SHA256()
        )
    )
    return mensagem_original

def executar_teste_rsa():    # Execução principal
    chave_privada, chave_publica = gerar_par_chaves_rsa()   # Gerar as chaves
    mensagem_original = b"Mensagem confidencial"
    # encriptação
    dados_encriptados = encriptar_mensagem(mensagem_original, chave_publica)
    # desencriptação
    dados_desencriptados = desencriptar_mensagem(dados_encriptados, chave_privada)
    # Resultados
    print(f"Mensagem original: {mensagem_original}")
    print(f"Dados encriptados (hex): {dados_encriptados.hex()}")
    print(f"Mensagem desencriptada: {dados_desencriptados}")

# Executar o teste
executar_teste_rsa()
```

A implementação consiste em 4 funções para gerar chaves, encriptar, desencriptar e testar o algoritmo.

A primeira função consiste em gerar uma chave pública e uma chave privada de acordo com o algoritmo RSA, aqui foram utilizados os valores public_exponent=65537 que é comum para RSA, e uma chave de 4096 bits por ser grande o suficiente para uma boa segurança.

A função de encriptação é realizada com a chave pública e é usado o preenchimento OAEP que por sua vez usa o algoritmo SHA256 para segurança e a geração de máscara MGF1.

A função de desencriptação é realizada com a chave privada usando tendo em conta o mesmo preenchimento OAEP previamente configurado para desencriptar.

A última função consiste somente em mostrar os resultados das outras funções.

Este método tem como base de confiança o fato de ser bastante complexo fatorizar números primos grandes.

Vantagens da encriptação assimétrica:

- Gestão de chaves: A execução é muito rápida comparada com criptografia assimétrica porque requer menos poder computacional.
- Escalabilidade: É mais fácil de usar em grandes sistemas.
- Segurança: É mais segura para estabelecer comunicação inicial

Desvantagens da encriptação assimétrica:

- Desempenho: É muito mais lenta que criptografia simétrica.
- Complexidade: Os algoritmos são mais complexos pelo que requerem mais conhecimento técnico.
- Tamanho das Chaves: Chaves muito maiores que na simétrica, causam maior consumo de banda.

5. Conclusão e reflexão sobre a estratégia

Resumo das medidas:
Gestão de credenciais (passwords fortes e renovação)
Proteção contra malware (antivírus)
Proteção física (UPS e segurança física)
Segurança de comunicações (encriptação)
Manutenção de sistemas (atualizações)
Controle de acesso físico
Formação de pessoal (anti-engenharia social)
Configuração de rede (firewall e políticas)
Planeamento de contingência

De um ponto de vista de gestão de risco, não existe risco zero, pelo que o sistema nunca será 100% eficaz. Porém, na minha proposta existem alguns pontos fortes visto que existe uma abordagem holística multicamadas na medida em que são tidos em conta aspetos técnicos, físicos e humanos, assim como foco em prevenção e resposta a incidentes.

Apostei também no aspeto de resiliência com o uso da UPS e formação do pessoal da empresa para ser mais fácil recuperar de ataques e falhas ou pelo menos minimizar o dano causado.

Alguns aspetos a melhorar, seriam a utilização de encriptação assimétrica, mais medidas contra ataques internos, monitorização de logs na rede, testes regulares de red team e auditorias assim como backups da informação. Em nenhum cenário me pareceu existir uma resposta 100% eficaz para qualquer rede, especialmente porque existe o fator humano que parece ser uma das maiores vulnerabilidades.

6. Referências

Stallings, W., Brown, L. (2024). Computer Security: Principles and Practice. 5th Edition (Global Edition), Pearson.

P. Santos, C. Pimentel, R. Bessa (2008). Cyberwar O Fenómeno, as Tecnologias e os Actores

https://owasp.org/

https://www.nist.gov/

slides PDF e fórum da disciplina

https://null-byte.wonderhowto.com/

https://www.youtube.com/watch?v=vq9oLUdBTrQ

https://mothereff.in/byte-counter

https://pt.wikipedia.org/wiki/RSA_(sistema_criptogr%C3%A1fico)

https://www.ibm.com/docs/en/linux-on-systems?topic=formats-rsa-public-key-token

https://onboardbase.com/blog/aes-encryption-decryption/

https://nitratine.net/blog/post/asymmetric-encryption-and-decryption-in-python/

https://www.teach.cs.toronto.edu/~csc110y/fall/notes/08-cryptography/05-rsa-cryptosystem-implementation.html

https://www.geeksforgeeks.org/rsa-algorithm-cryptography/

https://pycryptodome.readthedocs.io/en/latest/src/cipher/aes.html

https://stackoverflow.com/questions/12524994/encrypt-and-decrypt-using-pycrypto-aes-256

https://cryptography.io/en/latest/hazmat/primitives/symmetric-encryption/

https://en.wikipedia.org/wiki/Advanced_Encryption_Standard

https://learn.microsoft.com/en-us/dotnet/api/system.security.cryptography.aes?view=net-9.0

https://learn.microsoft.com/en-us/dotnet/api/system.security.cryptography.rsa?view=net-9.0

https://www.precisely.com/blog/data-security/aes-vs-rsa-encryption-differences

http://web.eecs.umich.edu/~valeria/research/publications/DATE10RSA.pdf

https://www.youtube.com/watch?v=GYCVmMCRmTM

https://www.youtube.com/watch?v=txz8wYLITGk&t=226s

https://www.youtube.com/watch?v=3MPkc-PFSRI

https://www.youtube.com/watch?v=34BtwcL7Mkg&t=835s