Ciências ULisboa | Informática

di.ciencias.ulisboa.pt

/difcul

Segurança e Confiabilidade

2016/2017

Trabalho 2

1 Objetivos

Esta fase do trabalho estende a anterior, possibilitando aos alunos experimentarem diversos mecanismos de segurança, tais como: cifras, assinaturas, comunicação com um protocolo seguro (TLS – Transport Layer Security) e gestão básica de certificados.
A envolvente do trabalho continua a ser a mesma, ou seja, a concretização de um sistema **simplificado** de controlo de versões de ficheiros, myGit. O trabalho será realizado utilizando a linguagem de programação Java e a API de segurança do Java, e ferramentas complementares. Neste trabalho, tal como no anterior, o cliente e o servidor devem ser executados dentro da sandbox.

2 Modelo de adversário

Iremos assumir no trabalho que existe um adversário que pretende comprometer o correto funcionamento do sistema. O adversário terá um conjunto de capacidades que poderão ser empregues na realização das suas ações maliciosas. Torna-se assim necessário dotar o sistema dos mecanismos de proteção que lhe possibilitem manter um funcionamento correto ainda que se encontre sob ataque.

Vamos assumir que o adversário tem as seguintes capacidades:

- Acesso à rede : tendo o adversário acesso à rede, poderá escutar os pacotes trocados entre o cliente e o servidor. Potencialmente, também poderá tentar corromper, alterar, introduzir, e reproduzir mensagens de forma a enganar quer o cliente quer o servidor.

- Controlar um ou mais utilizadores : o adversário controla uma (ou mais) conta(s) de utilizadores do sistema. Através desta(s) conta(s), ele poderia tentar aceder a ficheiros para os quais não tem permissões ou corromper ficheiros com informação de outros utilizadores.

- Acesso à máquina onde corre o servidor em modo de leitura : o adversário tem acesso em modo de leitura aos ficheiros armazenados no servidor. Com esse acesso, ele pode potencialmente observar informação que eventualmente seria confidencial.

Em seguida indicam-se e discutem-se as proteções que devem ser adicionadas ao sistema.

3 Proteções a adicionar ao sistema

Nesta fase, os alunos devem usar a mesma arquitectura da 1ª fase, e as funcionalidades a oferecer mantêm-se aproximadamente iguais. No entanto, o sistema será estendido de modo a ser garantida a sua segurança.

As alterações a introduzir são as seguintes:

1. O servidor deve **autenticar os utilizadores**. O utilizador durante a autenticação fornece o username e a password. Esta informação deve ser transmitida para o servidor protegida de ataques na rede (ver abaixo). O servidor deve validá-la, e apenas se esta estiver correta deve ser dado acesso ao utilizador.
2. O servidor deve proteger a **integridade do ficheiro das passwords**. Para tal, o ficheiro deve ser protegido com um MAC. O cálculo deste MAC utiliza uma chave simétrica calculada a partir de uma password que é pedida ao utilizador quando **inicia a execução do servidor**. No início da sua execução, o servidor deve usar o MAC para verificar a integridade do ficheiro. Se o MAC estiver errado, o servidor deve imprimir um aviso e terminar imediatamente a sua execução. Se não há MAC a proteger o ficheiro, o servidor deve imprimir um aviso, calcular o MAC e adicioná-lo ao sistema. O MAC deve ser verificado em todos os restantes acessos ao ficheiro e atualizado caso o ficheiro seja alterado. O MAC pode ser guardado num ficheiro utilizado apenas para este efeito.
3. O servidor também deve proteger a **confidencialidade das passwords**. Com este objetivo, o conteúdo do ficheiro das passwords deve ser mantido cifrado com uma chave calculada a partir da mesma password que é pedida no início da execução do servidor.

   Adicionalmente, de modo a evitar que as passwords sejam transmitidas em claro, quando um utilizador pretende autenticar-se, são efetuados os seguintes passos: (1) o servidor envia um *nonce* ao cliente, (2) o cliente calcula a síntese da password do utilizador concatenada com o *nonce*, (3) o cliente envia esta síntese para o servidor, (4) o servidor faz um cálculo semelhante da síntese usando a cópia da password que armazena localmente, e compara com a síntese recebida para autenticar o utilizador.
4. O(s) ficheiro(s) onde é mantida informação relativa aos utilizadores que têm acesso aos vários repositórios deve estar protegido em relação à **integridade**. Para tal, o servidor deve usar um MAC criado e mantido de maneira semelhante ao do ficheiro de passwords.
5. Na comunicação entre o cliente e o servidor pretende-se garantir a **autenticidade do servidor** (um atacante não deve ser capaz de fingir ser o servidor e assim obter a password de um utilizador) e a **confidencialidade** da comunicação entre cliente e servidor (um atacante não deve ser capaz de escutar a comunicação). Para este efeito, devem-se usar **canais seguros** (protocolo TLS/SSL). Este protocolo permite verificar a identidade do servidor utilizando chaves assimétrica.

   - Ligações TLS: Deve-se substituir a ligação TCP por uma ligação TLS/SSL. O protocolo TLS vai verificar a autenticidade do servidor e garantir a integridade e confidencialidade de toda a comunicação.
   - A utilização do protocolo TLS exige configurar as chaves tanto no cliente (truststore com o certificado do servidor) como no servidor (keystore com a sua chave privada).

6. Os ficheiros dos repositórios devem ser protegidos de eventuais ataques que possam ocorrer na máquina servidora, nomeadamente que tenham em vista **observar o seu conteúdo ou personificar a origem**. Para isso, deverão utilizar criptografia híbrida e assinaturas digitais (em alternativa a envelopes seguros para simplificar a implementação). A ideia seria aplicar genericamente o seguinte algoritmo quando se pretende enviar/actualizar um ficheiro no servidor:
    I. O cliente gera a assinatura digital do ficheiro em claro e envia-a para o servidor. O servidor guarda-a num ficheiro com extensão .sig
    II. O cliente gera aleatoriamente uma chave simétrica *K* para o algoritmo AES e cifra o ficheiro com *K* usando AES
    III. O cliente envia para o servidor a chave K e o ficheiro cifrado
    IV. O servidor cifra K usando a sua chave pública. O servidor armazena a chave cifrada num ficheiro separado com extensão *.key.server* (por ex., o ficheiro *fich.txt* do repositório da Alice poderia ter a chave armazenada em *fich.txt.key.server*)

    Quando um utilizador lê o ficheiro do servidor, deve ser executado o processo inverso ao definido anteriormente:

    I. O servidor obtém o respectivo ficheiro com a chave K cifrada (com extensão *.key.server)*. Em seguida, usa a sua chave privada para decifrar o conteúdo e ficar com K
    II. O servidor envia para o cliente o conteúdo do ficheiro cifrado e a chave K
    III. O cliente usa K para decifrar o conteúdo do ficheiro
    IV. O servidor envia para o cliente a assinatura do ficheiro
    V. O cliente verifica a assinatura do ficheiro
7. Quando é dado acesso a um repositório a um outro utilizador, este deve passar a ter acesso a todos os ficheiros.
8. Quando se retira o acesso dum utilizador a um repositório, este deixa de poder alterar o repositório no servidor ou de receber novas actualizações de ficheiros que venham a ocorrer.

NOTA: Toda criptografia assimétrica no projeto deve usar RSA com chaves de 2048 bits. A criptografia simétrica deve ser efetuada com AES e chaves de 128 bits. O algoritmo de síntese deve ser o SHA-256.

4 Relatório e discussão

No relatório devem ser apresentados e discutidos os seguintes aspetos:

- Os objetivos concretizados com êxito
- Os problemas encontrados.
- A segurança da aplicação criada, identificando possíveis **fraquezas e melhorias** a incluir em versões futuras.

O relatório deve ter no máximo 5 páginas (sem contar com o código) e **é necessário**

incluir o código fonte na versão em papel.

5 Entrega

1. **Código.** Dia **30 de Abril**, até as 23:55 horas. O código do trabalho deve ser entregue da seguinte forma:
    1. Os grupos devem inscrever-se atempadamente de acordo com as regras afixadas para o efeito, na página da disciplina.
    2. Na página da disciplina submeter o código do trabalho num ficheiro zip.

2. **Relatório.** Dia **2 de Maio**, até as 18:00 horas.
    1. Na página da disciplina submeter o relatório num ficheiro pdf e um readme (txt) sobre como executar o trabalho.
    2. No cacifo do professor das TPs, a impressão do relatório e do código fonte.

Não serão aceites trabalhos por email nem por qualquer outro meio não definido nesta secção. Se não se verificar algum destes requisitos o trabalho é considerado não entregue.