EA872 Laboratório de Programação de Software Básico
Atividade 12

Vinícius Esperança Mantovani **RA 247395**
Entrega (limite): 08/11/2023, em sala.

Exercício 1)

A ideia de base64 é codificar conjuntos de caracteres, que podem conter caracteres especiais que causem algum tipo de interpretação equivocada por máquinas que possam receber as informações passadas pela rede. Para tanto, o algoritmo funciona de modo a destrinchar os bytes de dados a serem enviados e organizá-los em blocos de 6, de maneira a garantir que os novos caracteres componentes da informação codificada sejam caracteres imprimíveis. Desta forma, o algoritmo garante que um conjunto de caracteres, por exemplo, um usuário e/ou uma senha, cheguem até a máquina de destino sem ser interceptado e interpretado por outra máquina da rede de maneira inadequada e, possivelmente, prejudicial.

Exercício 2)

A necessidade de se usar base64 dentro do escopo de autenticação do usuário se dá, pois, caso contrário, os dados de usuário e senha de um cliente podem ser corrompidos antes de chegarem à máquina do servidor por alguma outra máquina que possa interpretar algum dos caracteres como um caractere de controle e, usá-lo. Logo, base64 é preciso neste caso para evitar que caracteres especiais, que alguma máquina possa interpretar para controle, não sejam perdidos no caminho cliente-servidor.

Exercício 3)

A primeira limitação do crypt que é resolvida pelo uso não default é de número de caracteres considerados, que no modo default é de apenas 8, enquanto que, nos demais, é maior, conforme se segue:

Default:

```
Unset
( Salt = a7 ) + ( Password = abacatad ) ==> ( crypt = a7rGPumczgR9s )

( Salt = a7 ) + ( Password = abacatada ) ==> ( crypt = a7rGPumczgR9s )
```

Modo MD5:

```
Unset
( Salt = $1$a7$ ) + ( Password = abacatad ) ==> ( crypt =
$1$a7$KxGnl5XpGYTEmEmJ7yOrP/ )

( Salt = $1$a7$ ) + ( Password = abacatada ) ==> ( crypt =
$1$a7$y.8ilATu/3vtNx557S.Iv/ )
```

SHA-256:

```
Unset
( Salt = $5$a7$ ) + ( Password = abacatad ) ==> ( crypt =
$5$a7$RaoCo.27z3KrWSmwOK44.uVI7QmPov6gX6Tgi.FVrjA )

( Salt = $5$a7$ ) + ( Password = abacatada ) ==> ( crypt =
$5$a7$rFEISWd9vbqk2Mm50Ga/I2W4kcQbRNbMbxRRWBLwU70 )
```

SHA-512:

```
Unset
( Salt = $6$a7$ ) + ( Password = abacatad ) ==> ( crypt =
$6$a7$sQ3pXvtsU4txV63hSPKEXRtcF48Jfu.O1Jttswgh/MV3ct.Pxj.mk1argxlvorg2wk9q55
mJYgy8Pk20hMISw1 )

( Salt = $6$a7$ ) + ( Password = abacatada ) ==> ( crypt =
$6$a7$PKQ8fsSB3t8gTz7vg7.mVgkF04RNvqDSUeV5z6fvWyXL0m6ljVUowiLjr.3Qr0vghsPz0u
7amJ6atV0whsWSs/ )
```

Ademais, temos a limitação no tamanho do salt, que pode ser resolvida usando os modos não default para ter mais que dois caracteres de salt, conforme:

Default:

```
Unset
( Salt = a7 ) + ( Password = abacatad ) ==> ( crypt = a7rGPumczgR9s )

( Salt = a7b ) + ( Password = abacatad ) ==> ( crypt = a7rGPumczgR9s )
```

Modo MD5:

```
Unset
( Salt = $1$a7$ ) + ( Password = abacatad ) ==> ( crypt =
$1$a7$KxGnl5XpGYTEmEmJ7yOrP/ )
```

```
( Salt = $1$a7b$ ) + ( Password = abacatad ) ==> ( crypt =
$1$a7b$JFXpRwLMVyQsNDsrLIEBF/ )
```

SHA-256:

```
Unset
( Salt = $5$a7$ ) + ( Password = abacatad ) ==> ( crypt =
$5$a7$RaoCo.27z3KrWSmwOK44.uVI7QmPov6gX6Tgi.FVrjA )

( Salt = $5$a7b$ ) + ( Password = abacatad ) ==> ( crypt =
$5$a7b$6yGFI9AQFUKCIDU5UhLSLA.FZajtUZ1zAArjsMiHIIB )
```

SHA-512:

```
Unset
( Salt = $6$a7$ ) + ( Password = abacatad ) ==> ( crypt =
$6$a7$sQ3pXvtsU4txV63hSPKEXRtcF48Jfu.O1Jttswgh/MV3ct.Pxj.mk1argxlvorg2wk9q55
mJYgy8Pk20hMISw1 )

( Salt = $6$a7b$ ) + ( Password = abacatad ) ==> ( crypt =
$6$a7b$sLUujsAgxr.UbR7yvvGSe8t9qTQsPhOy8w8i1b0shF6rJ6zEp8nmMN6SqG3q8KzjyQLPP
rSLdpQG1OBVasJd30 )
```

Exercício 4)

Isso ocorre, pois caso as senhas sejam armazenadas em .htaccess, um possível invasor ou qualquer uma pessoa que tenha acesso ao sistema de arquivos do servidor poderia ver todas as senhas contidas neste arquivo .htaccess. Assim, por mais que as senhas estejam criptografadas, isso já seria um vazamento bastante trágico, que proveria a um possível invasor um conjunto muito útil de informações e a segurança do sistema seria comprometida. Logo, deve existir em .htaccess apenas o caminho para um arquivo que contenha as senhas e este deve estar armazenado em um diretório externo ao web-space.

Exercício 5)

O sistema basic authentication é considerado inseguro, por conta de não apresentar criptografia aos dados de usuário e senha enviados pela rede, apenas conter codificação base64. Deste modo, este sistema não esconde os dados de usuários, que, por este motivo, são enviados via rede sem qualquer tipo de proteção, podendo ser interceptados e obtidos por um invasor. Além disso, por não serem criptografados os dados, qualquer um que tenha acesso ao arquivo de senhas do servidor conheceria as senhas de todos os usuários.