JOSÉ RICARDO COSME LÉRIAS RIBEIRO

UM PROTOCOLO DE AUTENTICAÇÃO DE SENHAS EM AMBIENTE CIFRADO PARA ACESSO A BASE DE DADOS

Dissertação apresentada ao programa de pós-graduação em ciência da computação da Universidade Federal de Uberlândia, como requisito parcial para obtenção do título de mestre em ciência da computação.

Área de concentração: Banco de dados.

Orientador: Prof. Dr. João Nunes de Souza, da Universidade Federal de Uberlândia.

UBERLÂNDIA – MG

2005

José Ricardo Cosme Lérias Ribeiro

UM PROTOCOLO DE AUTENTICAÇÃO DE SENHAS EM AMBIENTE CIFRADO PARA ACESSO A BASE DE DADOS

Dissertação apresentada ao programa de pós-graduação em ciência da computação da Universidade Federal de Uberlândia, como requisito parcial para obtenção do título de mestre em ciência da computação.

Área de concentração: Banco de dados.

Banca examinadora:

Uberlândia, 03 de junho de 2005.

_____

Prof. Dr. João Nunes de Souza – Orientador - UFU

_____

Prof. Dr. Júlio César López Hernández - UNICAMP

_____

Prof. Dr. Luís Fernando Faina - UFU

_____

Prof. Dr. Ilmério Reis da Silva - UFU

AGRADECIMENTOS

Primeiramente a Deus por me conceder esta invejável oportunidade de aprendizado.

À Universidade Federal de Uberlândia, em especial, à Faculdade de Computação, pela oportunidade de realizar este curso.

Aos meus pais Jair e Lourdes e ao meu irmão Júlio César, que sempre me incentivaram e não mediram esforços para que eu pudesse me dedicar a todas as atividades do curso.

Ao meu orientador, Prof. Dr. João Nunes de Souza, pela oportunidade e constante orientação, pelo incentivo durante as pesquisas, seminários, confecção de artigos e a elaboração desta dissertação e por acreditar no meu trabalho e pela confiança e apoio sempre prestados.

À minha esposa, Adeliana, que sempre me incentivou, preocupando em me ajudar e sempre compreendeu a importância deste curso.

A funcionária Madalena do departamento de computação, pela grandiosa competência e profissionalismo.

Ao meu amigo e colega do curso, Gilson Marques da Silva, que contribuiu diretamente como co-orientador.

A minha amiga e colega do curso, Fernanda de Jesus Vieira, pela atenção prestada e dicas durante a produção de artigos e realização de pesquisas.

Aos meus colegas do grupo de segurança, pela evolução conjunta alcançada, fruto de nossa união e amizade.

“Nesta era de conectividade eletrônica universal, de vírus e hackers, de espionagem e fraude eletrônica, não há momento em que a segurança não importe”.Cryptography and Netword Security, William Stallings (tradução pessoal)

SUMÁRIO

RESUMO

O objetivo desta dissertação é apresentar um protocolo de autenticação de senhas para acesso a base de dados no qual todas as etapas de transmissão e verificação de senhas são feitas em ambiente cifrado. Na autenticação, a senha e o nome do usuário são transmitidos e apresentados na forma criptografada. Os dados armazenados no servidor de logins também estão cifrados e sua manipulação é executada sem a necessidade de decifrá-los. O protocolo proposto incrementa segurança da transmissão, verificação e armazenamento de logins.

Palavras-chave: segurança, senhas, autenticação, banco de dados e criptografia.

ABSTRACT

The objetive of this dissertation is to present a password authentication protocol to access the database in which all the phases of password transmission and verifications are done in a ciphered environment. On the user authentication, the password and the username are transmitted and presented by database in ciphered way. The storaged data on the logins server are also ciphered, and its manipulation is executed without the necessity of decipher them. The proposed protocol increase the security of the transmission, verification and login storage.

Key-words: security, password, autentication, database and criptografy.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AKE: Assimetric Key Exchange

CRIPTO-SRP: Criptography Secure Remote Password

DDL: Data Definition Language

DML: Data Manipulation Language

DSS: Digital Signature Standard

EKE: Encrypted Key Exchange

LEN: Length

PSO: Protocolo de senha original

PVS: Protocolo de verificação de senha

RSA: Ron Shamir Adleman

SHA: Secure Hash Algorithm

SQL: Structured Query Language

SRP: Secure Remote Password

SSL: Secure Socket Layer

TRD: Técnica Resposta ao Desafio

TUS: Técnica Usuário Senha

LISTA DE SÍMBOLOS

A: Chave pública

a: Chave secreta

B: Chave pública

b: Chave secreta

G: Grupo multiplicativo

g: Gerador de subgrupo

H: Subgrupo de G

h: Elemento público do subgrupo H

k: Número aleatório secreto

K: Chave secreta

m: Texto plano

M1: Mensagem do usuário

M2: Mensagem do servidor de banco de dados

p: Número inteiro

P: Senha secreta

Parameter: Atributo para armazenar o identificador do usuário

S: Chave de sessão

u: Número aleatório público

User_Name: Atributo para armazenar o nome do usuário

v: Verificador da senha do usuário

Verifier: Atributo para armazenar o verificador da senha do usuário

x: Número inteiro secreto

y: Texto criptografado

y_1: Chave pública

y_2: Chave pública

z_1: Chave pública

z_2: Chave pública

1 INTRODUÇÃO

Transmitir senhas de forma segura em protocolos de autenticação tem sido um dos mais urgentes desafios na área de segurança. A leitura ou alteração de senhas em transmissão representa uma grande ameaça a todos os usuários e aos sistemas que utilizam serviços de senhas como *sites* de bancos, comércio eletrônico e sistemas corporativos. Uma vez que a senha do usuário é transmitida numa rede de computadores, esta pode ser interceptada em vários pontos. Uma forma de proteger a senha¸ sendo ela transmitida ou armazenada num computador, é criptografá-la , isto é, torná-la ilegível a todos, exceto àqueles que realmente devem ter acesso a ela.

Para criptografar a senha e enviá-la em uma rede de computadores é necessário usar um protocolo de segurança que garanta: autenticidade, confidencialidade e integridade. Como a senha é uma informação secreta, o usuário deve provar ao servidor, de algum modo, que a senha é autêntica, ou seja, pertence ao usuário que está tentando se autenticar. A confidencialidade deve ser garantida em sistemas de autenticação de senhas, pois se a senha é revelada a um intruso, ele pode utilizá-la em autenticações futuras. Finalmente, a integridade permite ao servidor verificar se a senha não foi alterada durante a transmissão.

Atualmente, os sistemas de autenticação de senhas usam protocolos de segurança como o *SSL* (*Secure Socket Layer*) [STALLINGS 1999] para transmitir dados criptografados. Mas o protocolo *SSL* não é específico para transmitir senhas, ou seja, ele criptografa e transmite qualquer informação. Um protocolo para autenticação de senhas é o protocolo *SRP* (*Secure Remote Password*) [WU 2002] que é utilizado para garantir a segurança nas autenticações. Este protocolo utiliza troca de chaves e cálculos matemáticos para garantir a segurança da senha autenticada**.**

Um dos objetivos deste trabalho é apresentar uma proposta para aumentar a segurança no processo de autenticação de senhas. Como se sabe, o protocolo *SRP* utiliza um verificador para autenticar a senha do usuário, sem precisar transmiti-la. Esta funcionalidade fez com que o protocolo *SRP* fosse escolhido como base para a proposta apresentada nesta dissertação.

Entretanto o *SRP* não criptografa o nome do usuário a ser verificado e ainda não implementa as funcionalidades de segurança aqui apresentadas. Ao verificar o nome e a senha do usuário esta proposta utiliza operações modulares em grupos finitos e exponenciais não utilizadas pelo protocolo *SRP*, aumentando assim a segurança na fase de autenticação de senhas.

O protocolo proposto é uma extensão do protocolo *SRP* denominado ***Cripto-SRP***. Este protocolo trabalha sobre a camada de aplicação.

Um outro objetivo deste trabalho é a proposta de um sistema onde é possível efetuar consultas a uma base de dados cifrada sem decifrar os dados. Neste banco de dados são armazenadas tuplas que contém os nomes dos usuários na forma cifrada. Desta forma, cada usuário utiliza o banco de dados verificando os nomes sem decifrá-los. Assim, com a utilização do protocolo, os nomes dos usuários são transmitidos e manipulados na forma cifrada, melhorando a segurança nos processos de autenticação de senhas.

O estudo de autenticação utilizando senhas é um tema importante em segurança de base de dados e de sistemas de informação em geral. Várias propostas têm sido apresentadas, como por exemplo, em [BELLOVIN 1992] onde é proposto o uso do protocolo *EKE* (*Encrypted Key Exchange*) para proteger senhas contra ataques de dicionário. Como evolução desta proposta estes mesmos autores criaram mecanismos de proteção para arquivos compromissados [BELLOVIN 1994]. Um mecanismo de autenticação de senhas utilizando o protocolo *AKE* (*Assimetric Key Exchange*) é apresentado por [JABLON 1996]. Já em [WU

2002] é proposto o protocolo *SRP* (*Secure Remote Password*). Este protocolo utiliza troca de chaves e função *hash* [STALLINGS 1999] para garantir integridade na transmissão e autenticidade do usuário. Outras propostas também podem ser encontradas em [MACKENZIE 2001], [BELLARE 2000],[KWON 2000], [IVES 2004], [HER 2002], [PINKAS 2002], [MONROSE 1999].

Em todas estas propostas, não é considerado o contexto apresentado nesta dissertação onde os usuários manipulam os dados na forma cifrada. Isto significa que em propostas que têm sido estudadas, os dados de *login* devem ser protegidos, o que não ocorre com a proposta apresentada nesta dissertação.

Esta dissertação está estruturada em cinco capítulos. O primeiro, introduz a proposta e estudos realizados; O segundo, apresenta os fundamentos de criptografia; No terceiro, é apresentado o protocolo de autenticação de senhas *SRP*. O protocolo ***Cripto-SRP*** é apresentado no capítulo 4. E por fim, as conclusões são apresentadas no capítulo 5.

Os artigos e pôsteres relacionados a seguir, na ordem de publicação e aceitação, são trabalhos decorrentes da proposta apresentada nesta dissertação.

- **Melhorias em transmissão, armazenamento e checagem de logins em base de dados cifrados.**

  Publicado no **IMESA** em Agosto de 2004 em Assis/SP.

- **Uma extensão do protocolo SRP para transmissão segura de logins com acesso a base de dados cifrados.**

  Publicado no SECONF'04 em Novembro de 2004 em RioVerde/GO.

- **Transmissão segura de *logins* com acesso a base de dados cifrados.**

  Publicado no **I2TS -** International Information and Telecommunication Technologies em Dezembro de 2004 em São Carlos/SP.

- **Um sistema de autenticação de senhas para redes locais sem fios.**

  Aceito no Sucesu 2005 – Congresso nacional de tecnologia da informação e comunicação em Janeiro de 2005 em Belo Horizonte/MG.

2 FUNDAMENTOS DE CRIPTOGRAFIA

Este Capítulo tem como objetivo apresentar os fundamentos de criptografia, os princípios da área de segurança em computação e de como são construídos os protocolos criptográficos. Ele está estruturado da seguinte forma: a seção 2.1 apresenta a definição e os tipos de criptografia existentes. A seção 2.2 compara a criptografia simétrica e assimétrica e o uso da função *hash*. Já na seção 2.3 são apresentadas as técnicas de assinatura digital. E por fim, os ataques à segurança são apresentados na seção 2.4.

2.1 Visão Geral

Criptografia é a arte ou ciência de escrever em cifra ou em códigos. Transforma informações originais em informações codificadas, chamada de informação criptografada. Esta ciência visa proporcionar à informação transmitida ou armazenada, funcionalidades de segurança, tais como:

Confidencialidade: Garantir que a informação armazenada ou transmitida em um sistema de computador seja acessível apenas a pessoas autorizadas, garantindo sigilo da informação.

Integridade: Garantir ao receptor que a informação não foi alterada durante a transmissão, podendo esta ser alterada apenas pelas partes autorizadas.

Autenticidade: Garantir que a origem da informação transmitida está corretamente identificada e que a mesma pertence ao remetente que iniciou a transmissão.

Não recusa: Um remetente não poderá negar uma informação por ele enviada.

Criptografar: É o ato de codificar informações em alguma forma ilegível. O objetivo é tornar uma informação ilegível às pessoas não-autorizadas mesmo que estas consigam visualizar os dados criptografados.

Decriptografar: É o ato de decodificar informações, ou seja, transformar informação criptografa em informação original.

Para criptografar (C) ou decriptografar (D) uma informação (I) , é necessário um dado secreto denominado de chave (K). A segurança da criptografia baseia-se no uso desta chave que é uma espécie de código secreto que, combinado à informação criptografada, permite decriptografá-la.

O processo de criptografar é dado por:

$$I\text{-}Cripto = Cripto_k(I)$$

O processo de decriptografar é dado por:

$$I\text{-}Decripto = Decritpo_k(I)$$

Tipos de criptografia

A criptografia pode ser classificada em duas categorias, de acordo com o tipo de chave utilizada, sendo estes dois grupos, criptografia simétrica e assimétrica.

Criptografia simétrica

A criptografia de chave simétrica utiliza a mesma chave para criptografar e decriptografar uma informação. Para tanto, é necessário estabelecer um acordo entre o emissor e o receptor para que a mesma chave seja utilizada antes do início do processo de envio e recebimento da informação criptografada, descrito no exemplo a seguir.

Alice (A) deseja enviar uma informação para Bob (B), eles devem escolher uma chave secreta K antes de iniciar a transmissão. Após a escolha da chave secreta K, Alice criptografa a informação original com esta chave e envia a informação criptografada para Bob:

$$I\text{-}Cripto = Cripto_K(I)$$

De posse da chave *K*, Bob decriptografa a informação *I*. O processo de decriptografar é dado por:

$$I = Decritpto_K\ (I\text{-}Cripto)$$

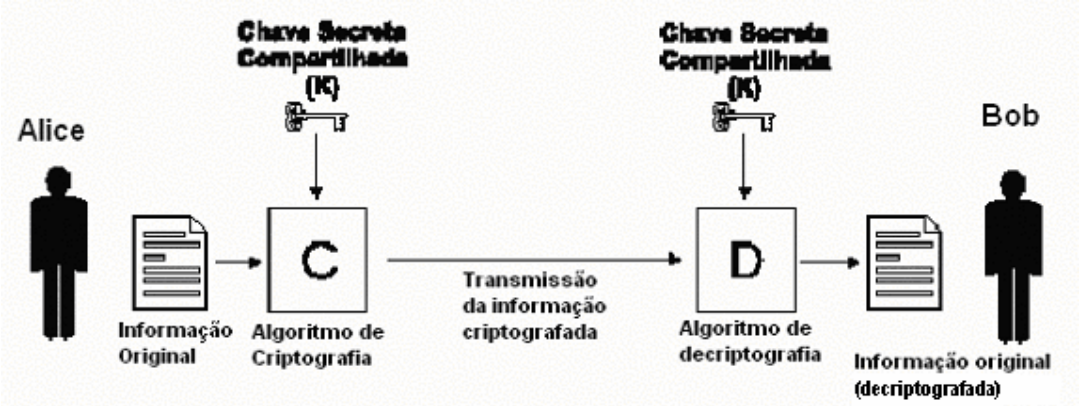


Figura 1: Um sistema de criptografia simétrica.

Na figura 1, Alice criptografa a informação original com a chave secreta *K*, e envia para Bob. Ao receber a informação criptografada, Bob utiliza a mesma chave secreta, *K*, para decriptografar a informação criptografada.

Criptografia assimétrica

A criptografia assimétrica também chamada de criptografia de chave pública utiliza de duas chaves, uma pública (*KU*) e uma secreta (*KR*). Neste caso estas chaves devem ser diferentes, ou seja, $KU \neq KR$. Ao utilizar uma chave pública (*KU*) para criptografar uma informação, deve-se utilizar uma chave privada (*KR*) para decriptograr esta informação. A chave pública é distribuída abertamente a qualquer pessoa. Pode ser publicada em catálogos, em *sites* da internet, enfim, deve se tornar conhecida de todos. A segurança dos sistemas criptográficos simétricos e assimétricos está na chave secreta, em geral quanto maior for o

tamanho desta chave em *bits* mais seguro é o sistema criptográfico. A chave secreta deve ser mantida em segredo, enquanto a chave pública deve se tornar pública, conforme é descrito no exemplo a seguir:

Alice (A) deseja enviar uma informação para Bob (B). Bob possui duas chaves, uma pública KU_b que é conhecida por ambas as partes, Alice e Bob, e uma chave secreta KR_b.

Para criptografar uma informação (*I*) a ser enviada para Bob, Alice utiliza a chave pública de Bob:

$$I\text{-}Cripto = C_{KUb}\,(I)$$

Ao receber a informação criptografada *I-Cripto*, Bob consegue decriptografar esta informação utilizando a sua chave secreta KR_b:

$$I = Decripto_{KRb}(I\text{-}Cripto)$$

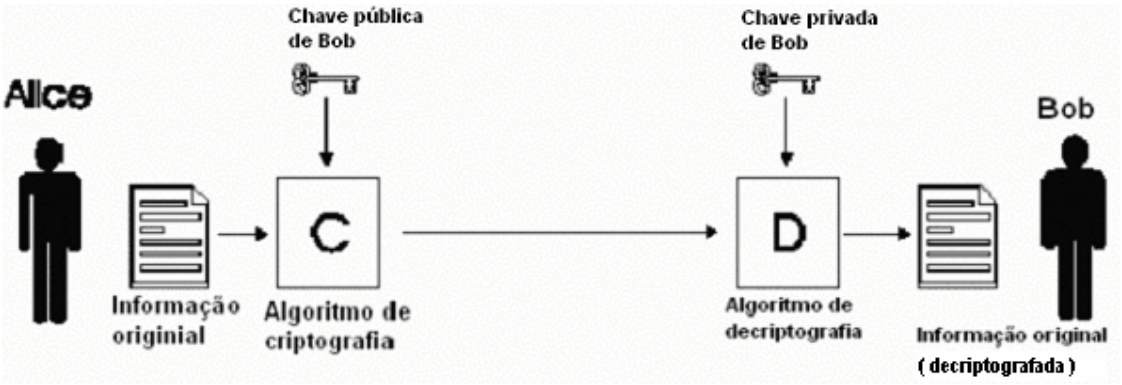


Figura 2: Um sistema de criptografia assimétrica

A figura 2 mostra que, para criptografar uma informação original, Alice utiliza a chave pública de Bob. Para decriptografar a informação, Bob utiliza sua chave secreta. Somente Bob poderá decriptografar a informação criptografada, pois somente ele conhece KR_b.

Grupos finitos

A implementação da maioria dos sistemas criptográficos de chave pública (RSA, ElGamal, etc.) é baseada em grupos multiplicativos como o $\mathbb{Z}_p$. Entretanto, outros grupos também são utilizados [KOBLITS 1994], [MENEZES 1997], [ROSIGN 1999], [SCHNEIER 1996], [STALLINGS 1999], [STINSON 1995]. O criptossistema das curvas elípticas de Koblits e Miller, por exemplo, é baseado em grupos elípticos os quais são definidos a partir de curvas elípticas [MENEZES 1993]. O criptossistema da sacula de Merkle-Hellman é baseado no problema da sacula [MERKLE 1978] e o criptossistema de McEliese é baseado na teoria de códigos [STINSON 1995]. Em geral, a segurança destes sistemas se fundamenta na suposta intratabilidade do problema do logaritmo discreto, o qual é objeto de grande estudo [GORDON 1993].

Problema do logaritmo discreto em um grupo multiplicativo *G*

Seja $(G, *)$ um grupo finito com a operação multiplicação * e g um gerador de um subgrupo H de G. Logo, tem-se que $H = \{g^i ; i \geq 0\}$. Dado $h \in H$, o problema do logaritmo discreto em um grupo multiplicativo G consiste em encontrar um inteiro a, tal que:

$$0 \leq a \leq |G| - 1 \text{ e } g^a = h$$

Neste caso, a notação g^a significa a multiplicação de g por si mesmo a vezes. O inteiro a é denotado por $log_g h$ (Equação 2-1). Utilizando a seguinte notação:

$$a = log_g h \qquad (2\text{-}1)$$

O problema do logaritmo discreto é considerado como sendo computacionalmente difícil se o grupo multiplicativo G e o gerador g são escolhidos com cuidado. Em particular, não se conhece um algoritmo de tempo polinomial que resolva o problema do logaritmo

discreto em $\mathbb{Z}_p$, se p contém pelo menos 150 dígitos e $p-1$ tem pelo menos um fator primo grande [GORDON 1993].

Criptossistema de ElGamal em um grupo multiplicativo *G*

Assume-se que G é um grupo finito com a operação multiplicação, e g um gerador do subgrupo H de G, tal que o problema do logaritmo discreto é considerado como sendo intratável em H.

No criptossistema de ElGamal [ELGAMAL 1985], os textos originais são elementos de G e os textos criptografados são pares em GxG. Inicialmente escolhe-se $h \in H$, dado a, $h = g^a$ (Equação 2-2) tal que:

$$g^a = h \qquad (2\text{-}2)$$

Os elementos h e g são públicos e a é secreto. Para um número aleatório secreto $k \in \mathbb{Z}_{|H|}$, define-se a codificação de um elemento $m \in G$, dado m, escolhe-se $k \in \mathbb{Z}_{|H|}$ tal que:

$$E_a(m,k) = (y_1, y_2) \qquad (2\text{-}3)$$

Onde:

$$y_1 = h^k$$

$$y_2 = m\, g^k$$

Dado o texto criptografado $y = (y_1, y_2)$ (Equação 2-3), a decifragem é dada por:

$$D_a(y) = (y_2 y_1^{a^{-1}})^{-1}$$

Tem-se que:

$$y_1^{a^{-1}} = h^{ka^{-1}} = g^{aka^{-1}} = g^k$$

Logo:

$$D_a(y) = y_2(y_1^{a^{-1}})^{-1}$$

$$D_a(y) = m\, g^k (g^k)^{-1}$$

$$D_a(y) = m$$

No criptossistema de ElGamal [ELGAMAL 1985], a chave privada é *a* e a chave pública são os valores *g, h* e o grupo multiplicativo *G*. Observe que o texto original *m* é criptografado usando unicamente elementos públicos. Por outro lado, o par (y_1, y_2) é decriptografado usando a chave secreta *a*.

Esquema de Assinatura ElGamal em $\mathbb{Z}_p$

Seja *p* um número primo, tal que o problema do logaritmo discreto seja intratável em $\mathbb{Z}p$ e *g* um gerador de $\mathbb{Z}_p^*$. No esquema de assinatura Elgamal**,** um texto plano *m* é um elemento de $\mathbb{Z}p$ e a assinatura de *m* é um par em $\mathbb{Z}p$ $\boldsymbol{x}$ $\mathbb{Z}_{p-1}^*$**.** Dado *a*, calcule $h \in H$:

$$h \equiv g^a (mod\ p) \qquad (2\text{-}4)$$

Os elementos *h* (Equação 2-4) e *g* são públicos e *a* é secreto. Para um número aleatório secreto $k \in \mathbb{Z}_{p-1}^*$, a assinatura de *m* é definida por:

$$sig_a(m,k) = (z_1, z_2) \qquad (2\text{-}5)$$

Onde:

$$z_1 \equiv g^k (mod\ p) \qquad (2\text{-}6)$$

$$z_2 \equiv (m - a\, z_1)\, k^{-1} (mod\ (p-1)) \qquad (2\text{-}7)$$

Dada a assinatura $z = (z_1, z_2)$ (Equação 2-5) de *m***,** a sua verificação é feita utilizando a função de verificação *ver*, como apresenta a seguir:

$$ver(m,z) = true \qquad h^{z_1}(z_1)^{z_2} \equiv g^m \pmod{p}$$

A correção da verificação da assinatura é demonstrada a seguir:

$$h^{z_1}(z_1)^{z_2} = g^{az_1} g^{kz_2} = g^{az_1 + kz_2} \equiv g^m \bmod p$$

Onde *a* e *k* são consideradas chaves secretas, já z_1 (Equação 2-6) e z_2 (Equação 2-7) são consideras chaves públicas tal que:

$$a\, z_1 + k\, z_2 \equiv m \bmod (p-1)$$

2.2 Comparação entre criptografia simétrica e assimétrica

Na criptografia assimétrica, as duas chaves usadas por cada parte são denominadas respectivamente como chave pública (*KU*) e chave secreta (*KR*). Por outro lado, a criptografia simétrica utiliza apenas uma chave secreta conhecida por ambas as partes. Comparada à criptografia simétrica, a criptografia assimétrica necessita de maior esforço computacional. Por isso, em geral ela não é apropriada para ser utilizada em casos onde há necessidade de criptografar um grande volume de informação tendo aplicação principalmente em protocolos que utilizam assinatura digital e troca de chaves [WU 1999]. Por outro lado, estudos feitos a partir dos algoritmos de criptografia assimétrica demonstram que o aumento no tamanho das chaves usadas pode tornar a segurança praticamente inviolável [STALLINGS 1999]. Porém, neste caso, o esforço computacional para a execução dos algoritmos se torna progressivamente maior. Esta característica resulta, portanto, em uma redução de performance, à medida que se aumenta o tamanho da chave. A tabela 1 apresenta uma comparação entre criptografia simétrica e assimétrica.

	Simétrica	Assimétrica
Tempo necessário para criptografar grande volume de informação	Baixo	Alto
Permite a assinatura digital	Não	Sim
Gerenciamento de chave	Menor segurança	Maior Segurança
Quantidade de chaves	1(secreta)	2(pública e secreta)

Tabela 1: Comparação entre criptografia simétrica e assimétrica

Uma das principais características da criptografia assimétrica é a distribuição de chaves. Na criptografia simétrica, há a necessidade do compartilhamento de uma chave secreta e, portanto, de um pré-acordo entre as partes interessadas. Tal fato não ocorre na criptografia assimétrica. Neste sentido, a vulnerabilidade do sistema assimétrico é menor. Por outro lado, a principal vantagem da criptografia simétrica é a velocidade dos processos de cifragem/decifragem. Em geral, os sistemas criptográficos simétricos são mais rápidos que os assimétricos.

Garantir a segurança dos dados criptografados é o desafio de todo sistema criptográfico em máquinas hostis, onde intrusos, podem ter acesso à memória principal do servidor e as senhas ali armazenadas. Em geral, para processar uma consulta em banco de dados onde as senhas estão armazenadas na forma criptografada, os sistemas de segurança fazem a leitura de blocos de dados na memória secundária e os decifram na memória principal do servidor do banco de dados. Considerando máquinas hostis, isto pode comprometer a segurança. Outra estratégia é utilizar funções *hash*, como é descrito a seguir.

Uso da função *Hash*

Uma função *hash* é uma função unidirecional conhecida como função resumo ou *message digest*. É uma função que aplicada a um documento ou mensagem de qualquer tamanho, gera um resumo de tamanho fixo. A função *hash* pode ser utilizada, por exemplo, para garantir a integridade dos dados no processo de assinatura. Ao coletar o resultado de uma função *hash* aplicada a uma senha, o intruso não tem acesso à senha. Neste caso, o servidor

armazena o resultado da função *hash* aplicada à senha. Este resultado é comparado com o valor do *hash* transmitido pelo usuário, que é igual ao valor da função *hash* aplicada à senha.

Assinatura de Senhas utilizando função *hash*

Considerando um sistema de assinatura de senhas onde cada usuário escolhe sua senha individual e secreta, denominada de *w, o* servidor de banco de dados armazena a imagem *f(w)* da senha *w*, onde *f* é uma função *hash*. Ao receber a senha *w*, o servidor determina *f(w)* (imagem da senha *w*) e compara o resultado com o valor armazenado. Esse método de autenticação de senhas não é seguro, pois o intruso pode capturar a senha *w* (valor secreto da senha) durante a transmissão. Para evitar este tipo de ataque, o usuário calcula e transmite *f(w)*. Neste caso, o intruso detecta *f(w)* e não consegue determinar *w*. Entretanto, este esquema pode sofrer o ataque do meio, tipo de ataque que ocorre quando o intruso se faz passar por um usuário.

As funções *hash* também podem ser utilizadas para autenticar mensagens, garantindo ao destinatário que a mensagem recebida não sofreu alteração durante sua transmissão. Estas funções em conjunto com recursos de criptografia, podem ser amplamente utilizadas em sistemas de assinatura digital. Alguns exemplos de funções *hash* são: MD4, MD5 e RIPEMD-160 [SCHNEIER 1996] e [MENEZES 1993].

2.3 Técnicas de assinatura digital

O conceito de assinatura digital se assemelha ao da assinatura manuscrita, onde nesta última uma pessoa assina em um papel ou documento. Na assinatura manuscrita o que está sendo assinado é o papel e ao utilizar a assinatura digital o que se assina é a informação. Segundo Menezes [MENEZES 1993], a “assinatura digital de uma mensagem é um número

que depende de um valor secreto conhecido apenas pelo próprio assinante e do conteúdo da própria mensagem assinada".

O resultado da criptografia de uma informação (documento), utilizando uma chave secreta por quem assina, é uma informação com assinatura digital. Por sua vez, para verificar esta assinatura o receptor utiliza a chave pública correspondente. Se o resultado da decriptografia for igual à informação original, a assinatura é considerada verdadeira, autêntica, ou seja, a informação veio do rementente que afirma ter enviado a informação. Isto ocorre porque apenas o proprietário da chave secreta, poderia ter gerado a informação criptografada. A figura 3 ilustra um exemplo de assinatura digital utilizando função *hash*.

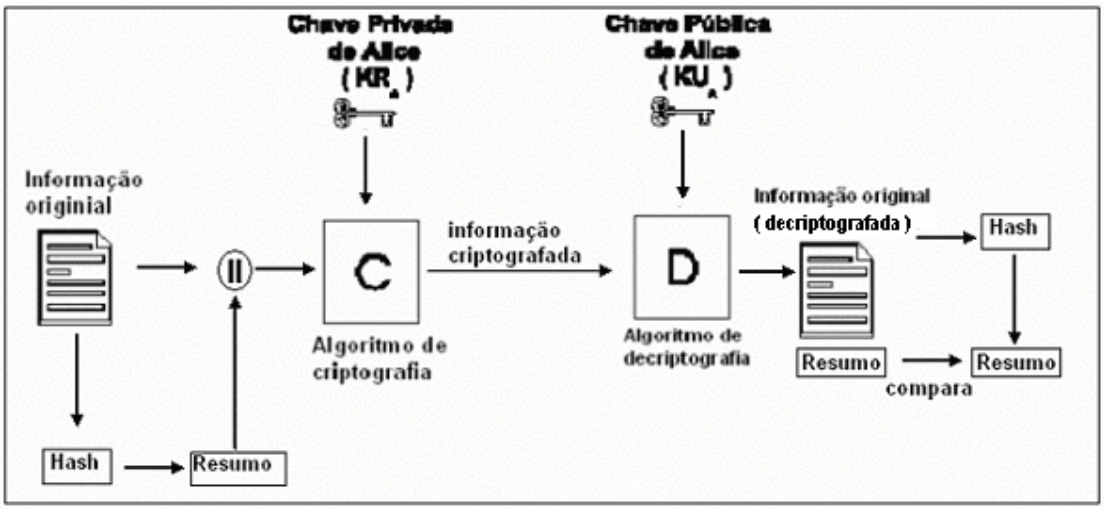


Figura 3: Assinatura digital utilizando função *hash*

Na Figura 3, Alice calcula o resumo da informação através da função *hash* e criptografa este resumo, juntamente com a informação original, com sua chave privada, KR_A. O resultado é então enviado para Bob**.** Bob, por sua vez decriptografa a informação concatenada com o resumo, utilizando a chave pública de Alice, $\boldsymbol{KU_A}$. Em seguida ele calcula o resumo da informação recebida e compara os dois resultados. Se os dois resultados forem iguais, ou seja, o resumo concatenado a informação for igual ao resumo gerado por Bob, tem-se a certeza de que a mensagem não foi alterada.

A assinatura digital tem sido implementada basicamente de duas formas [STALLINGS 1999]: função *hash* por meio dos padrões MD5, SHA (*Secure Hash Algorithm*) e DSS (*Digital Signature Standard*) e utilizando o conceito de chaves públicas com o padrão RSA.

Certificado digital

O certificado digital é um arquivo intransferível e assinado de forma digital por uma entidade certificadora. A divulgação da chave pública e a associação desta a uma pessoa ou entidade são os objetivos do certificado digital.

A entidade certificadora (AC) assina o certificado com sua chave secreta. A comprovação da autenticidade do certificado é feita com a verificação da assinatura deste utilizando a chave pública de AC. Nos sistemas de autenticação de senhas podem ser utilizados vários tipos de certificados, que se destinam a vários fins. Entretanto, um dos certificados mais importantes e utilizados é o certificado de chave pública.

A recomendação utilizada para certificados digitais é o X.509v3, elaborado pela ITU-T. As seguintes informações compõem a estrutura de dados de um certificado: chave pública, identificador do proprietário, número de série do certificado, nome da entidade certificadora que emitiu o certificado, assinatura digital desta entidade. A figura 4 ilustra, a estrutura de dados do certificado.

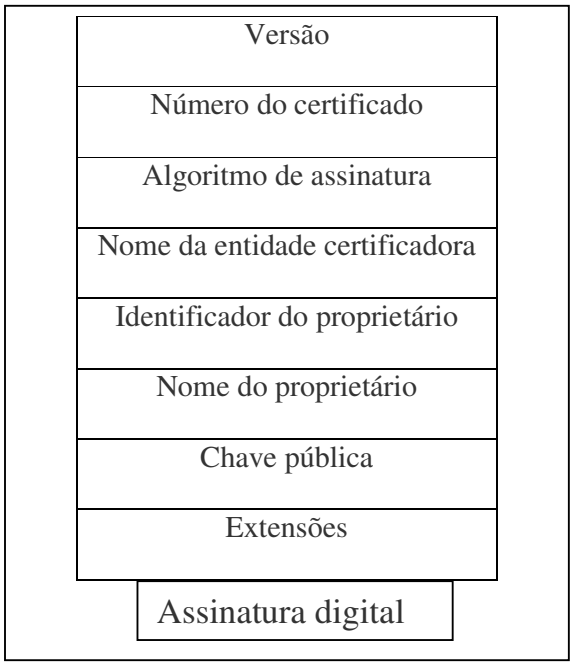


Figura 4: Certificado digital

2.4 Ataques à segurança

Os ataques realizados por pessoas não autorizadas com o objetivo de obter informações secretas, sendo estas protegidas por sistemas criptográficos, são caracterizadas como ataques a segurança. Os tipos de ataques descritos nesta seção pretendem demonstrar como os sistemas de segurança podem ser ameaçados. A partir dos estudos destas fragilidades é que são desenvolvidos sistemas mais seguros.

Em uma rede de computadores, a informação parte de uma origem e flui para um destino. Este fluxo de informação é conhecido como transmissão. Durante esta transmissão a informação pode sofrer ataques. Estudos sobre estes ataques são apresentados em [STALLINGS 1999] e detalhados a seguir:

Ataque somente à informação criptografada: O intruso obtém alguma informação criptografada, mas desconhece a original e as chaves secretas utilizadas. Seu objetivo é recuperar a informação original.

Ataque por meio de interceptação: Ocorre quando um intruso obtém acesso à informação, podendo visualizá-la ou copiá-la sem autorização. Este tipo de ataque diz respeito ao serviço de confidencialidade.

Ataque de chave escolhida: O intruso utiliza chaves diferentes ou tenta convencer usuários legítimos a utilizarem determinadas chaves. Neste caso, o objetivo é decriptografar as informações com estas chaves.

Ataque por força bruta: Conhecido também como busca exaustiva, este ataque permite ao intruso decriptografar a informação utilizando todas as possíveis chaves.

Ataque por meio de modificação: Ocorre quando um intruso, além de obter a informação, consegue ainda modificá-la durante a sua transmissão. Esta alteração caracteriza um ataque a integridade da informação transmitida.

Ataque adaptativo a texto escolhido: O intruso é capaz de escolher textos originais. Ele também é capaz de escolher novos textos originais como a função de textos cifrados obtidos, mas não conhece a chave. Ele tenta descobrir a chave ou decifrar outros textos, a partir dos textos cifrados e originais previamente escolhidos.

Para se ter segurança em um protocolo de autenticação de senhas é necessário utilizar os fundamentos da criptografia, que são: A criptografia simétrica, e assimétrica, função *hash*, certificado digital e assinatura digital. Neste capítulo foram explicados estes fundamentos que são essenciais ao entendimento do protocolo de autenticação de senhas proposto neste trabalho.

3. PROTOCOLO *SRP*

Este capítulo tem como objetivo apresentar o protocolo *SRP* (*Secure Remote Password*) [WU 1999] utilizado em autenticações de senhas. Este protocolo de troca de chaves e autenticação foi projetado para proteger o valor da senha em um canal de transmissão. Também é apresentada uma análise da segurança do protocolo *SRP*, estruturado em seções. Na seção 3.1 são apresentados os protocolos e técnicas de autenticações. E na seção 3.2 apresenta a primeira versão do protocolo *SRP*, conhecido como *SRP-3*. Quanto a esta seção 3.3, apresenta melhorias no protocolo *SRP*, conhecido como *SRP-6*. E, por ultimo, na seção 3.4 são apresentadas às análises da segurança deste protocolo.

3.1 Protocolos e técnicas de autenticação

A preocupação em transmitir informações secretas, como senhas, garantindo autenticidade, é a motivação para a criação de protocolos de autenticação. Tal fato tem se tornado relevante, pois nos últimos anos tem aumentado o número de ataques a usuários que utilizam redes públicas, como a internet, para realizar operações bancárias ou qualquer outro tipo de serviço que necessite de uma senha secreta para efetuar a autenticação.

Os protocolos de autenticação utilizam métodos de troca de chave e assinatura digital, para garantir a autenticidade da informação transmitida. O protocolo *SRP* tem como objetivo garantir que o valor da senha transmitida seja autêntico, ou seja, para que o servidor de dados tenha a garantia de que a senha pertence ao usuário que iniciou a autenticação. Por isso, este protocolo implementa o uso de uma função *hash* para garantir a autenticidade além

de cálculos matemáticos, tendo como resultado duas chaves públicas geradas a partir de chaves secretas conhecidas apenas pelo usuário e pelo servidor de dados.

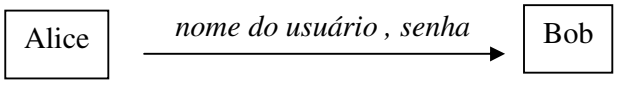
Para iniciar um processo de autenticação, inicialmente o usuário faz uma requisição ao servidor de dados. Este envia um desafio ao usuário. Se a resposta ao desafio é tida como sucesso, então a transmissão é iniciada. Caso contrário, o processo de autenticação é cancelado. Ao iniciar uma transmissão o servidor de dados deve dispor de um protocolo de autenticação para validar a senha do usuário. Há dois tipos de protocolos de autenticação:

- *Protocolo de senha original (PSO)*: Protocolo de autenticação no qual uma cópia da senha do usuário é armazenada no servidor.
- *Protocolo de verificação de senha (PVS)*: Protocolo de autenticação no qual apenas um verificador da senha é armazenado no servidor.

Os protocolos *PVS* têm uma vantagem significante sobre os protocolos *PSO*. Isto ocorre porque os sistemas que utilizam *(PSO)* podem ser comprometidos uma vez que o banco de dados seja revelado. A seguir são apresentadas duas técnicas de autenticação.

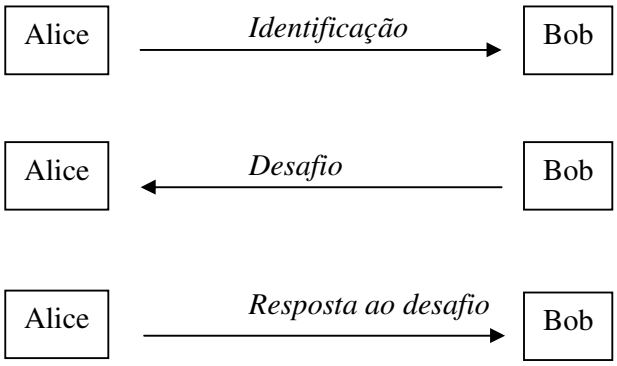
3.1.1 Técnicas de autenticações:

Esta seção descreve duas técnicas de autenticação. A primeira é denominada: *TUS* (*Técnica Usuário Senha*) que, neste caso, a usuária *Alice* se autentica enviando para *Bob* seu nome do usuário e senha.



A segunda técnica é denominada: *TRD (Técnica Resposta ao Desafio),* neste caso, o usuário se autentica enviando uma resposta a um desafio.

- Técnica *TRD*:



No exemplo acima, *Alice* apresenta inicialmente sua identificação a *Bob*. Após receber a identificação de *Alice*, *Bob* envia um desafio. Em seguida, *Alice* envia uma resposta ao desafio que é verificado por *Bob*.

Desde que o valor do *Desafio* seja diferente para cada tentativa de autenticação, uma *resposta ao desafio* capturado é inútil para outras seções.

Os valores do *Desafio* e a *resposta ao desafio* de uma autenticação devem ser tais que, se forem capturados é impossível descobrir a senha (ataque de dicionário).

3.1.2 Outros protocolos de autenticação

Esta seção apresenta soluções mais seguras para realizar autenticação, e são apresentados dois protocolos:

Protocolo de troca de chaves cifrada (*EKE):*

Em 1992 Bellovin e Merritt [MACKENZIE 2001] apresentaram um novo protocolo conhecido como troca de chave criptografada (*EKE*). Por usar uma combinação de criptografia simétrica e de chave pública, *EKE* resiste a ataques de dicionários, pois não fornece informação suficiente para revelar a senha secreta. Na forma mais geral do *EKE*, as duas partes cifram suas chaves públicas com criptografia simétrica, usando uma senha secreta compartilhada como chave. A família dos protocolos *EKE* representa um nível seguro de autenticação. Porém a maior falha do protocolo *EKE* é que ele usa um protocolo de autenticação de senha original *(PSO)* e não possui um verificador de valores secretos e tudo isto pode comprometer o valor da senha transmitida.

Protocolo de troca de chave assimétrica *(AKE):*

Como no protocolo de troca de chave cifrada **(***EKE)*, a função primária do *AKE* é a troca de chaves entre duas partes e o uso desta chave para verificar que ambos conhecem sua senha. Diferente do protocolo *EKE*, o *AKE* não cifra nenhuma informação, apenas utiliza relações matemáticas para combinar os valores trocados com os parâmetros da senha estabelecidos. Eliminar a cifragem é vantajoso pelas seguintes razões:

1. Simplifica o protocolo eliminando a necessidade de negociar um algoritmo de criptografia comum.

2. Qualquer vulnerabilidade na cifragem irá resultar em uma fraqueza no protocolo de autenticação.

O protocolo de troca de chaves assimétrica *(AKE)* descreve uma abordagem, na qual cada parte calcula uma chave secreta e aplica uma função unidirecional (*hash*) para gerar um verificador da senha, que é compartilhado com a outra parte. Para o protocolo *AKE* funcionar, é necessário que a seguinte equação seja satisfeita:

$$S(R(P(w),P(x),Q(y,z))) = S(R(P(y),P(z)),Q(w,x))$$

Onde se tem que:

w,x,y,z	*P(x)*	*Q(x,y), R(x,y)*	*S(x,y)*	*K*
Parâmetros arbitrários	Função ***hash*** de verificação	Funções "***mixing***" para parâmetros públicos e privados	Função de geração da chave de sessão	Chave de sessão

Tabela 2: Parâmetros e funções do protocolo *AKE*

A segurança do protocolo está associada à escolha das funções *P( ), Q( ), R( )* e *S( )*, onde *x* e *z* são termos secretos de cada parte e *w* e *y* são parâmetros gerados por cada parte para garantir que suas chaves de sessão mudem de uma para outra.

3.2 Protocolo *SRP*

Esta seção apresenta o protocolo *SRP* (*Secure Remote Password*) em sua primeira versão. Na seção 3.3 é apresentada à versão deste protocolo publicada em 2002, conhecida como *SRP-6*.

Em 1996, Thomas Wu, lançou à idéia de desenvolver um protocolo específico para transmitir e autenticar senhas de forma secreta. A idéia inicial para o desenvolvimento do protocolo *SRP* é conhecida como *SRP-3* que foi publicada em 1999 [WU 1999]. Atualmente, tem-se publicada a versão *SRP-6* [WU 2002], que mantém a segurança da versão *SRP-3*, mas com melhorias para a portabilidade e facilidade de ser incorporado em sistemas existentes. Detalhes técnicos do atual projeto *SRP* estão disponíveis em [WU 2002], sendo o mesmo de código aberto.

O protocolo de segurança remota de senhas (*SRP*) combina técnicas de prova de conhecimento zero [WU 1999] com protocolos de troca de chave assimétrica [STALLINGS 1999]. Ele oferece um ganho de performance se comparado com outros protocolos de troca de chave cifrada com assinatura digital [WU 1999]. Além disso, o protocolo *SRP* não é vulnerável ao ataque do meio como ocorre com protocolos que utilizam apenas funções *hash* [STALLINGS 1999].

Na definição do protocolo *SRP* são feitas as seguintes considerações:

- Assume-se que nenhuma terceira parte como servidor de chaves *(Key Server)* é utilizada, sendo que somente as duas partes envolvidas participam do protocolo de autenticação.
- O valor da senha *(password)* e o verificador da senha *(verifier),* correspondem respectivamente à *chave privada* e *pública*. A chave pública é facilmente calculada a partir do valor privado da senha, mas derivar o valor da senha a partir do verificador da senha é computacionalmente inviável.

O protocolo *SRP* é uma implementação de um protocolo da família *AKE,* onde, todos os cálculos são feitos em um grupo finito *GF(n)* [KOBLITZ 1994] e todos os parâmetros de entrada e saídas de *P(), Q(), R()* e *S()* são inteiros entre *0* e *n-1*. Neste caso tem-se que:

$$P(x) \equiv g^x \ (mod\ n)$$

Onde *g* é o gerador de *GF(n)*

$$Q(w,x) \equiv w + ux \ (mod\ n)$$

$$R(w,x) \equiv wx^u \ (mod\ n)$$

$$S(w,x) \equiv w^x (mod\ n)$$

$$u = f(w,x)$$

Para estabelecer uma senha com *Bob*, *Alice* usa palavra string randômica *s,* enviada por *Bob* em cada execução do protocolo *SRP* e calcula:

$$x = H(s,P) \qquad (3\text{-}1)$$

$$v = g^x \bmod n \qquad (3\text{-}2)$$

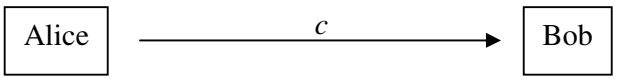
Onde *H* é uma função *hash* e *P* é a senha da Alice

Bob armazena *s* e o verificador da senha de *Alice*, denominado por *v* (Equação 3-2). Neste caso *s* e *v* são públicos.

Os passos a seguir definem o protocolo *SRP*.

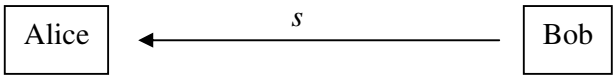
- **Passo 1**

Alice envia para *Bob* seu nome de usuário, indicado por *c***.**



- **Passo 2**

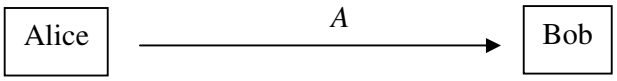
A partir do valor *c*, *Bob* acessa o valor *s*. Em seguida *Bob* envia *s* para *Alice*. De posse de *s*, *Alice* calcula sua chave privada *x* (Equação 3-1) usando *s* e *P***,** onde *P* é a senha secreta de *Alice*.



$$x = H(s, P)$$

- **Passo 3**

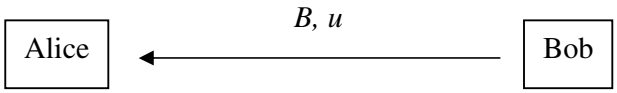
Alice gera um número aleatório *a, 1<a<n*, calcula sua chave pública *A* (Equação 3-3) e envia para Bob. Somente *Alice* pode gerar a chave pública *A*, pois o número aleatório *a* é secreto e só é conhecido por *Alice*.



$$A \equiv g^a \bmod n \qquad (3\text{-}3)$$

- **Passo 4**

Bob gera dois números aleatórios *b* e *u*, *1< b, u < n*, calcula sua chave pública *B* (Equação 3-4) e envia para *Alice* juntamente com o parâmetro *u*. Neste caso, *u* é um valor público e aleatório. Somente *Bob* pode gerar a chave pública *B*, pois o número aleatório *b* é secreto e só é conhecido por *Bob*.



$$B \equiv (v + g^{b}) \; mod \; n \qquad (3\text{-}4)$$

- **Passo 5**

Alice e *Bob* calculam o valor de *S* (Equações 3-5 e 3-6) usando os valores que cada um possui. O valor de *S* representa a chave de sessão.

$$Alice : S \equiv (B - g^{x})^{a + ux} \; mod \; n \qquad (3\text{-}5)$$

$$Bob : \; S \equiv (Av^{u})^{b} \; mod \; n \qquad (3\text{-}6)$$

- **Passo 6**

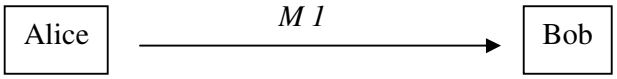
Ambos aplicam uma função *hash* (função unidirecional), denominada ***H,*** a chave de sessão *S*.

$$Alice: K = H(S)$$

$$Bob: K = H(S)$$

- **Passo 7**

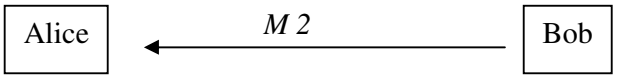
Alice envia uma mensagem *M1* (Equação 3-7) para *Bob* como evidência de que ela possui a chave de sessão correta. *Bob* calcula a mensagem *M1* (Equação 3-7) e verifica se combina com o que Alice enviou.



$$M\ 1 = H(A,B,K) \qquad (3\text{-}7)$$

- **Passo 8**

Bob envia uma mensagem *M2* para *Alice* como evidência de que ele também possui a chave de sessão correta. *Alice* verifica a mensagem *M2* (Equação 3-8).



$$M2 = H(A,M1,K) \qquad (3\text{-}8)$$

A Figura 5 apresenta os passos do protocolo *SRP-3*.

Alice		**Bob**
1.	c →	
2. x = H(s,P)	← s	(verifica s,v)
3. $A \equiv g^a \bmod n$	A →	
4.	← B,u	$B \equiv (v + g^b) \bmod n$.
5. $S \equiv (B - g^x)^{a + ux} \bmod n$		$S \equiv (Av^u)^b \bmod n$
6. K = H(S)		K = H(S)
7. M1 = H(A,B,K)	M1 →	(verifica M1)
8. (verifica M2)	← M2	M2 = H(A,M1,K)

Figura 5: Passos do protocolo *SRP-3*

Os cálculos de *S* (Equação 3-5, 3-6) são feitos de forma diferente por *Alice* e *Bob*. Entretanto, conforme é a seguir, eles coincidem.

$$Alice: \; S \equiv (B - g^{x})^{a+ux} \; mod\, n$$

$$S \equiv (v + g^{b} - g^{x})^{a+ux} \; mod\, n$$

$$S \equiv (g^{x} + g^{b} - g^{x})^{a+ux} \; mod\, n$$

$$S \equiv g^{ba+uxb} \; mod\, n$$

$$Bob: \quad S \equiv (Av^{u})^{b} \; mod\, n$$

$$S \equiv (g^{a}v^{u})^{b} \; mod\, n$$

$$S \equiv (g^{a}g^{xu})^{b} \; mod\, n$$

$$S \equiv g^{ab+xub} \; mod\, n$$

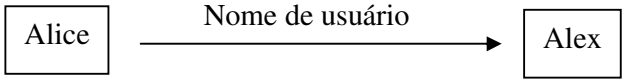
O cálculo da chave pública *B*

Observe que no passo 4, *Bob* calcula a chave pública *B* (Equação 3-4) , como a soma ($v + g^{b}$) *mod n*. Por que a chave pública *B* não é apenas calculada como $g^{b} mod\ n$, simplificando assim o protocolo?

Infelizmente, esta simplificação habilita o intruso a realizar ataques, fazendo com que este se passe por um usuário e tenha a sua autenticação aceita. Por exemplo, suponha que um intruso, *Alex*, consiga capturar o valor de *s* de uma sessão autêntica e realiza os seguintes passos:

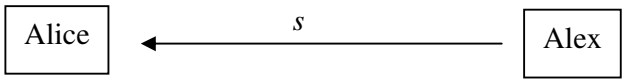
- Passo 1

  Alice envia para o intruso *Alex* seu nome de usuário.



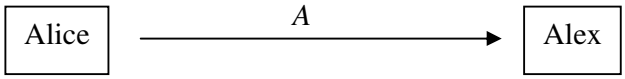
- Passo 2

O intruso *Alex* envia para *Alice* o valor de *s* capturado anteriormente.



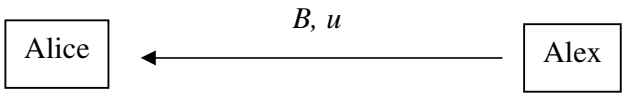
- Passo 3

Alice envia para o intruso *Alex* sua chave pública *A*.



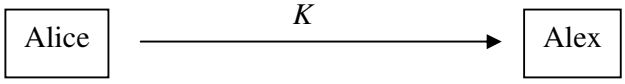
- Passo 4

O intruso *Alex* escolhe seus números aleatórios *b* e *u*, calcula a sua chave pública *B* e envia para *Alice*: *B* e *u*. Neste caso, $B \equiv g^b \ mod \ n$



- Passo 5

Neste caso, *Alice* calcula sua chave de sessão como sendo: $S \equiv B^{a+ux} mod \ n$. Em seguida, calcula *K* com base em *S* e envia para o intruso *Alex* o valor *K*, como prova de que a chave de sessão *K* é autêntica.



- Passo 6

O intruso *Alex* simula uma falha na rede ou simplesmente informa a *Alice* que a senha não está correta.

Neste caso, *Alice* e *Alex* também calculam o mesmo valor de *S* (Equação 3-9, 3-10).

$$Alice: \ S \equiv B^{a+ux} \ mod \ n \qquad (3\text{-}9)$$

$$S \equiv g^{b(a+ux)} \ mod \ n$$

$$S \equiv g^{ba+bux} \bmod n$$

$$Alex: \quad S \equiv (Av^u)^b \bmod n \qquad (3\text{-}10)$$

$$S \equiv (g^a v^u)^b \bmod n$$

$$S \equiv (g^a g^{au})^b \bmod n$$

$$S \equiv (g^a g^{au})^b \bmod n$$

$$S \equiv g^{ab+bxu} \bmod n$$

Deste modo, o intruso *Alex* possui a chave pública *A* ($A \equiv g^a \bmod n$), gerada por *Alice* e o valor de *b*, que prova que o valor *K* foi calculado por *Alice*.

O intruso *Alex* pode tentar descobrir por força bruta um valor da senha, *P*. Para isto basta supor um valor *P'* e calcular:

$$x' = H(s,P')$$

$$v' \equiv g^{x'} \bmod n$$

$$S' \equiv (Av'^u)^b \bmod n$$

$$K' = H(S')$$

Alex tenta vários valores de *P'* até obter *K'* = *K*. Quando a igualdade é obtida, tem-se a senha procurada.

A Regra do valor *u*

Por que *Bob* precisa enviar *u* e *B* para *Alice* e não apenas *B*, deixando *Alice* escolher *u*?

Razão: Um intruso que tenha capturado o valor secreto *v* pode conseguir acesso ao servidor, caso a chave de sessão *S* (Equação 3-11) seja criada apenas em função de *a* e *b*. Neste caso, suponha:

$$S \equiv g^{ab} \bmod n \qquad (3\text{-}11)$$

Desta forma, *S* não depende da chave secreta *x* mas apenas de *a*, conhecido por *Alice* e de *b*, conhecido por *Bob*. Além disso, *Bob* sabe que $A \equiv g^a \bmod n$. Relembrando o

cálculo da chave de sessão *S* e da chave pública *B,* descrito nos passos 4 e 5 do protocolo *SRP*, tem-se:

$$S \equiv (B - g^{x})^{a + ux} \bmod n$$

e

$$B \equiv (v + g^{b}) \bmod n$$

Suponha que o intruso calcule $A \equiv g^{a}v^{-u} \bmod n$ no lugar de $A \equiv g^{a} \bmod n$ e envie o resultado para *Bob* para fazer o novo cálculo $A \equiv g^{a}v^{-u} \bmod n$ o intruso utiliza o valor público $g^{a} \bmod n$, seu conhecimento de v e supõe um número aleatório u. O intruso calcula S da seguinte forma:

$$S \equiv (B - g^{x})^{a} \bmod n = g^{ab} \bmod n$$

Observe que $v \equiv g^{x} \bmod n$ é conhecido pelo intruso.

Bob calcula *S* da seguinte forma:

$$S \equiv (Av^{u})^{b} \bmod n \equiv g^{ab} \bmod n$$

Observe que neste caso, u é escolhido pelo intruso e enviado para *Bob*.

Deste modo, o intruso tenta convencer *Bob* se passando por *Alice*. Se o intruso consegue capturar a chave de sessão *K*, este pode tentar por força bruta descobrir a senha do usuário. Mas descobrir a senha do usuário a partir da chave de sessão *K* é computacionalmente inviável, pois $K = H(S)$ e H é uma função *hash*, esta função calcula o valor de *K* independente de chave secreta. Sendo assim, não é fácil por força bruta descobrir a senha do usuário a partir do valor da chave de sessão *K*.

Portanto, se o valor secreto de x é conhecido, o intruso pode se passar por *Alice* e caso o intruso conheça o valor secreto v, este pode se passar por *Bob*.

3.3 Protocolo *SRP-6*

Esta seção apresenta o protocolo *SRP-6*. Esta versão do protocolo *SRP* agrega dois mecanismos de segurança, o primeiro previne o ataque "*two-for-one*" onde o intruso usa duas supostas senhas para se passar por um servidor e o segundo mecanismo de segurança cria um pedido de mensagem (*Message Ordering*) no qual o servidor espera que o usuário envie o valor público *A,* antes de enviar o valor pseudo-aleatório *u*.

O protocolo *SRP-6* é uma extensão do *SRP-3* e visa eliminar algumas de suas deficiências. A primeira deficiência é remover a simetria da equação: $g^x+g^b \ mod \ n$**,** onde $g^x = v$. Isto é obtido multiplicando v por algum valor k, por exemplo: $B = kv + g^b$. Mas esta mudança não protege o servidor (Bob) do ataque *Two-for-One*, no caso em que o intruso conheçe k e j tal que, $k = g^j$, ele pode enviar B ao usuário como $B= kg^x + kg^y$. Se a senha atual é x, o usuário desconsidera kg^x e usa kg^y como base para calcular a chave de sessão.

Conforme [WU 2002] um valor seguro é $k = 3$. Usando $k = 3$**,** assegura-se também que qualquer conjunto de parâmetros n e g que são seguros para serem usados no protocolo *SRP* são também seguros para $k = 3$.

Outra melhoria do protocolo *SRP-6* em relação ao *SRP-3* é a sequenciamento dos passos 3 e 4. No passo 4 do protocolo *SRP-3,* a chave pública B (Equação 3-4) é calculada pelo servidor e enviada juntamente com o parâmetro pseudo-aleatório u. O parâmetro u não deve ser enviado ao cliente antes que este envie a sua chave pública A (Equação 3-3) **.** Enviar o parâmetro u para o usuário antes que este envie a sua chave pública A para o servidor**,** permite a um intruso que conheça o verificador v, mas não conheça x, se passar por um usuário e calcular a chave pública $A \equiv g^a v^{-u} \ mod \ n$ ao invés de calcular $A \equiv g^a \ mod \ n$, com esta nova chave pública A**,** o intruso irá cancelar o termo v^{-u} no cálculo da chave de sessão do servidor, como demonstrado a seguir:

$$S = (Av^{-u})^{b} = (g^{a}v^{-u}v^{u})^{b} = (g^{a})^{b}$$

Portanto, é necessário enviar *u* após receber a chave pública *A*. Deste modo o intruso não está habilitado para calcular a mensagem *M1*, sem antes conhecer o parâmetro *u*. A mensagem *M1* é calculada no passo 7 como indicado a seguir:

$$M1 = H(A,B,K)$$

Onde:

$$K = H(S)$$

e

$$S \equiv (B - g^{x})^{a + ux} \bmod n$$

A chave de sessão *K* depende dentre outros valores, do parâmetro pseudo-aleatório *u*, deste modo o intruso deve conhecer o valor de *u* para calcular a chave de sessão *K*. Uma solução para tais restrições é considerar o valor *u* não apenas um parâmetro pseudo-aleatório e sim, uma função unidirecional *hash,* aplicada às chaves públicas *A* e *B*, como apresentado a seguir:

$$u = H(A,B)$$

Para o intruso descobrir o valor de *u*, primeiramente ele deve calcular $A = g^{a}v$, já que a chave pública *B* é conhecida. Sendo assim, *u* é calculado como:

$$u = H(g^{a}v^{-u},B)$$

A Figura 6 apresenta os passos do protocolo *SRP-6*, como uma nova versão do protocolo *SRP-3*, sua interpretação é análoga a da Figura 5.

Alice		**Bob**
1.	I →	
2. x = H(s, I, P)	← S	(verifica s,v)
3. $A \equiv g^{a} \bmod n$	A →	
4. u = H (A,B)	← B	$B \equiv (3v + g^{b}) \bmod n$. u = H (A,B)
5. $S \equiv (B - 3g^{x})^{a + ux} \bmod n$		$S \equiv (Av^{u})^{b} \bmod n$
6. M1 = H(A,B,S)	M1 →	(verifica M1)
7. (verifica M2)	← M2	M2 = H(A,M1,S)
8. K = H(S)		K = H(S)

Figura 6: Passos do protocolo *SRP-6*

3.4 Análise da segurança do protocolo *SRP*

Uma análise da segurança do protocolo *SRP-6* é considerada em [WU 2002], onde se conclui que:

- Nenhuma informação sobre a senha ou a chave privada *x* é revelada.
- Nenhuma informação útil sobre a chave de seção *K* é revelada enquanto o protocolo é executado.
- Se o arquivo de senhas do servidor é capturado e o intruso captura o valor *v*, isso ainda não o permite passar por um usuário sem uma extensa pesquisa. Isto significa que não é possível executar o ataque do meio.
- Se alguma chave de sessão *K*, de sessões anteriores, for comprometida, não é possível que um intruso deduza o valor da senha de um usuário. Se o valor da senha é comprometido, não é possível ao intruso determinar a chave de sessão *K* de sessões anteriores e decifrá-las, pois a estrutura matemática do protocolo de segurança remota de senhas (*SRP)* é similar ao problema de *Diffie_Helman* [STALLINGS 1999], que é computacionalmente inviável em grupos onde o logaritmo discreto é intratável.
- $g^x \ mod \ n$ é uma exponenciação discreta e sua inversa é um logaritmo discreto. Encontrar logaritmo discreto é um problema computacionalmente difícil para grandes valores de *n*.

4 PROTOCOLO ***CRIPTO-SRP***

Neste capítulo é proposto um protocolo de autenticação de senhas em ambiente cifrado para acesso a base de dados, denominado ***Cripto-SRP*** (*Criptography Secure Remote Password*). Este protocolo utiliza conceitos apresentados no protocolo *SRP* e uma base de dados cifrada, onde as consultas são efetuadas sem que os dados sejam decifrados. Isto garante a segurança aos dados armazenados no servidor de banco de dados. As seções deste capítulo são as seguintes: a seção 4.1 apresenta e as funcionalidades do ***Cripto-SRP***. A seção 4.2 apresenta os fundamentos de criptografia utilizados pelo protocolo ***Cripto-SRP*** para transmitir, verificar e armazenar dados de *logins* (Senha e Nome do usuário) de forma criptografada. A seção 4.3 descreve o protocolo ***Cripto-SRP*** em um ambiente de banco de dados cifrado. Já uma análise computacional do protocolo ***Cripto-SRP*** é apresentada na seção 4.4. Na seção 4.5 são apresentadas as vantagens e desvantagens do protocolo ***Cripto-SRP***. E por fim é apresentado na seção 4.6 um resumo deste capítulo.

4.1 Funcionalidades do ***Cripto-SRP*** (Criptography Security Remote Password)

O protocolo ***Cripto-SRP*** foi desenvolvido para autenticação de senhas em banco de dados cifrado. A proposta é uma extensão do protocolo *SRP*, ela atende a todas as funcionalidades de segurança contempladas pelo protocolo *SRP*. Além das funcionalidades do protocolo *SRP*, também atende as funcionalidades a seguir:

- Fornecer confidencialidade a senha do usuário durante o processo de autenticação;
- Assegurar a integridade da informação transmitida;
- Fornecer autenticação ao usuário que apresenta senha secreta;

- Assegurar o uso de práticas de segurança, a fim de proteger todas as partes envolvidas em uma autenticação de senha;
- Manter as informações de *logins* (Senha e Nome) que estão sendo armazenadas sempre cifradas;

  Os dados devem sempre permanecer cifrados, tanto os dados que estão sendo transmitidos quanto os que estão armazenados no servidor de banco de dados (***SBD***), aumentando assim o nível de segurança em cada autenticação.
- Fornecer uma função de verificação definida como Ch_{User_name};

  O nome do usuário é criptografado utilizando chaves secretas e transmitido como chave pública. A função Ch_{User_name} realiza cálculos matemáticos utilizando os valores públicos transmitidos e chaves secretas, resultando em um valor de verificação para que o servidor de banco de dados possa identificar o nome do usuário no banco de dados cifrado. Esta funcionalidade é proposta pelo protocolo *SRP*, mas não com a segurança aqui implementada. A função Ch_{User_name} do protocolo proposto utiliza exponenciais, operações modulares e números aleatórios secretos para aumentar a segurança ao verificar o nome do usuário no banco de dados cifrado.
- Fornecer uma função de autenticação definida como $Ch_{password}$;

  A senha do usuário não é transmitida durante a execução do protocolo. O que ocorre é uma consulta ao verificador da senha, e este processo é realizado com o uso da função $Ch_{password}$. O protocolo *SRP* também implementa esta funcionalidade, mas não utiliza os cálculos matemáticos aqui propostos. A função $Ch_{password}$ nesta proposta verifica a senha do usuário, utilizando como parâmetro de entrada o nome cifrado e uma chave pública gerada a partir de um número aleatório secreto. Em seguida são feitos cálculos matemáticos utilizando exponenciais e operações modulares com chaves secretas,

aumentando assim a segurança ao verificar a senha do usuário em um banco de dados cifrado.

- Fornecer uma extensão da linguagem *SQL*(*Structured Query Language*);

  Os dados são manipulados no servidor de banco de dados utilizando uma extensão da linguagem *SQL*.

Este protocolo utiliza a mesma estrutura do protocolo *SRP* descrito no capítulo 3. Porém, este novo protocolo agrega novas funcionalidades de segurança as já existentes no protocolo *SRP*, que integrado ao ambiente de banco de dados cifrado, descrito mais adiante na seção 4.2, é um protocolo para autenticação de senhas.

Os participantes do ***Cripto-SRP*** são:

- Usuário;
- Servidor de banco de dados;
- Ambiente de banco de dados criptografado.

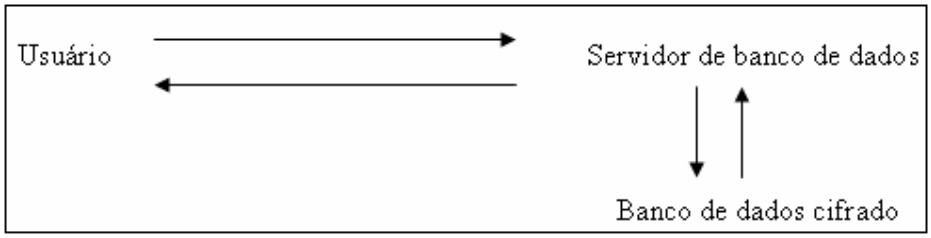


Figura 7: Participantes do *Cripto-SRP*.

4.2 Fundamentos criptográficos necessários para utilizar o ***Cripto-SRP***.

Esta seção apresenta os fundamentos criptográficos necessários para a execução do protocolo ***Cripto-SRP***, a ser descrito na seção 4.3**.**

Consideram-se a seguir os fundamentos de criptografia necessários à definição do protocolo ***Cripto-SRP***.

Na definição deste protocolo são utilizadas técnicas de criptografia assimétrica, em especial o criptossistema ElGamal. Portanto, é necessária a definição de estruturas auxiliares, que não são utilizadas, por exemplo, no protocolo *SRP*. Tais estruturas são descritas da seguinte forma:

- **Definição de um grupo finito $\mathbb{Z}_P^*$ e do gerador *g*.**

As partes envolvidas no protocolo, usuário e servidor de banco de dados, devem definir um grupo finito $\mathbb{Z}_p$, sendo p um número primo grande tal que o problema do logaritmo discreto seja intratável em $\mathbb{Z}_p$ [GORDON 1993]. Este grupo é utilizado nas autenticações do ***Cripto-SRP***. Além de $\mathbb{Z}_p$, as partes devem definir um gerador g de $\mathbb{Z}_p$. Os parâmentos g e p são públicos e devem ser disponibilizados ao servidor de banco de dados e ao usuário que está se autenticando. Eles serão utilizados para criptografar os dados dos *logins* contidos nas autenticaçãoes.

- **Geração de números aleatórios em $\mathbb{Z}_{p-1}^*$.**

As partes envolvidas no protocolo ***Cripto-SRP***, usuário e servidor de banco de dados devem ser capazes de gerar números aleatórios secretos pertencentes a $\mathbb{Z}_{p-1}^*$. Estes números são utilizados na criptografia dos dados contidos durante as autenticações.

Representando e armazenando *logins*.

Os dados dos *logins* formados por nome e senha, são em parte, armazenados na forma criptografada, utilizando o grupo finito público $\mathbb{Z}_p$. Seja I uma instância de relação sobre um esquema $(Q_1...,Q_v)$ [ELMASRI 2000]. Na primeira linha de *I*, os nomes dos

atributos definem o esquema do banco de dados. Estes nomes são representados por elementos de $\mathbb{Z}_p$, ou seja, são representados por números que pertencem ao conjunto:

- $\{0, 1, ..., p-1\}$

A primeira linha em I é uma lista e tem a forma $[q_1,...,q_v]$ onde

- $q_i \in \mathbb{Z}_p,\ 1 \leq i \leq v$

e q_i representa o nome do atributo Q_i. Os elementos de uma tupla na tabela I são também representados por elementos de $\mathbb{Z}_p$. Seja T_j a j-ézima tupla em I**.** A tupla T_j é uma lista:

- $T_j = [\ m_{j1},...,m_{ji}\ ]$

onde para $1 \leq j \leq t,\ 1 \leq i \leq v$, $m_{ji} \in \mathbb{Z}_p$.

A tabela I tem a forma:

$\mathbf{q_1}$	**.....**	$\mathbf{q_v}$
m_{11}	……..	m_{1v}
....	.....	....
m_{t1}	.....	m_{tv}

Tabela 3: Atributos de banco de dados

Para cada tabela I, a sua versão cifrada, é denotada por EI**.**

A tabela cifrada EI *tem* a forma:

a_1	.....	a_v
x_{11}	.....	x_{1v}
....	.....	....
x_{t1}	.....	x_{tv}

Tabela 4: Atributos de banco de dados criptografados

onde para $1 \leq j \leq t$, $1 \leq i \leq v$, $x_{ij}, a_i \in \mathbb{Z}_p$

No protocolo proposto, é criada a tabela denominada *Tab_Verifier* que possui respectivamente os atributos *user_name*, *parameter* e *verifier*.

A versão cifrada da tabela *Tab_Verifier* é a tabela *Cripto_Tab_Verifier* cujos dados, são criptografados utilizando os parâmetros públicos *g* e *p*, tal que *g* é um gerador do grupo $\mathbb{Z}_p^*$.

Neste ambiente, assume-se uma extensão da linguagem *SQL* (*Structure Query Language*), onde o usuário pode indicar qual(is) atributo(s) terá o seu valor criptografado. O comando a seguir cria a tabela *Cripto_Tab_Verifier* :

CREATE TABLE *Cripto_Tab_Verifier*

(user_name integer ***ENCVALUEATRIB****,*

parameter integer, verifier integer, ***Constraint*** *pk_Cripto_Tab_Verifier* ***primary key****(parameter, verifier));*

O comando ***ENCVALUEATRIB*** (n_{i1}) representa o cálculo:

$$g^{n_{i1}} \ (mod\ p)$$

O valor de n_{i1} é a i-ésima instância do atributo *user_name*.

Um exemplo do comando para inserção de uma instância na tabela *Cripto_Tab_Verifier* é:

Insert into *Cripto_Tab_Verifier*

Values $(g^{user\_name} \pmod p,\ s,\ v))$;

Onde p é um número primo $\in \mathbb{Z}_p$.

Os valores são armazenados na tabela *Tab_Verifier*, como indicado a seguir.

User_name	Parameter	Verifier
n_{11}	s_{12}	v_{13}
....	....	....
n_{t1}	s_{t2}	v_{t3}

Tabela 5: Tab_Verifier

Os valores n_{i1} representam os nomes de usuários na tabela *Tab_Verifier*. Já os valores s_{i2} e v_{i3} são inteiros cuja utilização será descrita mais adiante na definição do protocolo. Os valores s_{i2} e v_{i3} são determinados como no protocolo *SRP*, $x = H(s_{i2},P)$ e $v_{i3} \equiv g^x \bmod p$, onde P é a senha do usuário (***User***).

O protocolo proposto utiliza a versão criptografada de *Tab_Verifier (Cripto_Tab_Verifier)* para efetuar as autenticações dos usuários (***User***).

User_Name	Parameter	Verifier
x_{11}	s_{12}	v_{13}
....	....	....
x_{t1}	s_{t2}	v_{t3}

Tabela 6: Cripto_Tab_Verifier

Os elementos de *Cripto_Tab_Verifier* são calculados como se segue:

- $x_{i1} \equiv g^{n_{i1}} \pmod p$.

A seção a seguir utiliza este ambiente criptografado para executar os passos do protocolo proposto.

4.3 Protocolo ***Cripto-SRP*** em ambiente cifrado

O protocolo ***Cripto-SRP*** é definido em um ambiente de transmissão e verificação de senhas que possuí usuários (***User)*** e servidores de banco de dados (***SBD***) que são capazes de

executar cálculos matemáticos, como exponenciais e operações modulares. As operações de verificação são executadas sem a necessidade de decifrar os dados armazenados na memória secundária do servidor de banco de dados (***SBD***). O usuário ao se cadastrar ***(User)*** envia sua senha, seu nome e o servidor de banco de dados (***SBD***) armazena o nome na forma cifrada e o verificador da senha. A senha e nome de usuários são definidos respectivamente por: *P e user_name*.

Para autenticar senhas e armazenar *logins* em um ambiente cifrado, é proposta uma extensão do protocolo *SRP* [WU 2002] o protocolo ***Cripto-SRP***. Nesta proposta, cada servidor de banco de dados armazena a tabela *Cripto_Tab_Verifier* dada por:

User_Name	Parameter	Verifier
x_{11}	s_{12}	v_{13}
....	....	....
x_{t1}	s_{t2}	v_{tv}

Tabela 7: Cripto_Tab_Verifier

A tabela *Cripto_Tab_Verifier* armazena os elementos necessários ao desenvolvimento do protocolo ***Cripto-SRP***. Observe que tal tabela armazena também dados públicos s_{i2} e v_{i3}. Os passos a seguir definem o protocolo ***Cripto-SRP*** e são descritos a seguir:

Figura 8: Autenticação de usuário, utilizando o protocolo *Cripto-SRP*.

Passo 1. User –> SBD $\{y_1 , y_2\}$

Este passo inicia o processo de autenticação do usuário ao servidor de banco de dados. É válido lembrar que o dados dos *logins* (Senha e Nome do usuário) devem estar cadastrados na tabela *Cripto_Tab_Verifier* no servidor de banco de dados. O usuário envia para o servidor de banco de dados o conjunto de dados $\{y_1, y_2\}$. Este conjunto é calculado pelo usuário (***User***) da seguinte forma: Inicialmente o usuário determina dois números aleatórios secretos k_0 e k_1. Em seguida efetua os cálculos, como demonstra a equação a seguir:

$$a_0 \equiv g^{k_0} \bmod p \qquad (4\text{-}1)$$

$$y_1 \equiv a_0^{k_1} \bmod p$$

$$y_2 \equiv g^{user\_name + k_1} \bmod p$$

Neste caso a_0 (Equação 4-1) , k_0 e k_1 são chaves privadas do usuário (***User***) e *user_name* é o nome do usuário a ser verificado pelo servidor de banco de dados (***SBD***). Nesta fase, o usuário ainda não prova que conhece sua senha secreta *P*. Neste passo, é feita apenas uma verificação do nome do usuário no banco de dados do servidor de banco de dados (***SBD***).

É necessário utilizar números aleatórios k_0 e k_1 diferentes para cada vez que este passo do protocolo é executado. Caso tais números coincidam existe um tipo de ataque que pode revelar as chaves secretas [STINSON 1995]. Além disso, a utilização de números aleatórios diferentes inibe ataques do tipo repetição. Isto ocorre porque os valores y_1 e y_2 são diferentes cada vez que este passo do protocolo é executado.

Passo 2. SBD –> User *(H, s)*

Ao receber $\{y_1, y_2\}$, o servidor de banco de dados ***(SBD)*** verifica os dados y_1 e y_2 no banco de dados de *logins* e notifica o usuário. Caso o usuário não exista no banco de dados de *logins*, a autenticação é cancelada e nenhum passo a seguir é executado.

- Inicialmente, o servidor de banco de dados ***(SBD)*** calcula $Ch_{user\_name}(y_1, y_2)$ (Equação 4-2). A equação a seguir apresenta estes cálculos:

$$Ch_{user\_name}(y_1, y_2) = y_2(y_1^{k_0^{-1}})^{-1} \bmod p \equiv y_2(a_0^{k_1 k_0^{-1}})^{-1} \bmod p \qquad (4\text{-}2)$$

$$\equiv g^{user\_name + k_1}(g^{k_1 k_0^{-1}})^{-1} \bmod p$$

$$\equiv g^{user\_name + k_1}(g^{k_0 k_1 k_0^{-1}})^{-1} \bmod p$$

$$\equiv g^{user\_name + k_1}(g^{-k_1}) \bmod p$$

$$\equiv g^{user\_name} \bmod p$$

$Ch_{user\_name}(y_1, y_2)$ (Equação 4-2) é uma função de verificação do nome do usuário. Esta função não é utilizada pelo protocolo *SRP*. Em seguida, o servidor de banco de dados ***(SBD)*** executa o comando:

SELECT * FROM ***Cripto_Tab_Verifier***
WHERE ((User_Name) = $Ch_{user\_name}(y_1, y_2)$);

Caso este comando não retorne nenhuma tupla, significa que não existe o usuário $g^{user\_name} \bmod p$ armazenado na tabela *Cripto_Tab_Verifier* armazenada no servidor de banco de dados ***(SBD).*** Neste caso, é cancelada a transação não executando os passos seguintes. O servidor de banco de dados ***(SBD)***, após verificar o nome do usuário (*user_name*) na tabela *Cripto_Tab_Verifier* identifica a tupla $((g^{user\_name} \bmod p), s, v)$.

Em seguida, o servidor de banco de dados ***(SBD)*** envia para o usuário (***User***) o número *s* e qual função *Hash* [STALLINGS 1999] deve ser utilizada nos próximos passos. Observe que nos próximos passos o usuário (***User***) e o servidor de banco de dados ***(SBD)*** conhecem a relação entre *s* e *v* que é dada por:

$v \equiv g^x \bmod p$ *;* onde $x = H(s,P)$ e *P* é a senha secreta do usuário. Observe também que não é possível descobrir *P* a partir de *s* e *v*.

Passo 3. User - > SBD *(Pedido de autenticação do usuário, Enc(user_name),* a_2 *).*

Neste passo o usuário (***User***) gera um número aleatório secreto k_2 e calcula.

$$a_2 \equiv g^{k_2} \bmod p$$

$$Enc(user\_name) \equiv (user\_name - v\,a_2)\,k_2^{-1}\ (mod(p-1)) \qquad (4\text{-}3)$$

Para criptografar o conteúdo do atributo *user_name* o usuário calcula *Enc(user_name)* (Equação 4-3), o usuário deve conhecer *v*. Mas para calcular *v* o usuário (***User***) tem que conhecer *P* (senha secreta) e *s*, pois $v \equiv g^x \bmod p$ e $x = H(s,P)$. O número *s* foi enviado para o usuário (***User***) no passo anterior e *P* é a sua senha secreta. O usuário (***User***) envia ao servidor de banco de dados ***(SBD)*** o pedido de autenticação do usuário, o nome do usuário criptografado (*Enc(user_name)*) e a chave pública a_2 .

Passo 4. SBD -> User (*Resposta ao pedido de autenticação do usuário*)

O servidor de banco de dados ***(SBD)*** ao receber: *(Pedido de autenticação, Enc(user_name) e* a_2), faz os seguintes cálculos:

$$Ch_{password}(Enc(_{user\_name}), a_2) \equiv g^{va_2}(a_2)^{Enc(user\_name)} \bmod p \qquad (4\text{-}4)$$

$$\equiv g^{va_2}(g^{k_2})^{(user\_name - va_2)k_2^{-1}} \bmod p$$

$$\equiv g^{va_2} g^{user\_name - va_2} \bmod p$$

$$\equiv g^{user\_name} \bmod p$$

Observe que no cálculo acima o servidor de banco de dados ***(SBD)*** utiliza o valor *v* da tupla <(($g^{user\_name} \bmod p$),*s*,*v*> obtida no passo 2. No caso em que o resultado da função $Ch_{password}(Enc(_{user\_name}), a_2)$ (Equação 4-4) pertence à tupla que contém *s*, valor obtido no passo 2, o servidor de banco de dados ***(SBD)*** aceita a autenticação como um sucesso.

Tendo ocorrido sucesso na autenticação, os passos a seguir são executados. Eles estabelecem uma chave de sessão entre o usuário (***User***) e o servidor de banco de dados ***(SBD)***.

Passo 5. SBD - > User a_3

Neste passo, o servidor de banco de dados ***(SBD)*** gera um número aleatório secreto k_3 e envia ao usuário (***User***) a chave pública a_3 calculada como mostra a equação a seguir:

$$a_3 \equiv (3v + g^{k_3}) \bmod p \qquad (4\text{-}5)$$

Além de a_3, o servidor de banco de dados ***(SBD)*** calcula:

$$u = H(a_0, a_3)$$

$$S \equiv (a_0 v^u)^{k_3} \bmod p$$

$$K = H(S)$$

Passo 6. User - > SBD (M_1)

Tendo recebido a_3 (Equação 4-5) o usuário (***User***) calcula *u* (Equação 4-6) , *S* (Equação 4-7) e a mensagem M_1 (Equação 4-8) como demonstra as equações seguir:

$$u = H(a_0, a_3) \qquad (4\text{-}6)$$

$$S \equiv (a_3 - 3g^x)^{k_0 + ux} \bmod p \qquad (4\text{-}7)$$

$$M_1 = H(a_0, a_3, S) \qquad (4\text{-}8)$$

O valor de M_1 é enviado para o servidor de banco de dados ***(SBD)*** e a chave de sessão é dada por: S

Passo 7. SBD - > User *(M_2)*

Tendo recebido M_1, o servidor de banco de dados ***(SBD)*** verifica a autenticação deste valor, calculando:

$$\mathrm{H}(a_0, a_3, S)$$

Observe que o servidor de banco de dados ***(SBD)*** conhece a_0, a_3 e S. Se a verificação ocorre com sucesso, então o servidor de banco de dados ***(SBD)*** calcula o valor M_2 , como demonstra a equação a seguir e o envia para o usuário:

$$M_2 = H(a_0, M_1, S) \qquad (4\text{-}9)$$

Passo 8.User *(Usuário verifica o valor de M_2)*

Neste passo o usuário (***User***) verifica o valor de M_2 (Equação 4-9) calculando:

$$\mathrm{H}(a_0, M_1, S)$$

Observe que o usuário (***User***) conhece os valores a_0, M_1 e S. Tendo sido verificado o valor de M_2 (Equação 4-9), a chave de sessão K é estabelecida para a utilização do usuário (***User***) e do servidor de banco de dados ***(SBD)***.

Deste modo a fase de associação é iniciada com sucesso. A chave de sessão K gerada nos passos 5 e 6 é distribuída a cada processo de associação e a comunicação é feita utilizando tal chave. A chave K é calculada a partir de S, e o valor S é calculado de forma diferente nos passos 5 e 6. Conforme o passo 5, tem-se:

$$S \equiv (a_0 v^u)^{k_3} \bmod p$$

e conforme o passo 6 tem-se:

$$S \equiv (a_3 - 3g^{x})^{k_0 + ux} \bmod p$$

Mesmo tendo sido calculado de forma diferente, tais valores coincidem. A equação a seguir demonstra os cálculos tanto da chave de sessão do usuário quanto do servidor de banco de dados:

$$S \equiv (a_3 - 3g^{x})^{k_0 + ux} \bmod p \equiv (3v + g^{k_3} - 3g^{x})^{k_0 + ux} \bmod p$$
$$\equiv (3g^{x} + g^{k_3} - 3g^{x})^{k_0 + ux} \bmod p \equiv g^{k_3 (k_0 + ux) \bmod p}$$

Por outro lado:

$$S \equiv (a_0 v^{u})^{k_3} \bmod p \equiv (g^{k_0} . g^{xu})^{k_3} \bmod p \equiv g^{(k_0 + ux) k_3} \bmod p$$

4.4 Análise Computacional do protocolo ***Cripto-SRP***

O protocolo ***Cripto-SRP*** utiliza chaves secretas e operações com exponenciais para aumentar a segurança no processo de autenticação. Por outro lado estas operações necessitam de um esforço computacional que deve ser avaliado. Esta seção apresenta uma análise do custo computacional do protocolo ***Cripto-SRP*** em termos do número de operações no grupo finito $\mathbb{Z}_p$.

Assume-se que p é um número primo com *len(p)* (*len* é um acrônimo para "*length*"). Para avaliar o custo computacional do ***Cripto-SRP*** é necessário calcular o tempo estimado para executar todas as operações com exponenciais em um grupo finito $\mathbb{Z}_p$. Em outras palavras, dado $g \in \mathbb{Z}_p$ e um número inteiro *x*, deve-se calcular $g^x \bmod p$**.** Uma maneira de realizar tal cálculo é multiplicar *g* por si mesmo *x* vezes, cujo custo computacional é $O(len(x)\ log\ (len(p))^2)$ [SHOUP 2005].

Nos 4 passos da fase de autenticação do protocolo ***Cripto-SRP*** são feitas 19 exponenciações com números inteiros em $\mathbb{Z}_p$. Logo a complexidade do algoritmo é $O(len(p)^3)$ que é igual a complexidade do protocolo *SRP*.

Observe que nesta análise as adições e multiplicações são desconsideradas no cálculo da complexidade, dado que tais operações têm custo um menor que a exponenciação.

O algoritmo ***Cripto-SRP*** tem uma complexidade da mesma ordem do *SRP*. Entretanto, como são necessárias 13 exponenciações para autenticar um usuário no ***Cripto-SRP***, e apenas 2 no *SRP*, o protocolo proposto tem um custo computacional quase 7 vezes maior quando se considera apenas as exponenciações.

4.5 Vantagens e desvantagens do protocolo ***Cripto-SRP***

Esta seção apresenta algumas vantagens e desvantagens do protocolo proposto.

Vantagens:

- O nome do usuário é manipulado na forma cifrada. No passo 1 ao iniciar a fase de autenticação, é gerada uma chave pública $y_2 \equiv g^{user\_name+k_1} \bmod p$, ao invés de utilizar o nome do usuário de forma original. No passo 2 o servidor de banco de dados (***SBD***) verifica o nome do usuário utilizando a função $Ch_{user\_name}(y_1, y_2)$. Já no passo 3 o nome do usuário é cifrado utilizando o cálculo *Enc(user_name)*. E por fim, no passo 4 o servidor de banco de dados autentica o usuário utilizando a função $Ch_{password}(Enc(_{user\_name}), a_2)$.
- É utilizado um modelo de banco de dados cifrado. O protocolo proposto utiliza a tabela *Cripto_Tab_Verifier* para verificar os dados do usuário. Ao criar esta tabela é utilizado o comando ***ENCVALUEATRIB*** que cifra o nome do usuário toda vez que o mesmo é cadastrado.

- Verifica o nome e a senha separadamente em uma base de dados cifrada utilizando Ch_{User_Name} e $Ch_{Password}$. Observe que no *SRP* tais funcionalidades não existem.
- Mantém a mesma ordem de complexidade do protocolo *SRP*.

Desvantagens

- Requer um ambiente capaz de efetuar cálculos matemáticos com operação modular em grupos finitos.
- Aumento em tempo de execução. Porém este tempo pode ser reduzido pré-definindo alguns cálculos matemáticos e executando outros de forma paralela. Esta é a estratégia seguida, por exemplo, na melhoria do tempo de execução do protocolo *SRP*.

4.6 Resumo

Neste capítulo, é apresentado um protocolo de autenticação de senhas em ambiente cifrado para acesso a base de dados denominado ***Cripto-SRP.*** São apresentados também os objetivos, características e a formalização do protocolo ***Cripto-SRP.*** O protocolo proposto manipula dados em um ambiente cifrado, melhorando a segurança da autenticação de senhas.

Comparando o ***Cripto-SRP*** com o *SRP*, o protocolo proposto é considerado mais seguro visto que as funções para criptografar o nome e verificar a senha do usuário são executadas em um ambiente cifrado. O ***Cripto-SRP*** implementa uma combinação de assinatura digital com exponenciação modular em ambiente de banco de dados cifrado. Estes incrementos aumentam a segurança no processo de autenticação de senhas.

5 CONCLUSÃO

Esta pesquisa teve como objetivo realizar um estudo sobre protocolos de segurança, em especial autenticação de senhas. Foi proposto um protocolo (***Cripto-SRP***) que é dividido em duas fases, autenticação e associação.

No protocolo proposto são gerados números aleatórios secretos que são considerados chaves secretas. Para cada sessão de acesso do usuário (***User***) ao servidor de banco de dados (***SBD***) tais números podem ser modificados. Isto aumenta a segurança da transmissão dos dados cifrados evitando, por exemplo, ataques do tipo repetição. Além disso, no protocolo proposto não é possível que intrusos identifiquem o nome do usuário, que é cifrado e manipulado no servidor na forma cifrada.

No caso em que não é necessário ocultar o nome do usuário, não é necessário calcular y_1 e y_2 no passo 1. Neste caso o servidor de banco de dados (***SBD***) armazena o nome do usuário na forma não cifrada e os valores *s* e *v*.

É importante também observar que a senha *P* não é transmitida na forma não cifrada. Neste caso, a proposta utiliza um protocolo similar ao *SRP* [WU 2002] onde o usuário (***User***) prova que conhece *P* sem transmiti-lo. Observe também que a senha *P*, não é armazenada, mas, sim os valores *s* e *v* que são utilizados para verificar se o usuário conhece ou não o valor de *P*.

Deve-se também observar que o protocolo deve ser implementado apenas em sistemas com capacidade de efetuar operações modulares em grupos finitos onde o problema do logaritmo discreto é intratável. Tais implementações são foco de grande pesquisa [KOBLITZ 1994], [MENEZES 1997], [ROSIGN 1999], [SCHNEIER 1996], [STALLINGS 1999], [STINSON 1995].

Finalmente, como o protocolo é executado apenas na autenticação do usuário (***User***) perante o servidor de banco de dados (***SBD***), seu custo computacional não é alto ao considerar o funcionamento do sistema ao longo do tempo. Isto ocorre porque a autenticação de usuários, geralmente não ocorre a todo instante.

O protocolo proposto utiliza fundamentalmente os protocolos de cifragem e assinatura de El-gamal [ELGAMAL 1985] e o *SRP* [WU 2002]. Tais protocolos têm-se mostrados seguros a ataques criptoanalíticos e pelo que se sabe, ainda não aparecem na literatura ataques que efetuam a sua criptoanálise [MENEZES 1993], [WU 2002]. Neste sentido, o protocolo proposto também é seguro.

As funções de verificação do nome e da senha do usuário, Ch_{User_Name} e $Ch_{Password}$ foram alteradas, aumentando assim a segurança ao transmitir e verificar os dados do usuário no banco de dados cifrado.

O protocolo ***Cripto-SRP*** criptografa além da senha, o nome do usuário utilizando chaves secretas e operações com exponenciação. Para cada instância de *login* criado no servidor de banco de dados (***SBD***), o protocolo ***Cripto-SRP*** criptografa os dados desta instância protegendo os usuários de possíveis ataques.

Um intruso pode tentar, durante a transmissão, capturar os dados transmitidos. Mas como cada sessão contém uma chave única, ao tentar o acesso ao sistema a validade da chave de sessão *K* já teria expirado. Com isso, o protocolo ***Cripto-SRP*** aumenta a segurança na autenticação de senhas atendendo aos atuais requisitos de segurança.

Como trabalhos futuros têm-se:

- Implementação do protocolo em um contexto real e não apenas didático.
- Avaliação deste tipo de implementação.

- Definição, implementação e avaliação do protocolo proposto utilizando grupos finitos em curvas elípticas.
- Paralelização das operações do protocolo proposto tendo como objetivo sua melhoria computacional.

REFERÊNCIAS BIBLIOGRÁFICAS

BELLARE, M. and Rogaway, P., "The AuthA Protocol for Password-Based Authenticated Key Exchange", Submission to the IEEE P1363 Password Authentication Study Group, March 14, 2000.

BELLOVIN ,S. M. and Merritt, M., "Augmented Encrypted Key Exchange: a Password-Based Protocol Secure Against Dictionary Attacks and Password File Compromise", AT&T Bell Laboratories (c. 1994).

BELLOVIN ,S. M. and Merritt, M., "Encrypted Key Exchange: Password-Based Protocols Secure Against Dictionary Attacks", Proceedings of the 1992 IEEE Computer Society Conference on Research in Security and Privacy, May 1992.

ELGAMAL, T., “A Public Key Cryptosystem and a Signature Scheme based on Discrete Logarithms”. IEEE Transactions on Information Theory, 31, 1985.

ELMASRI, R., NAVATHE, S. B., “Fundamentals of Database Systems”, Addison-Wesley, 3rd edition, 2000.

FERGUSON, N. e SCHNEIER, B., “Practical Cryptography”, Wiley Publishing, 2003.

GORDON, D.M., MCCURLEY, K.S., “Massively parallel computation of discret logarithms”. LNCS, 740, 1993.

HER-TYAN YEH , HUNG-MIN SUN., “Simple authenticated key agreement protocol resistant to password guessing attacks”, ACM SIGOPS Operating Systems Review, v.36 n.4, p.14-22, October 2002.

IVES, B. , “The domino effect of password reuse”, Communications of the ACM, v.47 n.4, p.75-78, April 2004 .

JABLON, D., "Strong Password-Only Authenticated Key Exchange", ACM Computer Communications Review, October 1996.

KAMALJIT, S., “On improvements to password security”, ACM SIGOPS Operating Systems Review, v.19 n.1, p.53-60, January 1985.

KOBLITZ, N., “A Course in Number Theory and Cryptography”, Springer Verlag, 1994.

KWON , T., "Authentication and Key Agreement via Memorable Password", Submission to the IEEE P1363 Password Authentication Study Group, May 2000.

MACKENZIE , P. and Swaminathan , R., "Secure Network Authentication with Password Identification", Submission to IEEE P1363a, August 1999.

MACKENZIE , P.. On the security of the SPEKE password-Authenticated Key Exchange Protocol. Technical Report 2001/057, Lucent Technologies, 2001.

MENEZES, A. J., Elliptic Curve Public Key Cryptosystem, Kluwer Academic Publishers, 1993.

MENEZES, A. J., VAN OORSHOT, P. C., VASTONE, S. A., Handbook of Applied Cryptography, CRC Press, 1997d.

MERKLE, R. C., HELLMAN, M. E., "Hiding Information and Signatures in Trapdoor Knapsacks", IEEE Transaction on Information Theory, 24, 1978.

MONROSE ,F., "Password hardening based on keystroke dynamics", Proceedings of the 6th ACM conference on Computer and communications security, p.73-82, November 01-04, 1999, Kent Ridge Digital Labs, Singapore.

NEUMANN , P. G., Risks of passwords, Communications of the ACM, v.37 n.4, p.126, April 1994.

PINKAS ,B. , Sander, T., "Securing passwords against dictionary attacks", Proceedings of the 9th ACM conference on Computer and communications security, November 18-22, 2002, Washington, DC, USA .

RIBEIRO, J., VIEIRA. F.J., "Melhorias em transmissão, armazenamento e checagem de *logins* em base de dados cifrado", IMESA, Agosto de 2004.

RIBEIRO, J., SOUZA. N. J., VIEIRA. F.J., "Uma extensão do protocolo *SRP* para transmissão segura de *logins* com acesso a base de dados cifrado", SECONF, Novembro de 2004 [2].

RIBEIRO, J., SOUZA. N. J., VIEIRA. F.J., "Transmissão segura de *logins* com acesso a base de dados cifrado", Publicado na seção de pôsteres do I2TS – International Information and Telecomunication Technologies, Dezembro de 2004 [3].

ROSIGN, M.,"Implementing Elliptic Curve Cryptography", Manning, 1999.

SCHNEIER, B., "Applied Cryptography: Protocols, Algorithms, and Source Code in C. 2". Ed. John Wiley and Sons, Inc, 1996.

SHOUP, V.,"A computational introduction to number theory and algebra", 2005

STALLINGS, W., "Cryptography and Network Security", Prentice Hall, 1999.

STINSON, D. R., "Cryptography – Theory and Pratic". Boca Raton: CRC Press, 1995.

WU, THOMAS., "SRP–6 Improvements and Refinements to the Secure Remote", Computer Science Department, Stanford, 2002.

WU, THOMAS., "The Secure Remote Password Protocol", Computer Science Department, Stanford Univerity, 1999.

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo