

Segurança Informática

Trabalho 2

Parte 1

Exercício 1

a)

*Perfect forward secrecy*(PFS)é a propriedade do *protocolo handshake* que garante que, se a chave privada for comprometida, não é possível decifrar *master secret* anteriores. Contudo, ao usar o esquema de cifra assimétrica *RSA* o *perfect forward secrecy*(PFS) não é garantido devido à forma como as chaves são usadas durante o processo de aperto de mão.

Quando o RSA é usado no TLS para troca de chaves, o cliente gera uma *pre master key*, criptografa essa chave com a chave pública do servidor e a envia. O servidor, então, usa sua chave privada para desencriptar a chave *pre master key*.

O problema com essa abordagem em termos de *perfect forward secrecy*(PFS) é que, se um atacante em algum momento conseguir obter a chave privada do servidor, ele pode retroativamente desencriptar todas as sessões passadas que foram protegidas com a *pre master key* compartilhada. Isso comprometeria a confidencialidade dessas comunicações antigas.

b)

Dois possíveis ataques ao *record protocol*:

1. Ataque de Falsificação de Registos (*Record Forgery Attack*):

**Ataque:** Um atacante tenta forjar registos TLS/SSL para enviar informações falsas.

**Prevenção:** Uso de algoritmos de assinatura digital para verificar a autenticidade dos registos. Certificados digitais e cadeias de certificados são usados para autenticação mútua.

1. Ataque de Interceção (*Man-in-the-Middle - MitM*):

**Ataque:** Um atacante intercepta as comunicações entre o cliente e o servidor.

**Prevenção:** Autenticação mútua através de certificados digitais, onde o cliente e o servidor autenticam um ao outro, ajuda a garantir que a comunicação seja estabelecida com entidades confiáveis.

Exercício 2

Tendo em conta o contexto dado e a informação de validação das *passwords* descritas no exercício o uso de *CAPTCHA* pode sim ajudar a mitigar ataques de dicionário à informação de validação. Isso acontece, pois, o *CAPTCHA* tem como objetivo distinguir entre humanos e computadores, dando tarefas fáceis para diferenciá-los, dessa maneira o mesmo torna mais difícil com que programas automatizados submetam várias tentativas de senha, por exigir interação humana. *CAPTCHA* adiciona também uma camada de proteção contra ataques de dicionário, porque mesmo obtendo o *hash*, a *password* e o número aleatório o usuário continuaria a ter de resolver a tarefa do *CAPTCHA*.

Exercício 3

O servidor pode detetar se o conteúdo do *cookie* foi adulterado no navegador verificando a assinatura do *JSON Web Token* (JWT). O servidor, ao criar o JWT, assina digitalmente usando uma chave secreta. Ao receber o *cookie* de volta, o servidor verifica a assinatura para garantir que o *token* não foi alterado. Se a assinatura não corresponder, quer dizer que o conteúdo do cookie foi modificado.

Exercício 4

Exercício 5

Parte 2

Exercício 6

Exercício 7