Curso de Extensão em Tecnologias Microsoft

INF0998 Programação segura (segurança de software)

Atividades práticas de criptografia aplicada

As seguintes ferramentas são usadas ao longo das tarefas:

- Cryptool Online https://www.cryptool.org/en/cto
- Cryptool para Windows https://www.cryptool.org/en/ct1
- JCryptool https://www.cryptool.org/en/jct/
- OpenPGP, GPG4win e Kleopatra https://www.openpgp.org/software
- OpenSSL (Usar o openssl da sua distribuição Linux preferida)
- Apache web server (Usar a sua distribuição preferida)

Atividades com o OpenSSL

Cálculo de hash da string vazia

Usar o openssl para calcular o valor hash da string vazia (“”) em vários algoritmos diferentes.

```
C:\>echo "" | openssl dgst -blake2s256
```

```
C:\>echo "" | openssl dgst -md4 -r
```

```
C:\>echo "" | openssl dgst -md5 -r
```

```
C:\>echo "" | openssl dgst -sha1 -r
```

Repetir a tarefa para a string “ABC” como o exemplo

```
C:\>echo "ABC" | openssl dgst -blake2s256
```

Repetir a tarefa para o seu nome.

Cálculo e verificação de MAC com OpenSSL

Esta atividade usa a linha de comando do OpenSSL para ilustrar o cálculo de códigos de autenticação de mensagens e a combinação destes com a encriptação.

Cálculo e verificação de MAC

Calcular uma tag HMAC SHA256 para ao arquivo CancaoDoExilio.txt.

```
C:\> openssl dgst -sha256 -hmac 00112233445566778899aabbccddeeff -out tag1.hmac CancaoDoexilio.txt
```

O OpenSSL não possui um comando de verificação de tags. A verificação pela comparação de duas tags (uma recebida e outra calculada) pode ser feita com o comando `fc`. Verificar a tag calculada anteriormente.

```
C:\> openssl dgst -sha256 -hmac 00112233445566778899aabbccddeeff -out tag2.hmac CancaoDoexilio.txt
```

```
C:\> fc tag1.hmac tag2.hmac
```

Mudanças na tag, no texto claro ou na chave fazem a verificação falhar. Causar uma mudança na chave ou na tag recebida e tentar verificar a validade de tag.

```
C:\> openssl dgst -sha256 -hmac aa112233445566778899aabbccddeeff -out tag3.hmac CancaoDoexilio.txt
```

```
C:\> fc tag1.hmac tag3.hmac
```

Opcionalmente, repetir a tarefa para outras funções de HMAC disponíveis no openssl. Modificar os comandos anteriores para utilizar outras funções de hash, tais como o sha512 e sha3-256.

Encriptação autenticada “Encripta e MAC”

A encriptação autenticada por combinação manual é usada na tarefa.

Esta atividade utiliza o método “Encripta e Autentica”. A geração da tag e a encriptação do texto claro procedem do seguinte modo, onde os comandos podem ser executados em qualquer ordem.

1. Encriptar com AES-128-CTR o arquivo CancaoDoExilio.txt na pasta *Documents*.

```
C:\> openssl enc -aes-128-ctr -in CancaoDoExilio.txt -out
criptograma.aes -K 00112233445566778899aabbccddeeff -iv
00998877665544332211ffeeddccbbaa
```

2. Calcular uma tag hmac do texto claro.

```
C:\> openssl dgst -sha256  -hmac aabbccddeeff00112233445566778899  -
out CancaoDoExilio.hmac CancaoDoExilio.txt
```

No papel do destinatário que recebe a tag e o criptograma, realizar a verificação da tag e a decriptação do criptograma.

1. Executar o comando para decriptação com AES-128-CTR utilizando a mesma chave e mesmo IV usados na encriptação e obter uma cópia do texto claro.

```
C:\> openssl enc -d -aes-128-ctr -in criptograma.aes -K
00112233445566778899aabbccddeeff -iv 00998877665544332211ffeeddccbbaa
-out CancaoDoExilio2.txt
```

2. Calcular a tag hmac do texto claro obtido da decriptação.

```
C:\> openssl dgst  -sha256  -hmac aabbccddeeff00112233445566778899  -
out CancaoDoExilio2.hmac CancaoDoExilio2.txt
```

3. Em seguida, comparar a tag calculada com a tag recebida. Se não houver diferenças entre os arquivos hmac, a tag foi verificada com sucesso.

```
C:\> fc CancaoDoExilio2.hmac CancaoDoExilio.hmac
```

4. O texto claro recuperado prematuramente é considerado confiável somente após a verificação bem-sucedida da tag.

Encriptação autenticada “Encripta então MAC”

Esta atividade utiliza o método “Encripta então Autentica”. A geração da tag e a encriptação do texto claro procedem do seguinte modo.

1. Primeiro, encriptar com AES-128-CTR o arquivo CancaoDoExilio.txt.

```
C:\> openssl enc -aes-128-ctr -in CancaoDoExilio.txt -out
criptograma.aes -K 00112233445566778899aabbccddeeff -iv
00998877665544332211ffeeddccbbaa
```

2. Em seguida, calcular a tag hmac do criptograma produzido pelo comando anterior.

```
C:\> openssl dgst -sha256 -hmac aabbccddeeff00112233445566778899 -out
criptograma.hmac criptograma.aes
```

No papel do destinatário que recebe a tag e o criptograma, realizar a verificação da tag e a decriptação do criptograma.

1. Primeiro, calcular a tag hmac do criptograma recebido.

```
C:\> openssl dgst -sha256 -hmac aabbccddeeff00112233445566778899 -out
criptograma2.hmac criptograma.aes
```

2. Em seguida, comparar com a tag hmac recebida. Se não houver diferenças entre os arquivos hmac, a tag foi verificada com sucesso.

```
$ fc criptograma.hmac criptograma1.hmac
```

3. Finalmente, recomenda-se decriptar o criptograma somente se a tag for verificada com sucesso. Para tal, executar o comando para decriptação com AES-128-CTR utilizando a mesma chave e mesmo IV usados na encriptação.

```
C:\> openssl enc  -d -aes-128-ctr  -in criptograma.aes  -K
00112233445566778899aabbccddeeff -iv 009988776655443322 11ffeeddccbbaa
```

4. O resultado desse comando de decriptação é a exibição do texto claro diretamente na janela do terminal.

Teste SSL/TLS com OpenSSL

O OpenSSL pode ser usado diretamente para realizar muitos dos testes automáticos feitos por outras ferramentas. Esta atividade usa o cliente SSL do OpenSSL para descobrir se um servidor já aceita conexões no TLSv1.3 ou só permite conexões até o TLS1.2. Além disso, ainda faz testes por suites criptográficas específicas.

O comportamento padrão do comando s_client é tentar uma conexão na versão mais alta do protocolo disponível na instalação do OpenSSL. Versões específicas podem ser usadas explicitamente com as opções `-ssl2`, `-ssl3`, `-tls1`, `-tls1_1`, `-tls1_2` e `-tls1_3` (somente em distribuições novas). Além disso, versões específicas podem ser excluídas com as opções `-no_ssl2`, `-no_ssl3`, `-no_tls1`, `-no_tls1_1`, `-no_tls1_2`. O comando a seguir testa por conexões na versão 1.2.

```
C:\> openssl s_client -connect <<url>> -port 443 -tls1_2
```

O comando a seguir testa por conexões na versão 1.3.

```
C:\> openssl s_client -connect <<url>> -port 443 -tls1_3
```

Testar esses comandos para cada uma das URLs a seguir: www.amazon.com, www.yahoo.com, www.google.com, www.facebook.com.

- Quais conectam na versão 1.2?
- Quais conectam na versão 1.3?
- Quais as suites criptográficas selecionadas em cada caso?

O suporte a algoritmos específicos pode ser testado pela opção `-cipher` do comando `s_client` seguido do nome da suite criptográfica. Já o comando `ciphers -s` a seguir lista os algoritmos suportados na instalação local no OpenSSL.

```
C:\> openssl ciphers -s
```

Os comandos a seguir testam um servidor para uma conexão com AES128-SHA, uma configuração fraca.

```
C:\> openssl s_client -connect <<url>>:443 -cipher AES128-SHA
```

O comando a seguir usa a opção `-cipher kECDHE` para validar o uso do ECDH efêmero no acordo de chaves.

```
C:\> openssl s_client -connect <<url>>:443 -cipher kECDHE
```

Testar esses comandos para cada uma das URLs a seguir: www.amazon.com, www.yahoo.com, www.google.com, www.facebook.com.

- Quais aceitam a configuração fraca?
- Quais aceitam o ECDH efêmero?

Encriptação/decriptação AES no OpenSSL

Usar o OpenSSL disponível na VM do curso para executar os comandos desta atividade.

Usar o arquivo textoclaro.txt.

```
C:\> type textoclaro.txt
Este eh o texto claro.
Ele carrega informacao sensivel.
Por isto, deve ser protegido.
```

Criptografia baseada em senhas

Usa o OpenSSL para encriptar o arquivo com uma chave derivada de uma senha e AES-256-CTR, codificando o criptograma em base64 e mostrando no terminal os parâmetros usados na encriptação.

```
C:\> openssl enc -e -a -p -aes-256-ctr -pbkdf2 -in textoclaro.txt -out
criptograma.aes -k 1234
```

Visualizar o criptograma com o comando cat ou type

```
C:\> type criptograma.aes
U2FsdGVkX1/sRpbhNblSB2+tSItra/OJECL/uErcP1AocQ9yojnRctRLeNK05Swi97Nj4DLX
qaNjdhQPRhtj7jNb5gOrkbhLA3KDwwVtHE4p4q4J2LRAU9YqQM4gOBna1ClhgIM3lw==
```

Usar o OpenSSL para decriptar o criptograma anterior.

```
C:\> openssl enc -d -a -p -aes-256-ctr -pbkdf2 -in criptograma.aes -out
textoclaro3.txt -k 1234
```

Visualizar o texto claro com o comando cat ou type

```
C:\> type textoclaro3.txt
Este eh o texto claro.
Ele carrega informacao sensivel.
```

```
Por isto, deve ser protegido.
```

Criptografia com chaves aleatórias

Esta atividade usa o OpenSSL para encriptar o arquivo textoclaro.txt com AES-256-CTR, usando uma chave e um IV gerados aleatoriamente.

Gerar sequências hexadecimais pseudoaleatórias para serem usadas como IV de 16 bytes e chave criptográfica de 32 bytes.

```
C:\> openssl rand -hex 32
C:\> openssl rand -hex 16
```

Encriptar o texto claro com AES-256-CTR e chave e IV gerados.

```
C:\> openssl enc -e -a -p -aes-256-ctr -in textoclaro.txt -out
criptograma.aes -iv E570548E2B1376147E3342F08082E29A -K
6CB40CAEF6FDEDB31DBD908256F63276DB8FBC1E3430CEF18B6E0F5CD112D5F2
```

Visualizar o criptograma com o comando cat ou type.

```
C:\>type criptograma.aes
JnBQe28cnflvtIPGg/67SAIivfHMiLFxuVPpiQhPthiRZgbCjoHS53d7rqgBg/xlNgvvDkWw
var7CUVJ/39FjISl0TwEIZhUlvtq/4mb8HViYqlK9kTH
```

Usar o comando OpenSSL a seguir para decriptar o criptograma anterior. (Usar mesma chave e iv da encriptação)

```
C:\> openssl enc -d -a -p -aes-256-ctr -in criptograma.aes -out
textoclaro2.txt -iv E570548E2B1376147E3342F08082E29A -K
6CB40CAEF6FDEDB31DBD908256F63276DB8FBC1E3430CEF18B6E0F5CD112D5F2
```

Visualizar o texto claro com o comando cat ou type.

```
C:\>type textoclaro2.txt
Este eh o texto claro.
Ele carrega informacao sensivel.
Por isto, deve ser protegido.
```

Padrões do modo ECB

Quando o modo ECB é usado na encriptação, padrões de bits presentes no texto claro produzem padrões de bits no criptograma. Esta atividade usa o OpenSSL para gerar um arquivo de criptograma com 16 blocos iguais para que seja possível visualizar o padrão de repetição. O programa JCrypTool é usado para visualizar os padrões.

Na atividade, utilizar o arquivo *16blocostextoclaro.txt*. Encriptar o arquivo com AES-128-ECB.

```
C:\> openssl enc  -e  -aes-128-ecb  -in 16blocostextoclaro.txt -out
criptograma.ecb -K 00112233445566778899aabbccddeeff
```

Visualizar os padrões no criptograma com o JCrypTool.

- Abrir o arquivo `criptograma.ecb`
- Visualizar o arquivo em hexadecimal com Menu *Edit → Open with → Hexeditor*.

Padding na encriptação de bloco

Esta atividade usa o OpenSSL para decriptar criptogramas sem remover o padding e depois visualizar o texto claro com padding residual no JCrypTool.

15 bytes de padding

Visualizar o padding em um arquivo de 1 byte

1. Criar um arquivo com 1 byte apenas. Digitar apenas o 1 sem quebra de linha.
   ```
   C:\> notepad.exe 1byte.txt
   ```

2. Encriptar o arquivo 1byte.txt
   ```
   C:\> openssl enc  -e  -aes-128-ecb  -in 1byte.txt -out 1byte.ecb
        -K 00112233445566778899aabbccddeeff
   ```

3. Decriptar o arquivo sem remover o padding
   ```
   C:\> openssl enc  -d  -aes-128-ecb  -nopad  -in 1byte.ecb  -out
   1byte2.txt -K 00112233445566778899aabbccddeeff
   ```

4. Visualizar o conteúdo dos 15 bytes de padding

- No JCrypTool, abrir o arquivo `1byte2.txt`
- Visualizar o arquivo em hexadecimal com Menu *Edit → Open with → Hexeditor*.

1 byte de padding

Visualizar o padding em um arquivo de 15 bytes

1. Criar um arquivo com 15 bytes apenas. Digitar apenas `123456789abcdef` sem quebra de linha.
   ```
   C:\>  notepad.exe 15bytes.txt
   ```

2. Encriptar o arquivo 15bytes.txt
   ```
   C:\> openssl enc  -e  -aes-128-ecb  -in 15bytes.txt -out
        15bytes.ecb    -K 00112233445566778899aabbccddeeff
   ```

3. Decriptar o arquivo sem remover o padding
   ```
   C:\> openssl enc  -d  -aes-128-ecb  -nopad  -in 15bytes.ecb
        -out 15bytes2.txt -K 00112233445566778899aabbccddeeff
   ```

4. Visualizar conteúdo do byte de padding

- No JCrypTool, abrir o arquivo `15byte2.txt`
- Visualizar o arquivo em hexadecimal com menu *Edit → Open with → Hexeditor*.

1 bloco (16 bytes) de padding

Visualizar o padding em um arquivo de 16 bytes

1. Criar um arquivo com 16 bytes apenas. Digitar apenas `0123456789abcdef` sem quebra de linha.

```
C:\>  notepad.exe 16bytes.txt
```

2. Encriptar o arquivo 16byte.txt

```
C:\> openssl enc  -e  -aes-128-ecb  -in 16bytes.txt -out
     16bytes.ecb    -K 00112233445566778899aabbccddeeff
```

3. Decriptar o arquivo sem remover o padding

```
C:\> openssl enc  -d  -aes-128-ecb  -nopad  -in 15bytes.ecb
     -out 15bytes2.txt -K 00112233445566778899aabbccddeeff
```

4. Visualizar conteúdo do bloco de padding com o hexdump

- No JCrypTool, abrir o arquivo `16byte2.txt`
- Visualizar o arquivo em hexadecimal com Menu Edit → Open with → Hexeditor.

Teste SSL/TLS e HTTPS com Mozilla Observatory

A Mozilla oferece gratuitamente um serviço chamado HTTPS Observatory para teste de segurança SSL/TLS em servidores HTTPS. O conjunto de testes oferecido pelo site Mozilla Observatory é geralmente mais completo e atualizado que aqueles realizados por ferramentas individuais ou executados manualmente. Testar URLs com Mozilla Observatory.

- Acessar o site https://observatory.mozilla.org.
- Digitar no campo de url alguma rls de teste. Por exemplo, usar os endereços a seguir (um de cada vez): www.amazon.com, www.yahoo.com, www.google.com, www.facebook.com.
    - Analisar os resultados da varredura na guia TLS Observatory
        - Identificar as suites criptográficas do TLSv1.3, TLSv1.2, TLSv1.1 e TLSv1.0
        - Identificar os tamanhos das chaves dos certificados e o algoritmo de assinatura.
- Qual das URLs testadas têm o HTTPS mais forte?

Teste SSL/TLS e HTTPS com Qualys SSLLabs

A empresa de segurança cibernética Qualys oferece gratuitamente um serviço chamado SSL Labs para teste de segurança SSL/TLS em servidores HTTPS. Os testes oferecidos pelo Qualys SSL Labs são geralmente mais completos e atualizados que outros. Testar as URLs conhecidas com o SSLLabs.

- Acessar o site https://www.ssllabs.com/ssltest/.
- Digitar no campo de url alguma url de teste. Por exemplo, usar os endereços a seguir (um de cada vez): www.amazon.com, www.yahoo.com, www.google.com, www.facebook.com.
    - Analisar os resultados da varredura
        - Identificar as suites criptográficas do TLSv1.3, TLSv1.2, TLSv1.1 e TLSv1.0
        - Identificar os tamanhos das chaves dos certificados e o algoritmo de assinatura

- Quais testes de segurança contra ataques conhecidos são realizados?
- Como as URLs testadas se comportam nos testes de ataques conhecidos?