Grupo 7 – Segurança Informática

Autores: André Graça nº 47224, Diogo Santos nº 48459

1-

a) A propriedade Perfect Forward Secrecy não é garantida usando o processo base com RSA para estabelecimento do master secret porque se o master secret for comprometido, isto é, um atacante tiver acesso a este, ele consegue decifrar todas as mensagens desta sessão e de todas as outras sessões anteriores e futuras entre os 2 endpoints, pois o master secret é usado para decifra e cifra de ambos os endpoints. Para que esta propriedade seja garantida é necessário gerar um master secret por sessão, para garantir que se uma dos master secrets seja comprometido, a única sessão que o atacante tem acesso é a onde esse master secret foi usado, as sessões anteriores e futuras continuam inacessíveis para outros para alem dos endpoints.

b) Embora o Record Protocol possa parecer uma simples aplicação de esquemas de cifra e MAC, há alguns detalhes específicos que o tornam imune a certos tipos de ataque:

Ataque de repetição

Um dos possiveis ataques é o ataque de repetição, onde o atacante observa as mensagens trocadas entre as partes legítimas da comunicação e escolhe reenviar uma ou várias num momento oportuno. Apesar das mensagens estarem protegidas por esquemas criptográficos para os quais o atacante não possui as chaves corretas, o reenvio de uma mensagem capturada poderia ser aceite pelas partes legítimas da comunicação, já que o seu conteúdo foi gerado a partir das chaves adequadas, comprometendo assim a integridade dos dados (já que a porções de informações seriam processadas duas vezes pelo recetor). Para se proteger contra este ataque o TLS usa uma tag de autenticação da mensagem que é computada sobre o seu número de sequência original. Assim, caso o atacante capture a mensagem e, logo em seguida, a repita como se fosse uma nova mensagem com número de sequência incrementado. Quando o recetor a receber, ele realiza a verificação da tag, como este número de sequência é diferente daquele utilizado originalmente na geração da marca, a verificação falha, o que demonstra alguma troca de posição da mensagem.

Ataque de reflexão

Outro ataque é o ataque de reflexão. Onde o atacante intercepta uma mensagem transmitida do endpoint A para o endpoint B e transmite-a no sentido oposto. Como esta mensagem foi legitimamente gerada por A, resultante da aplicação de esquemas criptográficos calculados com chaves corretas, então a mensagem deveria passar pelas verificações de integridade e autenticidade. Para solucionar isto o TLS faz uso de material criptográfico diferente para cada sentido da comunicação. Como as chaves e IV utilizados no sentido de A para B são diferentes daqueles usados no sentido oposto, a verificação da marca de autenticação falhará e A facilmente detetará a tentativa de ataque.

2- A técnica CAPTCHA (Completely Automated Public Turing Test to Tell Computers and Humans Apart) é usada para distinguir utilizadores humanos de programas automatizados, como bots. Ela geralmente envolve a apresentação de desafios visuais ou problemas que são fáceis para os seres humanos resolverem, mas difíceis para computadores. Se o atacante tentar realizar um ataque á interface de autenticação que usufrui da técnica CAPTCHA este irá demorar mais tempo a realizar cada tentativa de autenticação, pois o custo de completar um CAPTCHA para um computador é demorado em relação a um humano, isto inviabiliza o ataque de dicionário visto que o tempo para realizar este ataque usando cada entrada do dicionário iria demorar mais tempo.

3- Para garantir que o cookie que guarda o token do utilizador não é adulterado por nenhuma autoridade diferente no browser podemos proteger o cookie usando um MAC, caso a assinatura não coincidir no momento de rececão do servidor então o conteúdo do cookie foi alterado.

4-

a) O JWT ( Json Web Token ) é usado quando a autenticação do usuário e a aplicação estão em servidores diferentes, este é gerado pelo servidor de autorização com informações sobre o usuário, para manter a sessão do usuário na aplicação.

b) Após o dono de recursos ter autorizado e consentido o uso de um dos seus recursos, este irá reencaminhar o utilizador de volta para aplicação cliente, mandando um código temporário na query string (redirect\_uri). A aplicação cliente irá usar este código temporário para fazer um pedido para o servidor de autorização, para obter o token relativo a este utilizador. Com este token o usuário pode realizar todas as operações permitidas nos scopes na criação do token.

5- Do enunciado:

Uma imagem com círculo, captura de ecrã, preto e branco

Descrição gerada automaticamenteApós a análise do conjunto das relações user assigment (UA) podemos concluir que o utilizador tem o papel de (S)upervisor, pois neste conjunto está o par . Se analisarmos o conjunto das relações permission assigment (PA), apenas com esta informação, não conseguimos determinar as operações que um (S)upervisor pode realizar, pois não existe nenhum par com (S) neste conjunto. Visto que o modelo de controlo de acessos devemos analisar a hierarquia de roles (RH):

Com a análise do grafo acima podemos concluir que um (S)upervisor pode realizar as mesmas operações que os (T)esters , os (D)evelopers e os (M)embers, que são as operações p1, p2 e p3.

6- Configurações iniciais:

1. Instalar o certificado “CA1.cer” no browser em Autoridades de certificação de raiz fidedigna, pois é este certificado que assinou o certificado “CA1-int.cer”, e este assinou o certificado do servidor.
2. Editar o ficheiro hosts (C:\Windows\System32\drivers\etc\hosts) e adicionar a linha “127.0.0.1 [www.secure-server.edu](http://www.secure-server.edu)”.
3. Na variável “options” defenir como certificado do servidor e chave privada do servidor os ficheiros fornecidos pelo docente que são o “secure-server-17Nov.pem” e “secure-server-key-17nov.pem”, respetivamente. O certificado do servidor é necessário durante o TLS para o cliente verificar que o servidor é uma entidade legítima e de confiança, para isso não basta apenas o certificado “secure-server-17nov.pem”, é necessário fornecer os certificados necessários para construir a cadeia. Temos 2 formas de alcançar este objetivo:
   1. Instalar no browser o certificado “CA1-int.cer”, com este e o certificado raiz “CA1.cer” o browser consegue verificar que o certificado fornecido pelo servidor é legitimo pois este foi gerado pelo certificado “CA1-int.cer”, e o cliente consegue construir a cadeia.
   2. Adicionar ao certificado do servidor o certificado “CA1-int.cer” em formato .pem, assim basta apenas o cliente ter instalado a raiz “CA1.cer” para ser feito a cadeia.

Concluindo, é necessário fornecer ao cliente o certificado “CA1-int.cer” para que este consiga fazer a verificação do certificado fornecido pelo servidor, o nosso grupo decidiu adotar a segunda solução, o ficheiro “secure-server-chain.pem” contêm o certificado do servidor e o certificado “CA1-int.cer” em formato .pem. Para converter o certificado em um ficheiro .pem usamos o comando *./openssl x509 -inform der -in CA1-int.cer -out CA1-int.pem*.

Sem autenticação:

1. Executar o comando *node https-server-unauthorized-server.js*.
2. Colocar na barra de endereços do browser <https://secure-server.edu:4433>.

Com autenticação:

1. Executar o comando node *https-server-authorized-server.js*.
2. Instalar nos certificados pessoais, o certificado da alice\_2, que é o ficheiro alice\_2.pfx.
3. Colocar na barra de endereços do browser <https://secure-server.edu:4433>.

7.

Uma imagem com texto, captura de ecrã, Tipo de letra, círculo

Descrição gerada automaticamenteOnde p1 é as permissões de visualização de tarefas e p2 é as permissões de visualização e criação de novas tarefas a partir das milestones do GitHub.