Algorítmo HASH

Um algoritmo de *hashing* é um método de cifrar dados de forma a manter a sua privacidade. A função de *hash* a partir de uma cadeia de caracteres (*string*) de qualquer tamanho, cria uma *string* de tamanho fixo. Um Hash pode ser comparado com um selo de embalagem que indica clara e inequivocamente se a embalagem já foi aberta ou violada. O tamanho dessa string gerada é fixo idependente do tamanho do arquivo ou string analisado. Qualquer alteração efetuada no arquivo, por mínima que seja, altera substancialmente o resultado hash. Isto ocorre porque, mesmo se apenas um dos bits do arquivo for alterado, muitos bits do resultado serão afetados. Este comportamento é conhecido como **efeito avalanche**.

Hoje em dia é importante assegurar a integridade dos dados, esse conceito é muito usado hoje em dia em grandes e pequenas empresas. Tem diversas forma de fazer isso com diversos algoritmos. Porém vou mostrar como trabalhar com o algoritmo MD5.

MD5: O MD5 (Message-Digest algorithm 5) é um algoritmo de hash de 128 bits unidirecional desenvolvido pela RSA Data Security, Inc., descrito na RFC 1321, e muito utilizado por softwares com protocolo ponto-a-ponto (P2P, ou Peer-to-Peer, em inglês), verificação de integridade e logins.

Vulnerabilidade

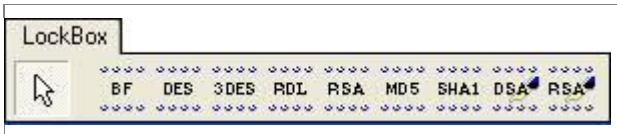
O MD5 só faz uma varredura sobre os dados, se dois prefixos com o mesmo hash podem ser construídos, um sufixo comum pode ser adicionado a ambos para fazer a colisão mais "real". Por isso, existe a possibilidade de duas strings ou arquivos produzirem um mesmo hash.

Aplicando no Delphi...

O Objetivo dessa matéria é justamente dar a orientação necessária para usarmos o código Hash em nossos projetos. Para isso existe um conjunto de componentes chamado “TurboPower LockBox”, que fornece algumas bibliotecas de encriptação RSA, MD5, SHA-1, DESDES etc. Vamos trabalhar com eles. Para fazer download acesse o site http://sourceforge.net/projects/tplockbox que é compátivel com o Delphi 03/04/07.

Instalando o Turbo Power

Estou trabalhando com o Delphi 7. Após extrair os arquivos adicione na sessão “Library Path” da IDE do Delphi o caminho da pasta SOURCE (ex: C:\lockbox\source) do componente. Abra o arquivo L207vd70.dpk que corresponde ao Delphi 7 Clique em



Compile e depois em *Install*. Pronto deverá aparecer uma aba com o nome LockBox - veja a figura 01.

Figura 01 – Componentes Turbo Power

O componente a ser destacado dessa palheta é o MD5. Com ele podemos gerar um código hash de 32 caracteres em uma *STRING* ou *ARQUIVO*. É bem fácil usar o componente. Vamos ver em duas situações que poderíamos usar o HASH:

1- Integridade de Arquivos

2- Armazenamento de Senha – Login

Integridade na transferência dos arquivos

Como vimos anteriormente cada arquivo tem o seu próprio código "Hash". Em uma tranferência de arquivos, downloads, para se assegurar que o arquivo não sofreu nenhum tipo de alteração na tranferencia é gerado um código hash do arquivo e enviado ao destinatário para assegurar que o arquivo é o mesmo.

Essa técnica é empregada pela ANS (Agência Nacional de Saúde) quando as famárcias enviam a relação dos medicamentos ao seu servidor. Para garantir que as informações dentro do arquivo não tenham sido alteradas.

Como aplicar esse algoritmo em um arquivo? É bem simples. Crie um novo projeto e adicione o componente LBMD5 da aba LockBox, um Button, um OpenDialog e um Label. No evento OnClick insira os seguintes códigos

```
uses  LbUtils; //Declarar a Unit LBUtils
var
   MD5Digest : TMD5Digest; //Variável "Digesto"
procedure TForm1.btnHashFileClick(Sender: TObject);
 begin
     if OpenDialog1.Execute then
     begin
       LbMD51.HashFile(OpenDialog1.FileName);
       LbMD51.GetDigest(MD5Digest);
       edtHash.Text := BufferToHex(MD5Digest, SizeOf(MD5Digest));
     end;
 end;
```

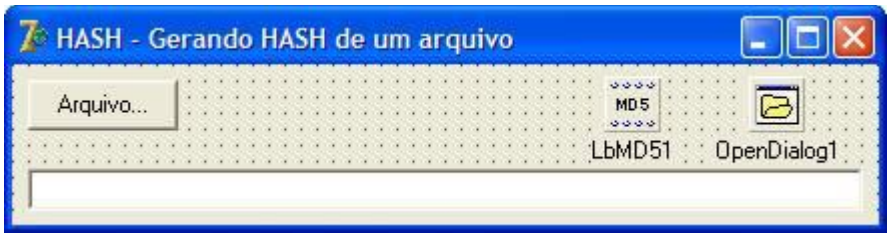


Figura 01 – Imagem do exemplo criado.

É criado uma variável do tipo TMD5Digest. Este tipo de variável armazenar o que chamamos de "Digesto". Digesto é resultado do algoritmo aplicado no arquivo, de tamanho fixo, numa operação de via única, ou seja, não é possível obter informações do arquivo selecionado a partir de um resultado hash.

O arquivo é selecionado pelo OpenDialog é gerado o hash (digesto). As informações dessa variável é convertida e atribuída ao Edit.

No exemplo selecionei o arquivo "Intro.txt" localizado na pasta "C:\WINDOWS\Help\Tours\mmTour". Retornou o seguinte código HASH:

A4AFEAB6EEA54A5D08184D9130B57451

Quando enviamos para alguém um arquivo por email, podemos enviar também o código HASH junto. Justamente para o destinatário verificar a integridade do arquivo, se ele sofreu algum tipo de alteração na transferência/envio. Isso é muito usado por diversos servidores na internet que disponibilizam grande quantidade de arquivos para download.

Um exemplo é o site http://www.codeblocks.org/downloads.shtml aonde disponibiliza diversos resources para desenvolvedores. Observe que abaixo de cada download (link) tem uma informação "MD5 Hash", que é o código hash dos arquivos disponíveis (figura 02).

Figura 02 – Exemplo de HASH no site http://www.codeblocks.org/downloads.shtml

2- Armazenamento de Senha – Login

Hoje em dia a segurança é muito destacada quando desenvolvemos aplicações

com Banco de Dados. Normalmente em um banco de dados é gravado a senha de cada usuário de sistema. É muito mais seguro armazenar os resultados hash das senhas. Nesse caso é o usuário digitar a sua senha, é gerado o hash e comparado com o hash que está no Banco de Dados. Se forem idênticos, a senha confere. Usando este recurso reduz muito os riscos de descobrir a senha do usuário, justamente porque só será possível ser roubada após o usuário estiver digitado antes de transformá-la em hash.

Vamos ver como gerar um hash a partir de uma string.

Adicione um Button e dois Edits (Figura 03) e digite o seguinte código no evento OnClick desse botão:

```
Uses LbUtils, LbCipher; // declarar na cláusula USES no começo da unit
...
var
                    Form1:                    TForm1;
          MD5Digest          :          TMD5Digest;
Implementation
{$R                                                  *.dfm}
procedure          TForm1.HashGerClick(Sender:          TObject);
begin
                StringHashMD5(MD5Digest,                eMens.Text);
     eHash.Text     :=     BufferToHex(MD5Digest,     SizeOf(MD5Digest));
end;
```

O Código é bem simples. É gerado o "Digesto" (Hash) usando a função StringHashMD5 e depois é convertido o valor para ser apresentado no componente Edit.

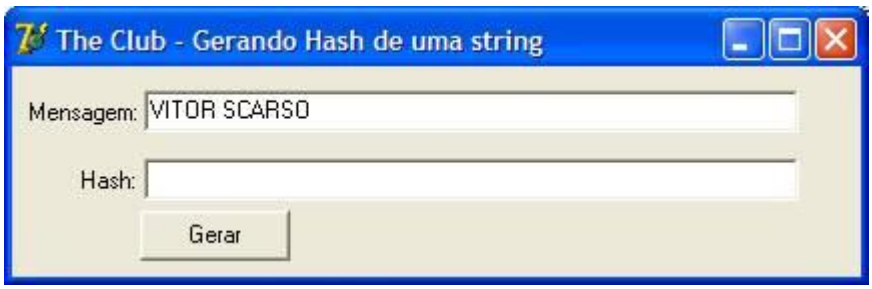


Figura 03 – Imagens de como aplicar o Hash em uma String

Exemplo para Download:

Hash no Arquivo: (link)

MD5: D0CE0011A81BC8A4124BB3A692CE5B73

Hash em uma String: (link)

Voltar MD5: DD8912AD0743288C1AB6A1B65FE9C0D0

Conclusão

Essa matéria abordou o algoritmo Hash MD5. Mesmo sendo muito usado não é um algoritmo 100% seguro. Mas poderá ajudar muito na segurança. Estou aberto a novas idéias com respeito à criptografia e integridade dos arquivos. Um abraço a todos e até a próxima!

Av. Prof. Celso Ferreira da Silva, 190 - Avaré - SP - CEP 18707-150 - Tel: (014) 3732-1529
Contato via Skype: theclub_cadastro - e-mail: cadastro@theclub.com.br