Criptografia em PHP + Apache

Bruno Nunes

1. INTRODUÇÃO 3
2. MOTIVAÇÃO PARA O TRABALHO 4
3. ONE WAY CYPHER 5
4. MCRYPT 5
5. OPENSSL - MOD_SSL 7
6. WEBDAV – MOD_DAV 8
FUNCIONAMENTO 9
DELTAV 11
7. GNUPG 11
FUNCIONAMENTO 12
UTILIZAÇÃO 13
CUIDADOS 15
8. CONCLUSÃO 15
9. BIBLIOGRAFIA 17

1. Introdução

O grande objectivo deste trabalho é apresentar uma série de opções para a criação de um ambiente de desenvolvimento *web* seguro, com base em tecnologias e plataformas gratuitas, nomeadamente o servidor *web* Apache e a linguagem PHP.

A linguagem PHP é uma linguagem de programação reflectiva, originalmente desenvolvida para a produção de páginas *web* dinâmicas. É normalmente utilizada como uma linguagem de scripting ao nível do servidor (*server-side*), mas pode também ser utilizada em linha de comandos ou em aplicações gráficas *stand-alone*. É também possível o desenvolvimento de interfaces textuais recorrendo à biblioteca *ncurses*.

A PHP pode ser instalada na maioria dos servidores *web* e em praticamente todos os sistemas operativos e plataformas, de forma completamente gratuita. O *PHP Group* disponibiliza também o código fonte completo, permitindo que os utilizadores personalizem, aumentem e adequem a linguagem para que responda às suas necessidades individuais. Ao correr o *parser* PHP num servidor *web*, este pode ser comparado a outras linguagens *server-side* como a ASP.NET da Microsoft, a JavaServer Pages da Sun Microsystems, mod_perl ou a *framework* Ruby on Rails, uma vez que todas estas permitem a disponibilização de conteúdo dinâmico a um cliente, a partir do servidor *web*. Para melhor conseguir competir com a abordagem "*framework*", a Zend disponibiliza *Zend Framework,* um conjunto de funcionalidades e *best practices*; outras frameworks semelhantes incluem a CakePHP, PRADO e Symfony

A arquitectura LAMP tem vindo a ganhar notoriedade na indústria *web,* como um meio de implementar aplicações *web* baratas, fiáveis, escaláveis e seguras. A PHP corresponde à letra P na suite LAMP, em associação com o Linux, Apache e MySql.

Esta flexibilidade faz com que a PHP tenha uma grande quota de utilização na Internet, com cerca de 19 milhões de domínios implementados com base nesta linguagem. Como exemplos de aplicações PHP populares temos o phphBB, WordPress e MediaWiki.

O servidor Apache é provavelmente o servidor *web* de maior sucesso na internet. Cria em 1995 por Rob McCool (então funcionário do *National Center for Supercomputing Applications*, NCSA). Numa pesquisa realizada em Dezembro de 2005, chegou-se à conclusão que a utilização do Apache supera os 60% de servidores *web* activos.

È a principal tecnologia da Apache Software Foundation, responsável por mais de uma dezena de projectos envolvendo tecnologias de transmissão por via *web*, processamento de dados e execução de aplicações distribuídas.

O servidor é compatível com o protocolo HTTP versão 1.1. As funcionalidade disponibilizadas são geridas através de uma estrutura de módulos, podendo o utilizador inclusivamente escrever os seus próprios módulos, através da API do software.

Existem versões para os sistemas Windows, Novell Netware, OS/2 e vários outros do padrão POSIX (Unix, Linux, FreeBSD, entre outros).

2. Motivação para o trabalho

A proposta de um trabalho sobre criptografia surgiu numa altura em que um familiar meu tinha demonstrado interesse na criação de um website que pudesse servir de suporte à sua actividade profissional. O seu desejo era criar um website que lhe permitisse estar em constante contacto com os seus clientes, podendo partilhar documentos com estes quando tal se revela-se necessário.

Devido a uma curiosidade pessoal em relação à linguagem PHP resolvi que esta seria a opção escolhida. Assim tornou-se evidente que uma solução como a pretendida iria requerer mecanismos que garantissem a segurança das comunicações, bem como dos documentos que estariam disponíveis, e autenticação das várias partes envolvidas.

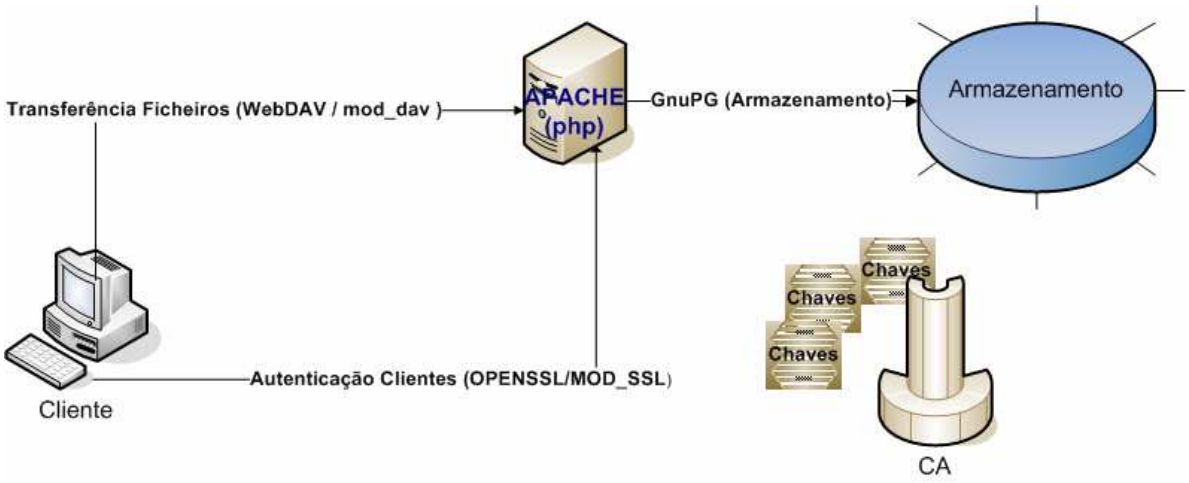

Ilustração 1 - Arquitectura pretendida

Após uma pesquisa inicial, considerou-se que a arquitectura pretendida seria algo como o apresentado na imagem anterior. Seria necessário permitir estabelecer comunicações seguras com o site (através do OpenSSL/mod_ssl, por exemplo), garantir um armazenamento seguros dos documentos (através, por exemplo, de GnuPG) e garantir que a transferência de dados em si seria também ela segura (através, neste caso, do WebDAV/mod_dav).

Procurou-se, dentro dos possíveis, criar toda a infrastrutura recorrendo a tecnologias/ferramentas open-source e/ou gratuitas (daí a opção pelo servidor web Apache).

Este trabalho terá por objectivo procurar apresentar uma pequena explicação acerca de cada uma das opções indicadas na ilustração.

3. One Way Cypher

A maneira mais simples de acedermos a uma funcionalidade de criptografia na linguagem PHP é através da função Crypt("texto a encriptar"). Esta função é tão simples que não tem sequer uma função de desencriptação correspondente! A pergunta lógica que suge após esta explicação é "Qual é a utilidade de uma funcionalidade que encripta, mas não disponibiliza qualquer meio para realizar a desencriptação?" A resposta é que esta função pode ser de grande utilidade para a implementação de sistemas de autenticação, uma vez que o texto a encriptar surtirá sempre o mesmo criptograma (como iremos ver mais abaixo, dependendo de um segundo parãmetro que poderemos definir) de cada vez que for utilizada a função *Crypt*, este pode ser utilizado para verificar *passwords*, sem que estas sejam explícitamente referênciadas no processo de autenticação. A *password* associada a um utilizador (definida como um registo numa tabela da base de dados), pode ela própria ser armazenada de forma encriptada.

Tal como foi referido anteriormente, a função não devolve exactamente sempre o mesmo resultado para um mesmo texto limpo indicado como parãmetro. Se apenas indicarmos o parãmetro "texto limpo", a função gerará um valor aleatório que fará com que o resultado de uma encriptação seja diferente de cada vez que é chamada, para um mesmo texto limpo. A verdade é que a função *Crypt* aceita dois parãmetros: o texto a encriptar, e um *salt*. É este *salt* que irá permitir que personalizemos e alteremos o processo de encriptação realizado pela função. Com esta personalização será possivel condicionarmos a encriptação de forma a que esta devolva sempre o mesmo valor para um mesmo "texto limpo", e assim conseguir fazer a autenticação de um utilizador, por exemplo (o *salt* deverá ser armazenado de forma a que o mesmo valor possa ser utilizado para fazer a validação da autenticação).

Por defeito, o algoritmo de encriptação utilizado pela função Crypt é o MD5, mas podem também ser utilizados o DES, Extended DES e BLOWFISH. Para verificar que algoritmos uma dada versão da função *Crypt* suporta, basta analisar o valor das constantes CRYPT_MD5, CRYPT_STD_DES, CRYPT_EXT_DES e CRYPT_BLOWFISH, respectivamente (se alguma delas devolver o valor 0, então a versão da função não suportará esse algoritmo). A indicação de qual o algoritmo que se pretende utilizar, é dada pelo tamanho e composição do parãmetro *salt* . Assim, para encriptar através do algoritmo MD5, o *salt* deverá apresentar um tamanho de 12 caracteres, começando com "1"; se pretender-mos utilizar o DES, então o *salt* deverá apresentar 2 caracteres; para o Extended DES será necessário um *salt* com 9 caracteres ; finalmente se pretender-mos utilizar o BLOWFISH, então o *salt* deverá apresentar 16 caracteres, começando com "2". O tamanho padrão do *salt* utilizado pode ser consultado através da constante CRYPT_SALT_LENGTH.

4. MCrypt

Uma outra hipótese para se ter acesso a funcionalidades criptográficas é a utilização da biblioteca MCrypt. Esta não vem incluída à partida no PHP, sendo necessária a sua instalação (disponível a partir de http://mcrypt.sourceforge.net) e configuração.

O MCrypt disponibiliza uma variedade de algoritmos como, Arcfour, Blowfish, Cast, DES, Triple DES, Enigma, Gost, Idea, RC2, RC6, Loki, Mars, Panama, Rijndael, Safer, Safer+, Serpent, Skipjack, Twofish, Wake ou Xtea. Estes podem depois ser utilizados sob uma série de modos, nomeadamente, CBC, CFB, CTR, ECB, OFB e NCFB. É importante saliente que nem todos os modos estão disponíveis para todos os algoritmos.

A biblioteca MCrypt é particularmente útil, não só pelo número de algoritmos criptográficos que disponibiliza, mas também por permitir a sua utilização para a cifra e decifragem de dados. Para além disto, a extensão MCrypt oferece 35 funções úteis à manipulação de dados.

Depois de instalada e configurada, a MCrypt pode ser utilizada da seguinte forma:

```
<?php

// Indicar uma texto limpo a ser cifrado
$string = "É fácil cifra texto com a biblioteca MCrypt para
PHP";

// Chave para a cifragem/decifragem
$key = "Uma chave";

// Indicar que algoritmo se pretende utilizar
$cipher_alg = MCRYPT_RIJNDAEL_128;

// Indicar um sector de inicialização para aumentar a
// segurança.
$iv = mcrypt_create_iv(mcrypt_get_iv_size($cipher_alg,
MCRYPT_MODE_ECB), MCRYPT_RAND);

// Imprimir texto limpo
print "Texto limpo: $string <p>";

// Encriptar o texto limpo ($string)
$encrypted_string = mcrypt_encrypt($cipher_alg, $key,
$string, MCRYPT_MODE_CBC, $iv);

// Converter para hexadecimal e apresentar no browser
print "Texto cifrado: ".bin2hex($encrypted_string)."<p>";

$decrypted_string = mcrypt_decrypt($cipher_alg, $key,
$encrypted_string, MCRYPT_MODE_CBC, $iv);

print "Texto decifrado: $decrypted_string";

?>
```

Provavelmente, as duas funções que chamam mais a atenção no exemplo apresentado são a mcrypt_encrypt() e a mcrypt_decrypt(), sendo evidentes os respectivos objectivos. Neste caso é utilizado o modo ECB ("*Electronic Codebook Mode*"), no entanto a MCrypt suporta também CBC ("*Cipher Block Chaining*"), CFB ("*Cipher Feedback*") e OFB ("*Output Feedback*"). O modo ECB é adequado à cifra de pequenas quantidades de dados aleatórios, como números de cartões de crédito, ou outras chaves utilizadas ppra cifragem. O CBC é útil quando se pretende cifrar grandes quantidades de dados como ficheiros, apresentando uma segurança bastante superior à apresentada pelo modo ECB. Se por outro lado, se pretender cifrar quantidades muito pequenas de dados, como por exemplo bytes indivíduais (*streams*). Fiialmente, está também disponível o modo OFB, que é análogo ao CFB, mas que pode ser utilizado em aplicações em que a propagação de erros não pode ser tolerada.

No exemplo foi também utilizado um vector de inicialização, que é um meio de aumentar a segurança da cifra através da introdução de um grau de aleatoriedade. Este deve ser único, mas igual tanto na cifragem como decifragem

5. OpenSSL - mod_ssl

O *Secure Sockets Layer* (SSL) é um protocolo que pode ser colocado como uma camada entre um protocolo fiável orienteado a conecções de rede (como o TCP/IP) e um protocolo aplicacional (como HTTP). O SSL permite uma comunicação segura entre o cliente e o servidor, ao diponibilizar autenticação mútua, utilização de assinaturas digitais (para garantir integridade), e encriptação (garantindo privacidade).

O protocolo foi desenhado, procurando suportar um grande espectro de opções para algoritmos de criptografia, *digests*, e assinaturas. Esta filosofia de desenvolvimento permite que a selecção de algoritmos para um servidor específico possa ser feita com base em considerações legais ou outras, permitindo também que o protocolo possa tirar partido de novos algoritmos. As escolhas são negociadas pelo cliente e servidor no início do estabelecimento da sessão.

Um utilização comum do SSL, é a garantia da segurança de comunicação web HTTP, entre um browser e um servidor web. Esta opção não excluí a possibilidade de HTTP não-seguro. A versão segura é principalmente HTTP sobre SSL (denominado HTTPS), com uma grande diferença: utiliza um URL http**s** (por oposição ao http) e uma porta distinta (normalmente a 443). É esta a funcionalidade básica que o mod_ssl apresenta para um servidor Apache. O pacote mod_ssl consiste num módulo SSL (identificado como 1 na ilustração abaixo) e um conjunto de patchs para o Apache, adicionando a *Extended* API

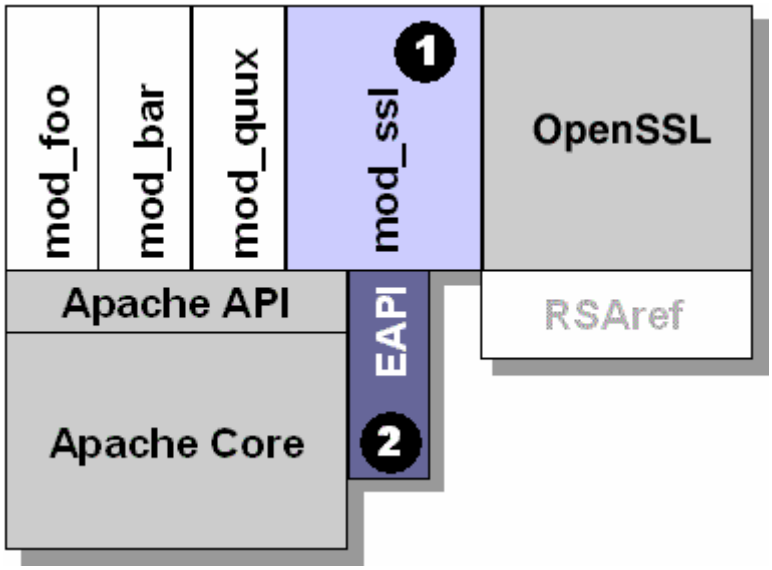

Ilustração 2 - Arquitectura do Módulo mod_ssl [2]

(EAPI) (o 2 na ilustração), sendo estas um pré-requisito essencial para a utilização do mod_ssl. Só é possível utilizar o mod_ssl se o código base do Apache possuir a *Extended* API. No entanto, como quando se aplica o mod_ssl à àrvore de código do Apache, a EAPI é automaticamente adicionada, este factor não necessita nomalmente o envolvimento do utilizador.

6. WebDAV – mod_dav

O *Web-based Distributed Authoring and Versioning* (WebDAV, RFC 2518) define uma série de extensões ao HTTP, que permitem a edição distribuída de documentos através da *web*. O WebDAV disponibiliza um protocolo de rede para a criação de aplicações colaborativas e interoperáveis, transformando a *web* num meio editável e colaborativo. O objectivo básico doWebDAV é permitir o *Input/Outuput* de ficheiros através de HTTP, permitindo assim o carregamento, edição, eliminação e criação de ficheiros e directórios, no fundo possibilitando a gestão remota de ficheiros localizados em servidores *Web*, através de aplicações-cliente específicas.

O completo suporte de funcionalidades de controlo de versões encontra-se ainda em desenvolvimento, no entanto a sua especificação (o DeltaV) contempla já os métodos VERSION-CONTROL, CHECKOUT, CHECKIN e UNCHECKOUT.

O suporte ao WebDAV é cada vez maior. Ferramentas como o Internet Explorer, Frontpage, Dreamweaver, Photoshope GoLive possibilitam desde já a interacção com servidores que disponibilizem funcionalidades WebDAV. Vários sistemas operativos também já possibilitam a sua utilização, nomeadamente o Windows (através dos denominados *Web Folders*), Mac OS X (*mount*) e Gnome (Gnome VFS e Nautilus).

O protocolo WebDAV podem ser sumariamente resumido com base em três grandes conceitos: *Locking* (controlo de concorrência), Propriedades, e Manipulação de *Namespaces*. O controlo de concorrência caracteriza-se pelo estabelecimento de *locks*

para escrita de grande duração, que previnem problemas de sobre-escrita (*overwriting*), nos quais dois ou mais colaboradores escrevem sobre um mesmo recurso, sem terem inicialmente realizado uma actualização do recurso com base na sua versão local (*merge*). Para conseguir obter capacidades de colaboração robusta e escalável à dimensão da Internet, onde as ligações à rede podem ser quebradas arbitrariamente, e em que cada ligação aberta consome recursos do servidor, a duração dos *locks* do WebDAV é independente de qualquer ligação de rede individual.

São utilizadas propriedades XML que permitem o armazenamento de metadados arbitrários, tais como listas de autores de determinados recursos *web*. Esta propriedades podem ser criadas, eliminadas e obtidas através do protocolo WebDAV. O protocolo *DAV Searching and Locating* (DASL) possibilita a pesquisa de recursos *web*, com base em valores de propriedades.

O terceiro e último conceito é o da Manipulação de *Namespaces*. Como pode haver a necessidade de copiar ou mover recursos à medida que um *website* evolui, o protocolo WebDAV suporta estas duas operações. Não existe a noção de directórios em WebDAV, sendo este conceito substituído por *Collections*, que podem ser criadas e listadas.

Funcionamento

Cada utilizador (com permissões de escrita e leitura) pode montar no seu computador pessoal uma unidades WebDAV localizada num servidor partilhado. A partir do momento em que o servidor suporta WebDAV, os utilizadores podem montar unidades independentemente do sistema operativo instalado no servidor. Os utilizadores podem então aceder aos ficheiros como se de uma outra qualquer unidade de rede se tratasse

Actualmente, quando um grupo de utilizadores necessita de colaborar na criação de um documento, este é normalmente reenviado entre os participantes através de email. Esta situação pode tornar-se lenta, confusa e propícia a erros, à medida que a tarefa se prolonga, tornando difícil o acompanhamento de versões do documento. O WebDAV permite que equipas cujos elementos se encontram geograficamente distribuídos possam colaborar no desenvolvimento, edição e gestão de conteúdos, independentemente do local em que cada elemento da equipa se encontre.

As operações do protocolo HTTP denominam-se métodos. O WebDAV acrescenta sete novos métodos (GET, HEAD, POST, OPTIONS, PUT, DELETE e TRACE) àqueles definidos pelo HTTP/1.1. Estes métodos permitem a protecção de sobreescrita (overwriting), gestão de metadados, e gestão de *namespaces*. Tal como um utilizador de um *browser* ignora todo o tráfego HTTP que acontece quando acede a um *wesite*, um utilizador de uma aplicação que suporte o WebDAV também não terá consciência da sua utilização. O protocolo WebDAV foi desenvolvido para ser integrado com as ferramentas de edição existentes, adicionando suporte para edição remota de recursos através da *web*, a aplicações com as quais os utilizadores já estão habituados a trabalhar.

A disponibilização do WebDAV num servidor Apache é um processo que comporta duas grandes fases:

- Compilar ou instalar o módulo mod_dav de forma a adicionar os serviços WebDAV à instalação do Apache.

- Alterar a configuração do Apache, indicando em que directórios os serviços WebDAV devem ser implementados.

Para utilizar o WebDAV é necessário indicar a localização do ficheiro de *lock* (através da directiva *DAVLockDB*), que será utilizado para registar que utilizador tem determinado ficheiro aberto para edição ou actualização. O ficheiro de lock impede também que vários utilizadores estejam a alterar um mesmo ficheiro ao mesmo tempo. Pode também ser definido um valor para *timeout* para a duração dos *locks* sobre um recurso. Este *lock* será liberto se durante o período indicado pelo *timeout*, não for realizado qualquer acesso ao recurso. O utilizador/cliente define um *timeout* para a sua ligação sempre que esta é iniciada. Este valor pode no entanto ser sobreposto através da directiva *DAVMinTimeout*. Estas duas directivas podem ser indicadas no ficheiro "httpd.conf " do servidor Apache, através da inclusão das seguintes linhas:

```
DAVLockDB /tmp/DAVLock
DAVMinTimeout 600
```

A primeira indica o ficheiro de *locks*, e a segunda a duração do *timeout* (em segundos). É importante salientar que estas definições são globais à instância do servidor Apache. Podem no entanto ser também definidos valores específicos para o *timeout* de directórios individuais, através da definição de valores específicos para a "*DAVMinTimeout*", na especificação desse directório.

Para definir individualmente que um dado directório suporta serviços WebDAV basta utilizar a directiva "Location":

```
<Location /dav/>
    DAV On
    AuthType Basic
    AuthName "WebDAV Restricted"
    AuthUserFile /export/http/webs/pri.mcslp/dav/.DAVlogin
    <LimitExcept GET HEAD OPTIONS>
        Require user webdav
    </LimitExcept>
</Location>
```

Esta directiva indica ao Apache que os serviços WebDAV devem ser activados para esta localização (directório). Esta directiva pode ser também utilizada para activar autenticação para um dado directório, garantindo assim a segurança da informação armazenada, e permitindo acesso apenas a entidades autorizadas. No exemplo apresentado, apenas o utilizador "webdav" tem pleno accesso ao directório, podendo os restante utilizadores apenas ler.
Finalmente resta apenas criar o ficheiro de passwords (se este ainda não existir), e criar o directório onde estamos a activar o WebDAV. Depois de definidos todos estes pontos será necessário reiniciar o servidor apache para que as opções entrem em efeito.

DeltaV

O protocolo DeltaV anteriormente referido, adiciona onze métodos àqueles disponibilizados pelo HTTP e WebDAV. A capacidade de controlo de versões é disponibilizada atravéz dos métodos VERSION-CONTROL, CHECKIN, CHECKOUT, UNCHECKOUT e REPORT. Um recurso sem controlo de versões pode passar a sê-lo através do método VERSION-CONTROL. A partir do momento em que um recurso se encontre sobre controlo de versões, um processo de edição é normalmente iniciado por um CHECKOUT, seguido por uma série de escritas (PUT's) sobre o recurso, sendo finalmente terminada com um CHECKIN. Uma sessão de edição pode ser cancelada através de um UNCHECKOUT. O histórico de versões de um recurso pode ser obtido através do método REPORT. Podem ser também atribuídas descrições interpretáveis pelos seres humanos através do LABEL. A revisão padrão actual pode ser indicada através de um UPDATE. Dois ramos distintos de versões de um recurso podem ser integrados através do MERGE.Uma actividade representa alterações lógicas que se estendem ao longo de uma série de revisões; MKACTIVIVITY cria novas actividades. O conceito de *workspace* permite que vários colaboradores possam trabalhar de forma isolada num conjunto de recursos; O MKWORKSPACE cria novos *workspaces*. Se se pretender obter uma espécie de imagem das versões actuais dos recursos pode ser utilizado o conceito de *baseline*. Este conceito é útil para armazenar, por exemplo, o estado de um sistema de software antes de ser disponibilizada uma nova versão para o público.

Embora o WebDAV não seja um protocolo directamente relacionado com criptografia, a verdade é que é extremamente útil num sistema como o proposto neste trabalho. Mais, com o auxílio de uma das outras possibilidades de encriptação em ambiente Apache+PHP seria possível criar um conjunto de recursos, acompanhando o respectivo histórico de versões, com garantias que as alterações teriam sido realizadas apenas por elementos autorizados, e que as alterações teriam sido de facto realizadas por determinado elemento (através da associação de assinaturas digitais às várias versões).

7. GnuPG

O *GNU Privacy Guard* (GnuPG ou GPG) é um software livre que pode ser utilizado em substituição da suite criptográfica PGP (*Pretty Good Privacy*). Faz parte do projecto de software livre da GNU da *Free Software Foundation*, tendo recebido grande apoio por parte do governo Alemão. O GnuPG respeita o RFC 2440 e o padrão para OpendPGP da IETF. As versões actuais do PGP podem interagir com o GnuPG e outros sistemas que respeitem o OpenPGP.

O GnuPG é um software estável e de nível comercial. É frequentemente incluído em sistemas operativos livres como FreeBSD, OpenBSD, e NetBSD, bem como na grande maioria das distribuições de GNU/Linux.

O programa GnuPG básico utiliza um interface de linha de comandos, mas existe uma série de *front-ends* que permitem a utilização de um ambiente gráfico. Como exemplo, foi integrado suporte de encriptação no KMail e Evolution (os clientes de email gráficos utilizados na maioria das distribuições Linux pessoais) através da utilização do GnuPG. Estes oferecem também front-ends gráficos para o GnuPG (Seahorse no GNOME, e KGPG no KDE). No Mac OS X, o projecto Mac GPG disponibiliza um número de *front-ends* Aqua para a integração de encriptação e gestão de chaves no sistema operativo, bem como instalações de GnuPG através de pacotes de instalação. Aplicações de *instant-messaging* como a Psi e a Fire podem encriptar as mensagens automáticamente quando o GnuPG está instalado e configurado. Até mesmo software *web-based* (como o Horde) pode fazer uso do GnuPG. Estão também disponíveis *plugins* para vários progrmas, como por exemplo o *plugin* Enigmail que permite integrar o suporte GnuPG no Mozila Thunderbird e SeaMonkey. Já o Enigform permite disponibilizar o GnuPG no Mozilla Firefox.

Em 2005 a G10 Code e a Intevation lançaram o Gpg4win, uma suite de software que incluía o GnuPG para Windows, WinPT, *Gnu Privacy Assistant*, e *plugins* para o *Windows Explorer* e *Outlook*. Todas estas aplicações estavam reunidas num instalador Windows, tornando trivial a sua instalação.

Funcionamento

O GnuPG encripta as mensagens utilizando um par assimétrico de chaves, geradas por cada utilizador GnuPG individual. Estas chaves podem depois ser trocadas com outros utilizadores de diversas formas, como por exemplo através de servidores de chaves acessíveis através da *web*. Esta troca de chaves deve ser sempre realizada de forma cuidada, para impedir a corrupção da relação entre a chave e o seu dono legítimo (permitindo assim que um terceiro pudesse assumir a identidade do dono da chave). É também possível adicionar a uma mensagem uma assinatura digital encriptada, de forma a assegurar a integridade e remetente da mensagem.

O GnuPG não utiliza software ou algoritmos patenteados ou restrictos, como o algoritmo de encriptação IDEA, que estava presente no PGP practicamente desde o seu início. Assim, o GnuPG utiliza antes uma variedade de outros algoritmos não patentiados como o CAST5, Triple DES, AES, Blowfish e Twofish. Continua a ser possível utilizar o IDEA na GnuPG através de um plugin. A sua utilização requer no entanto a obtenção de uma licença em alguns países em que o algoritmo está patenteado.

O GnuPG é um software de encriptação híbrido, no sentido que usa uma combinação de criptografia simétrica convencional para velocidade, e de criptografia de chave pública para uma troca segura de chaves, normalmente utilizando a chave pública do destinatário para encriptar uma chave de sessão que é utilizada uma única vez. Este modo de operação é parte do padrão OpenPGP, e tem sido utilizado no PGP desde a sua versão inicial.

Utilização

Como vimos, o GnuPG é uma aplicação apenas de linha de comandos. Existem vários interfaces gráficos, mas a sua utilização pressupõe a existência da aplicação de linha de comandos em si.

Tal como na maioria das aplicações de linha de comandos, deve ser chamado o comando "gpg" seguido de uma série de *switches* que afectam o resultado do comando. Por exemplo, para encriptar um ficheiro chamado "Dados_Secretos.txt" seria chamado o comando "GPG" seguido do parâmetro "-e" (indicando que se pretendia encriptar) e do parâmetro "-r NOME" (para indicar que utilizador deveria poder desencriptar o ficheiro). A variável "NOME" neste caso deve ser o primeiro nome ou endereço de email da pessoa que irá receber o ficheiro (é de notar que o utilizador indicado em "NOME" deve estar incluído no porta-chaves público do utilizador que está a encriptar o ficheiro).

```
GPG –e –r ze@ninguem.com Dados_Secretos.txt
```

Este comando terá como resultado a criação de um ficheiro encriptado chamado "Dados_Secretos.txt.gpg". Para desencriptar o ficheiro, o destinatário teria apenas de chamar o seguinte comando:

```
GPG –d Dados_Secretos.txt.gpg
```

Foi apenas necessário introduzir o *switch* "-d" e indicar o ficheiro encriptado. Como cada utilizador tem a sua chave privada no seu próprio porta chaves, o GnuPG consegue determinar para quem é que o ficheiro encriptado se destinava, e desencriptá-lo-á desde que a palavra-passe correcta seja introduzida.

A utilização do GnuPG no PHP não é mais do que a chamada destes comandos a partir de um script PHP. Assim para encriptar um ficheiro num *script* PHP através do GnuPG teríamos de criar algo deste género:

```
<?php
   $gpg = '/usr/bin/gpg';
   $destinatario = ze@ninguem.com';
   $ficheiro_secreto = 'Dados_Secretos.txt';

   echo shell_exec("$gpg -e -r $destinatario
   $ficheiro_secreto");
?>
```

Depois de correr este *script*, teria sido criado um ficheiro com o nome "Dados_Secretos.txt.gpg".

È também possível encriptar dados que não estejam organizados sob a forma de um ficheiro. Para este propósito poderia ser utilizado um script deste tipo:

```
<?php
   $gpg = '/usr/bin/gpg';
   $destinatario = 'ze@ninguem.com';
```

```
    $ficheiro_encriptado = 'foo.gpg';

    shell_exec("echo $argv[1] | $gpg -e -r $destinatario
    -o $ficheiro_encriptado");
?>
```

Este script pega no valor de $arg[1] (o primeiro argumento após o nome do script) e passa-o ao GnuPG para este ser encriptado. O GnuPG, através do *switch* –o, escreve os dados encriptados para o ficheiro com o nome indicado pela variável $ficheiro_encriptado.
Outra possibilidade ao nosso alcance é a de armazenar os dados encriptados sob a forma de uma variável. Assim poderemos utilizar o próprio PHP para tratar os dados encriptados, poupando I/O. Como exemplo é apresentado o seguinte script:

```
<?php
     $gpg = '/usr/bin/gpg';
     $destinatario = 'ze@ninguem.com';
     $mensagem_encriptada =
     base64_encode(shell_exec("echo $argv[1] | $gpg -e
     -r $destinatario"));

      mail(ze@ninguem.com',
                  'Aqui está a mensagem encriptada',
                 $mensagem_encriptada);
?>
```

É de notar a importância de codificar os dados em Base-64 para que seja fácil a sua interpretação pelo cliente de *email*.

A desencriptação com o GnuPG através do PHP é um pouco mais complexa que a encriptação, já que é necessário indicar uma palavra-passe ao GnuPG. A solução para não ser necessária a introdução da palavra-passe sempre que se pretende desencriptar passa pela utilização de um *switch* chamado `--passphrase-fd`. Este *switch* indica ao GnuPG que este deve aceitar a palavra-passe apresentada num file descriptor, sendo assim possível fazer um `echo` de um ficheiro ou `string` que será depois aceite pelo GnuPG como palavra-passe. Esta técnica encontra-se exemplificada em seguida:

```
<?php
     $gpg = '/usr/bin/gpg';
     $palavra_passe = 'A minha palavra-passe secreta';
     $ficheiro_encriptado = 'foo.gpg';
     $ficheiro_desencriptado = 'foo.txt';

     echo shell_exec("echo $palavra_passe | $gpg --
     passphrase-fd 0 -o $ficheiro_desencriptado -d
     $ficheiro_encriptado");
?>
```

Este *script* indica ao GPG que deve aceitar a palavra-passe definida no STDIN (indicado pelo 0 a seguir ao *switch*) e desencriptar a informação, guardando-a num ficheiro denominado “foo.txt”. Tal como no caso da encriptação de informação, podemos omitir o *switch* –o, podendeo assim capturar a informação desencriptada sob a forma de uma variável. Deve no entanto ter-se sempre em conta que o *switch* –o deve surgir sempre antes do –d.

Cuidados

Tal como tudo o que tem a ver com encriptação, existem alguns cuidados que devem ser tidos em atenção. O grande cuidado que se deve ter é com o armazenamento da palavra-passe no *script* sob a forma de texto limpo. Se o código-fonte do *script* for de alguma forma acedido antes de ser realizado o processamento, a palavra-passe será descoberta.

Um segundo cuidado prende-se com a utilização do comando shell_exec do PHP. De facto, devido à utilização de um comando de *shell*, a palavra-passe fica acessível a qualquer utilizador do servidor, já que esta teve de ser *echoed*. Este método não deve ser utilizado se não se tiver total confiança de que o servidor é seguro.

Assim uma alternativa será a utilização de outros métodos como o GnuPG Made Easy (GPGME). É um projecto relativamente recente que procura desenvolver uma biblioteca para que o GnuPG possa ser embebido em aplicações e linguagens como o PHP. Quando tal for possível, não será mais necessário recorrer à utilização do comando shell_exec().

8. Conclusão

Como podemos constatar existem várias opções para a implementação de funcionalidades criptográficas num ambiente de desenvolvimento *web* com base no servidor Apache e linguagem PHP.

Cada uma das opções apresenta características distintas que a tornam mais ou menos adequadas a determinada funcionalidade. O ideal será sempre conjugar as várias hipóteses para obter um sistema mais completo e seguro.

Nunca deve ser assumido que um site é seguro. Devesse antes falar em níveis de segurança, e que determinadas características de segurança queremos que um dado site apresente. Assim, qualquer política de segurança implementada deve ter em atenção os objectivos proposto, e socorrer-se das várias opções de criptografia que estão disponíveis. Não deve ser utilizada determinada opção sem haver uma justificação válida para a sua utilização.

Este trabalho procurou dar uma ideia geral das possibilidades que estão ao dispôr de um *developer* que utilize a linguagem PHP sobre Apache. Não é contudo uma cobertura

exaustiva de todas as hipóteses, mas apresenta uma noção das várias filosofias que podem ser apresentadas.

Resta apenas referir que a criptografia quando associada á segurança de um *site* não deve ser encarada como um mero exercício técnico. Como foi já referido a criptografia é uma ferramenta ao dispôr do *developer*, e deve ser utilizada convenientemente, mas apenas quando necessário. Pior que um *site* inseguro é um *site* que se pensa, erróneamente, estar seguro devido à utilização inconsequente de funcionalidade criptográficas.

9. Bibliografia

[1] http://www.wikipedia.org

[2] http://www.modssl.org

[3] http://sambar.jalyn.net/syshelp/webdav.htm

[4] http://www.gnupg.org/

[5] http://devzone.zend.com

[6] http://www.serverwatch.com

[7] http://www.onlamp.com/php/