CENTRO UNIVERSITÁRIO DE BRASÍLIA – UNICEUB

FATECS – FACULDADE DE TECNOLOGIA E CIÊNCIAS SOCIAIS APLICADA

CURSO DE ENGENHARIA DA COMPUTAÇÃO

LEONARDO ARAÚJO DE OLIVEIRA MORALE

ANÁLISE DE ALTERNATIVAS PARA TRANSMISSÃO DE DADOS DE DISPOSITIVO MÓVEL PARA SERVIDOR REMOTO UTILIZANDO CRIPTOGRAFIA SIMÉTRICA

Orientador: Prof. Edison Ishikawa, D. SC.

Brasília
Junho, 2010.

ANÁLISE DE ALTERNATIVAS PARA TRANSMISSÃO DE DADOS DE DISPOSITIVO MÓVEL PARA SERVIDOR REMOTO UTILIZANDO CRIPTOGRAFIA SIMÉTRICA

Trabalho de conclusão do curso de Engenharia da Computação, da Faculdade de Tecnologia de Ciências Sociais Aplicadas do Centro Universitário de Brasília - UniCEUB.

Orientador: Profº. Edison Ishikawa, D.SC.

Brasília
Junho, 2010

AGRADECIMENTOS

Agradeço primeiramente ao meu amigo Fábio Garbin que me auxiliou em todos os momentos que precisei durante esse semestre, me apoiando, tirando todas as minhas dúvidas e também me ajudou com a implantação deste projeto.

Abaixo gostaria de citar algumas pessoas que contribuíram para o fechamento dessa minha etapa:

Ao meu professor e orientador, Dr. Edison Ishikawa, que compartilhou comigo seu imenso conhecimento técnico e por me direcionar para o caminho correto.

Ao meu pai por me auxiliar nas correções dos textos, por tirar minhas dúvidas com a parte técnica com a maior paciência do mundo e por todas as orientações que foram de grande valor;

À minha mãe que sempre me incentivou, esteve ao meu lado em todos momentos que precisei e me fez acreditar nos meus sonhos;

À minha irmã Carol que me ajudou com as traduções;

Ao meu amigo Luiz Campos que me ajudou muito com a parte de segurança em redes;

À minha querida esposa que sempre esteve ao meu lado, me dando apoio e força durante inúmeras madrugadas em claro;

E a todos os meus amigos que me incentivaram a concluir o curso.

DEDICATÓRIA

Dedico este trabalho ao meu pai Claudio Morale,

que durante todo o percurso de minha vida sempre foi o meu orgulho,

além de ser a fonte de incentivo, inspiração e exemplo em minha vida.

A minha mãe Maria Dalva,

que foi a garra, determinação e perseverança.

E a minha esposa Milena D. A. Morale,

por sempre estar ao meu lado me dando apoio, força e fazendo com que as coisas boas aconteçam.

RESUMO

Este trabalho visa implementar uma transmissão segura de dados, utilizando código não proprietário para aplicações em dispositivos móveis. Para isto, é utilizado um Sistema de Controle de Inventário como estudo de caso. Para a escolha do algoritmo de criptografia está sendo pesquisado na literatura dados para comparação dos algoritmos mais apropriados aos dispositivos móveis; ou seja, que fossem rápidos mesmo em dispositivos com pouco poder de processamento e baixo consumo de energia. A solução desenvolvida está implementada utilizando um computador notebook com sistema operacional Windows Vista e um Palm Top da PALM. No notebook está instalados uma máquina virtual (VM Ware) com sistema operacional Windows 2003 Server. Em outra máquina virtual está instalado um firewall em iptables para controlar o tráfego de entrada e saída de informações da rede. Para assegurar o conteúdo das informações entre o palm e o servidor, utiliza-se um algoritmo de criptografia simétrico. O sistema de inventário é utilizado por um Gerente Territorial que, com o palm top em mãos, dirige-se, em datas pré definidas, até um dos clientes de sua carteira para realizar as contagens de produtos existentes no estoque. Após as contagens de produtos nos clientes, o Gerente Territorial insere essas informações no palm e, por meio dele, faz a transmissão das informações para o servidor na sede da empresa. Por isso, a necessidade de efetuar uma transmissão segura entre o dispositivo móvel e o servidor.

ABSTRACT

This work presents an theorical analysis of 3 (three) algorithms of symmetric cryptography and it is also a comparative study of these three algorithms. Finally, this work describes the using of one of these algorithms in a inventory of products control system. A computer which has a Windows Vista operational system was used to carry out the procedure. Other systems were used, like the Palm Top (from PALM), a virtual machine (VM Ware) with Windows 2003 Server operational system, another VM Ware with a firewall using iptables to control the input and output of net information. An algorithm of symmetric cryptography was used to ensure the information between Palm and the server. The inventory system is done by a territorial Manager that uses the Palm Top to head for one of the clients, on dates that were defined before. This procedure is done in order to count the products that are stored in the clients storage. After counting up the clients products, the Territorial Manager insert these information in the PALM. After that, the PALM tranfer on the information to the servers at the company.

SUMÁRIO

LISTA DE FIGURAS E TABELAS

LISTA DE SIGLAS, ABREVIATURAS E PALAVRAS CHAVE

CRIPTOGRAFIA	É o estudo dos princípios e técnicas pelas quais a informação pode ser transformada da sua forma original para outra ilegível, de modo que possa ser conhecida apenas por seu destinatário.
ATAQUES PASSIVOS	Ataques que possuem a natureza de esquadrinhar ou monitorar informações, sem alterá-las.
ATAQUES ATIVOS	Ataques que envolvem alguma modificação do fluxo de dados ou a criação de um fluxo falso, tais como: modificação de mensagem.
AUTENTICAÇÃO	É garantir ao destinatário a origem da mensagem, isto é, que a mensagem é proveniente de onde ela afirma ter vindo.
INTEGRIDADE DE DADOS	"É a garantia de que os dados recebidos estão exatamente como foram enviados por uma entidade autorizada, ou seja, não contém qualquer modificação." [1]
CIFRA	É o ato de alterar a mensagem original por meio de mudança, de suas letras ou fonemas, de ordem, aparência ou tipo, de modo a torná-la ilegível a quem a interceptar e não possuir a informação de como reproduzir a mensagem original.
ADVPL	É a linguagem de programação nativa do Sistema de Gestão Integrado, da Microsiga.
AES	Sigla para Advanced Encryption Standard.
DES	Sigla para Data Encryption Standard.
3DES	Triplo DES sigla para Triple Data Encryption Standard.
BLOWFISH	Cifra simétrica de blocos.
RC4	RC4 é o algoritmo de criptografia de fluxo.
ERP	Enterprise Resourcing Planning

1 - INTRODUÇÃO

Neste capítulo é feita a apresentação deste projeto descrevendo por meio dos tópicos motivação, objetivos e estrutura da monografia.

1.1 Motivação

A motivação desse projeto surgiu de uma necessidade da empresa Dinâmica Serviços, empresa privada prestadora de serviço de limpeza em geral, que atua com clientes particulares e com órgãos públicos, a qual utiliza um sistema de Gestão Integrado (ERP) para gerenciar seus processos internos, tais como: Contabilidade, Faturamento, Finanças, Estoque, Compras e etc. Com base nesse ERP, desenvolveu-se uma aplicação para funcionar em palm top, modelo PalmTX, do fabricante PALM, com o objetivo de inventariar os estoques existentes nas frentes de serviços, ou seja, o estoque existente em cada um dos clientes da empresa Dinâmica Servicos. Essa aplicação foi desenvolvida na mesma linguagem do sistema ERP, que é a ADVPL. Hoje essa aplicação está pronta e em produção.

O inventário do estoque nos clientes é feito diretamente nas frentes de serviços, por Gerentes Territoriais - GTO, sendo cada um deles, responsável por uma região que engloba um número variável de clientes. Ao final do dia, os GTO´s necessitam descarregar as informações inventariadas contidas no palm top no Servidor, atualizando, desta forma, a base de dados do sistema ERP. Essa transmissão pode ser feita também internamente pela rede local, além da Internet.

Para essas transmissões entre o palm e o servidor, principalmente nos casos das transmissões fora da rede local, ou seja, via internet, percebeu-se que a

integridade das informações durante a transmissão tornava-se um alvo muito frágil para ataques passivos e ataques ativos, colocando em risco, dados valiosos da empresa. Esta constatação mostrou a necessidade de uma transmissão mais segura, objeto do estudo e implementação deste projeto.

1.2 Objetivos

O objetivo do projeto é realizar uma análise teórica de alternativas de criptografia simétrica, que possam ser utilizadas para a transmissão segura de dados entre um dispositivo móvel e um servidor.

Considerar as transmissões fora da rede local, onde a vulnerabilidade das informações é maior e, por essa razão, necessita de maiores requisitos de segurança.

Suprir as necessidades de segurança das transmissões entre o palm e o servidor, considerando também as restrições físicas do dispositivo móvel e os requisitos de desempenho da solução para, ao final, chegar ao algoritmo que melhor se enquadre no contexto do estudo de caso.

Avaliar soluções proprietárias e open source de modo a ter parâmetros que orientem a melhor escolha.

Por fim, implementar o algoritmo de criptografia simétrica que melhor se adapte ao quadro proposto de modo a impedir ataques e, também, garantir a integridade e a confidencialidade das informações.

1.3 Estrutura da Monografia

Este projeto é constituído de cinco capítulos. Segue uma breve descrição.

Capítulo 1 – **INTRODUÇÃO –** aborda o que será tratado no projeto bem como a motivação e o objetivo.

Capítulo 2 - APRESENTAÇÃO DO PROBLEMA – aborda o problema identificado e a solução proposta.

Capítulo 3 - REFERENCIAL TEÓRICO - aborda conceito e definições como, história da criptografia, tipos de criptografia, suas características de segurança, segurança na WEB, comunicação do palm com o servidor, assim como o uso das tecnologias dos algoritmos apresentados. Também, abordar o uso de firewall Iptables com a ferramenta Firewall Builder e uso máquina virtual VM Ware.

Capítulo 4 - PROPOSTA DE SOLUÇÃO E MODELO – apresenta o protótipo desenvolvido neste projeto, ferramentas utilizadas, detalhes de implantação, hardware e software.

Capítulo 5 – IMPLEMENTAÇÃO DA ALTERNATIVA QUE UTILIZA O ALGORITMO RC4 – Apresenta solução implementada.

Capítulo 6 – CONCLUSÃO - Descreve as conclusões obtidas nesse projeto.

REFERÊNCIAS BIBLIOGRÁFICAS

2 - APRESENTAÇÃO DO PROBLEMA

Este capítulo contém as descrições do problema e da solução proposta.

“A criptografia é um dos principais métodos existentes para proteger informações eletrônicas valiosas.” [01]

O parágrafo acima relata uma realidade dos dias atuais e por isso, corrobora a necessidade de implementar um algoritmo de criptografia para garantir a autenticidade e a integridade do conteúdo das informações transferidas entre o palm top e o servidor.

Considerando a descrição, feita na motivação, sobre a forma de trabalho da empresa para inventariar estoques junto aos clientes e a sua necessidade de transmitir os dados inventariados fora da sua rede local, onde a segurança contra a possibilidade de interferência e violação do conteúdo transmitido por terceiros não é conhecida, torna o processo vulnerável e, por tanto, não confiável.

Na busca da solução do problema acima referenciado, o presente trabalho implementa, após o trabalho de análises teórica dos algoritmos, a melhor alternativa.

2.1 Análise dos algoritmos de criptografia

Para a implementação da solução é necessária, previamente, a análise de forma geral dos principais algoritmos simétricos citados na literatura e, também, de algumas soluções proprietárias existentes. Após a análise, selecionar três dos

algoritmos simétricos que melhor se enquadram na solução proposta e apresentá-los no referencial teórico desse projeto. Por fim, a escolha do melhor algoritmo como solução.

Dessa forma, os algoritmos comparados com código fonte aberto, são: AES, DES, 3DES, Blowfish e o RC4 e as soluções proprietárias são: Mergic VPN [02], Versa Mail [03] e AnthaVPN [04].

O Quadro 2.1, apresenta um resumo comparativo com algumas características de cada um dos algoritmos propostos. Outra forma com que os algoritmos podem ser classificados é de acordo como tratam os dados. Ou seja, os dados podem ser criptografados em blocos de tamanhos determinados ou de forma contínua denominada criptografia de fluxo. Assim, temos aplicações ideais para cada tipo de trabalho de cifragem onde cada algoritmo deve ser determinado não só pelas características de segurança, mas também pela aplicação como visto no Quadro 2.2. [05], [06]

Os algoritmos de fluxo, geralmente, são mais utilizados para criptografar imagens, e, os algoritmos de blocos são mais utilizados para cifrar textos, pois para o tratamento de imagens utilizando as cifras de blocos, seus métodos não escondem os padrões de dados repetitivos. Com isso, não oferece confidencialidade. No entanto, há estudos em que os algoritmos de bloco podem ser usados como algoritmo de fluxo, usando diferentes modos de operação. Entretanto essas operações tornam esses algoritmos mais lentos em relação à cifra de fluxo. [07]

A Figura 2.1, ilustra de forma bem clara a vulnerabilidade ao se usar métodos de cifras de blocos para criptografar imagens quando comparados com

outras maneiras mais seguras, pois se percebe facilmente que padrões na imagem original acabam gerando padrões na imagem criptografada, o que permite reconhecer a imagem original.

Figura 2.1 – Vulnerabilidade da criptografia de blocos em imagens [08]

Quadro 2.1 – Características dos algoritmos simétricos proposto [05]

Algoritmo	**Tipo de Algoritmo**	**Tamanho da Chave**	**Tamanho do Bloco**	**N°de rodadas**	**Operações matemáticas**
AES	Algoritmo de cifra de blocos.	128, 192 ou 256 bits	128, 192, 256 bits	10, 12, 14	XOR, S-Boxes fixas
DES	Algoritmo de cifra de blocos.	56 bits	64 bits	16	XOR, S-Boxes fixas
3DES	Algoritmo de cifra de blocos baseado no DES.	112 ou 168 bits	64 bits	48	XOR, S-Boxes fixas
Blowfish	Algoritmo de cifra de bloco.	Variável entre 32 ate 448 bits	64 bits	16	XOR, S-Boxes variáveis, adição
RC4	Algoritmo de cifra de fluxo.	Variável entre 40 até 256 bits	-	-	Adição, XOR

Quadro 2.2 – Algoritmos por aplicação [18]

Aplicação	**Cifragem Recomendada**	**Comentários**
Banco de Dados	Cifra de Bloco	A interoperabilidade de um outro software não é relevante, mas é necessário reutilizar as chaves.
SSL	RC4	A velocidade é extremamente importante, cada conexão pode ter uma nova chave. Assim a maioria dos navegadores e servidores possuem RC4.
Criptografia de Arquivos	Cifra de Bloco	A interoperabilidade não é relevante, porém cada arquivo pode ser cifrado com a mesma chave.

Já em relação às soluções proprietárias, para cada uma delas, foram apresentadas as seguintes características abaixo:

Mergic VPN

O Mergic VPN foi implementado pela empresa Mergic Inc. [02]

Ele possui as seguintes características:

Auto Connect. Usuário selecionado automaticamente aplicações podem se conectar e desconectar do servidor VPN, conforme necessário, sem que o usuário necessite iniciar a aplicação Mergic VPN;

Split tunelamento. Permite o acesso simultâneo à Internet e rede privada;

Suporta conexões TCP/IP usando IEEE 802.11 (Wi-Fi) módulos LAN sem fio, modems com e sem fio.

Possui um custo de US $49,00 (quarenta e nove) dólares por PALM. [02]

Versa Mail 3.1c

O Versa Mail 3.1c foi desenvolvido pela empresa Palm. [03] Ele suporta POP3, IMAP e Exchange ActiveSync e-mail, com suporte a VPN. Além disso, há soluções de terceiros para enviar e-mail de qualquer lugar. Também possui suporte para enviar e receber documentos do MS Word e MS Excel, fotos, ringtones, vídeos entre outros.

Suporta conexões TCP/IP por meio de Wi-Fi e, também, por meio de Bluetooth e GPRS.

Possui um custo de US $9,99 (nove e noventa e nove) dólares por PALM.[03]

AnthaVPN

AnthaVPN é um cliente VPN, construído no padrão IPSec, que fornece acesso sem fio seguro aos recursos da rede corporativa. [04]

AnthaVPN suporta algoritmos como, RSA, 3DES e criptografia de chave pública, criptografia de curva elíptica (ECC) é uma técnica de criptografia de chave pública oferecendo 128 - bits de segurança. Possui o custo de €59,00 (cinqüenta e nove) euros por PALM. [04]

No entanto, apesar das soluções pagas apresentarem um custo financeiro baixo, levando-se em conta apenas o custo unitário de apenas uma licença por palm top, não é possível saber o que a fonte desses programas efetivamente fazem, já que o código fonte é completamente fechado, bem como confiar nas especificações que dizem estar implementando determinado algoritmo. Também não é possível

garantir total transparência de suas rotinas, muito menos a inexistência de “trap doors” e/ou abertura para possíveis ataques.

Outro fator importante que se deve levar em consideração é a quantidade significativa de recursos computacionais, tais como: tempo de CPU, memória e bateria que um algoritmo de criptografia consome. Um dispositivo sem fio, geralmente possui recursos bastante limitados, principalmente a bateria, que está sujeita ao problema do consumo de energia, devido ao processamento extra dos algoritmos de criptografia. A criptografia também é essencial para outros serviços de segurança, como autenticação, integridade de dados e controle de acesso. [09]

Nas pesquisas realizadas para esse projeto não foram encontradas soluções abertas para o PALM que está sendo utilizado, que é o Palm-TX.

Sendo assim, após a leitura do capitulo 2, é possível concluir que a criptografia em dispositivos móveis, consome recursos e isto deve ser levado em conta no projeto. Após alguns comparativos, é feita a escolha dos algoritmos: AES, Blowfish e RC4 que, pelo fato de apresentar melhor nível de segurança, maior simplicidade em suas operações e maior velocidade quando comparados ao DES e ao 3DES.

No capítulo 3, é feita uma análise teórica, de cada um dos algoritmos citados no parágrafo anterior e, com isso, concluir-se-á qual será a melhor opção para a implementação neste projeto. Também, são definidos alguns conceitos, tais como: criptografia simétrica, criptografia assimétrica ou de chave pública, cifra de fluxo e cifra de blocos, VPN, Firewall, virtualização e segurança na rede WiFi.

3 - REFERENCIAL TEÓRICO

Neste capítulo, faz-se referência teórica aos assuntos abordados no desenvolvimento deste projeto.

3.1. Criptografia

3.1.1. História da criptografia

Acredita-se que o conceito de criptografia foi utilizado pela primeira vez em 1900 a.c., no Egito, em suas inscrições. Alguns séculos depois, os hebreus vieram a utilizar cifras de substituição simples. Posteriormente, o Império Romano utilizou a conhecida Cifra de Cesar, que apresentava métodos mais clássicos de criptografia, para proteger-se de seus inimigos nas comunicações durante as batalhas. Já na idade moderna, devido aos esforços das guerras, vale destacar que a criptografia foi largamente utilizada pelos EUA, União Soviética e Alemanha, que criaram diversos algoritmos de criptografia, a fim de cifrar mensagens a respeito de estratégias e operações de guerra. [01]

Com o avanço da criptografia e da comunicação digital, surgiram padrões e diversos tipos de algoritmos de criptografia, tais como cifras de bloco e de fluxo, algoritmos por chave simétrica, chave assimétrica, dentre outros. Atualmente, a criptografia é amplamente utilizada na WEB, a fim de autenticar os usuários, para lhes fornecer acesso, na proteção de transações financeiras e em comunicações. [01]

Também é importante destacar, algumas concepções erradas com relação à criptografia simétrica. Primeiro é que a “criptografia de chave pública é mais segura contra a criptoanálise do que a criptografia simétrica”, pois a segurança de qualquer criptografia depende do tamanho da chave e do trabalho computacional envolvido para quebrar uma cifra. Do ponto de vista de resistência à criptoanálise, não há nada que torne uma superior à outra. [01]

O segundo erro conceitual é que a “criptografia de chave pública é uma tecnologia que tornou a criptografia simétrica obsoleta”. Pois, a criptografia de chave pública é usada para resolver o problema da distribuição de chaves. Isso porque em redes de comunicação de grande porte, a distribuição de chaves pode ser um problema significativo. Por esse motivo, utilizando um par de chaves, uma para criptografia e outra para a decriptografia, não há necessidade de distribuí-las antes da transmissão. [10]

Assim, podemos dizer que os algoritmos de criptografia simétrica, tendem a ser mais rápidos do que os algoritmos de chave pública pelo fato de usarem a mesma chave de cifragem. Isso é um dos condicionantes que define a sua escolha para grandes volumes de dados. [01]

3.1.2. Tipos de criptografia

Atualmente existem inúmeras técnicas de segurança e de criptografia, tais como, segurança para utilização de e-mail, segurança na web, segurança sobre ip, firewalls, criptografia simétrica, criptografia assimétrica e outros. Portanto, nos próximos tópicos são abordados alguns assuntos, como: algoritmo de criptografia

simétrica, técnicas de substituição e transformação, algoritmo de criptografia assimétrica e VPN. Também é feita uma breve abordagem sobre firewall e virtualização.

3.1.3. Criptografia simétrica

A criptografia simétrica, também conhecida como criptografia convencional, é uma forma de criptosistema em que a cifragem e a decifragem são realizadas usando a mesma chave, portanto, se tanto o emissor quanto o receptor utilizarem a mesma chave, o sistema é considerado simétrico. [01]

A criptografia simétrica transforma o texto claro em texto cifrado, usando uma chave secreta e um algoritmo de criptografia. Usando a mesma chave e um algoritmo de decriptografia, o texto claro é recuperado a partir do texto cifrado. [01]

A característica da criptografia simétrica é que não precisamos manter o algoritmo secreto, o que realmente precisamos, é manter apenas a chave secreta.[10]

Geralmente, a maioria dos algoritmos criptográficos usados em dispositivos sem fio são baseadas na criptografia de chaves simétricas. Um exemplo é o RC4. O RC4 é uma cifra de fluxo desenhada por Ron Rivest em 1987 e é amplamente utilizada em muitas aplicações e em redes sem fio como o IEEE 802.11 e WEP [11].

Assim, podemos dizer que a criptografia simétrica oferece um grau de autenticação aceitável, bem como de confidencialidade, porém não oferece assinatura. [01]

3.1.4 Modelo de Cifra simétrica

A Figura 3.1 ilustra um resumo simplificado do modelo de criptografia simétrica, indicando os seus 5 estágios: [01]

1. Texto Claro: Dados ou mensagens originais alimentados no algoritmo;
2. Algoritmo de Criptografia: O algoritmo realiza diversas substituições e transformações no texto claro;
3. Chave secreta: A chave também é entrada para o algoritmo. A chave é um valor independente da mensagem e do algoritmo. Dependendo da chave utilizada no momento, o algoritmo produzirá uma saída diferente.
4. Texto cifrado: Essa é a mensagem codificada, cifrada.
5. Algoritmo de decriptografia: É basicamente o processo inverso do algoritmo de criptografia. Juntamente com a chave secreta, a partir do texto cifrado é gerado o texto claro.

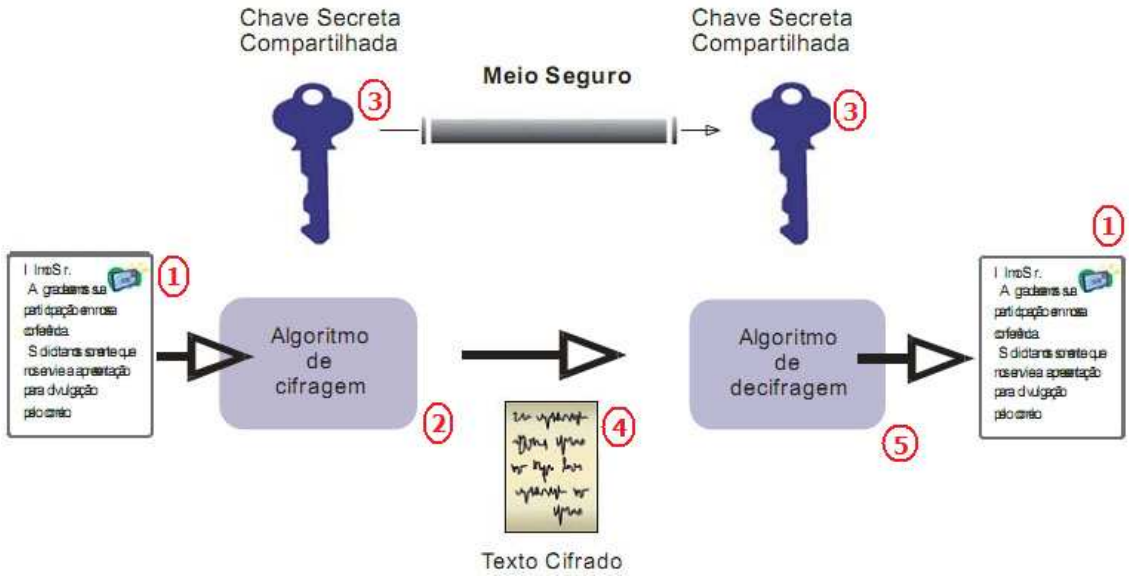

Figura 3.1 - Modelo simplificado de criptografia convencional [12]

O modo como o texto claro é processado, determina-se se o tipo de cifra é de bloco ou de fluxo. Na cifra de bloco cada bloco é processado por vez, produzindo um bloco de saída para cada bloco de entrada. Já a cifra em fluxo processa os elementos de entrada, continuamente, produzindo a saída de um elemento de cada vez, enquanto prossegue.

3.1.5 Técnicas de Substituição

Técnicas de substituição são aquelas em que as letras de texto em claro são substituídas por outras letras ou por números ou símbolos ou, ainda, por uma combinação destes de acordo com um sistema predefinido e uma chave. Alguns exemplos dessa técnica são: Cifra de Cesar [01], Cifras monoalfabéticas [01], cifras polialfabéticas[01], outras. [01]

Para um melhor entendimento, será descrito, de forma breve, um exemplo com a cifra de César.

A Cifra de César foi uma das primeiras técnicas de substituição a ser utilizada, e também uma das mais simples. Ela consiste em substituir cada letra do alfabeto pela letra que fica algumas posições adiante no alfabeto; O exemplo a seguir usa três posições adiante no alfabeto:

TEXTO CLARO: Encontre-me hoje à noite

TEXTO CIFRADO: HQFRQXUH PH LRNH D QRMXH

3.1.6 Técnicas de Transposição [01]

Essa técnica é obtida utilizando algum tipo de permutação nas letras do texto claro, conforme a cifra utilizada. Por exemplo, a cifra mais simples desse tipo é a de *rail fence*, em que o texto claro é escrito como uma seqüência de diagonais e depois lido como uma seqüência de linhas. Por exemplo, para cifrar a mensagem, Encontre-me hoje à noite, com essa cifra de profundidade 2, escrevemos assim:

e c n r m h j a o t
n o t e e o e n i e

A mensagem cifrada é: ECNRMHJAOTNOTEEOENIE

3.1.7. Criptografia assimétrica

A criptografia assimétrica, também conhecida como criptografia de chave pública, é uma forma de criptosistema onde a criptografia e a decriptografia são realizadas usando diferentes chaves, ou seja, uma chave pública e uma chave privada. Em um algoritmo de criptografia assimétrica, uma mensagem criptografada com a chave pública pode somente ser decifrada pela sua chave privada correspondente. Do mesmo modo, uma mensagem criptografada com a chave privada pode somente ser decifrada pela sua chave pública correspondente, ou seja, o emissor e o receptor precisam ter uma das chaves do par casado de chaves, não a mesma chave. [01]

A criptografia de chave pública também é usada para resolver o problema da distribuição de chaves, pois em redes de comunicação de grande porte, a

distribuição de chaves pode ser um problema significativo. Por esse motivo, utilizando duas chaves, uma para criptografia e outra para a decriptografia, não há necessidade de distribuí-las antes da transmissão. Entretanto, a criptografia de chave pública é baseada em funções matemáticas, computação intensiva e não muito eficiente para pequenos dispositivos sem fio. No entanto, ela é utilizada para distribuições de chaves simétricas [13].

3.1.8 VPN

A Virtual Private Network (VPN), é uma rede privada construída sobre a infra-estrutura de uma rede pública, normalmente a Internet. Ou seja, ao invés de se utilizar links dedicados ou redes de pacotes (como Frame Relay e X.25) para conectar redes remotas, utiliza-se a infra-estrutura da Internet. [14]

Como o custo de links dedicados são muito maiores quando comparados a uma simples conexão de internet, as empresas cada vez mais utilizam esse recurso mais econômico. No entanto, o quesito segurança certamente não é o ponto forte.

Para solucionar esse problema a VPN surgiu para superar o problema de segurança, usando protocolos de tunelamento e procedimentos de criptografia. A integridade e autenticidade dos dados é assegurada, conforme a Figura 3.2.

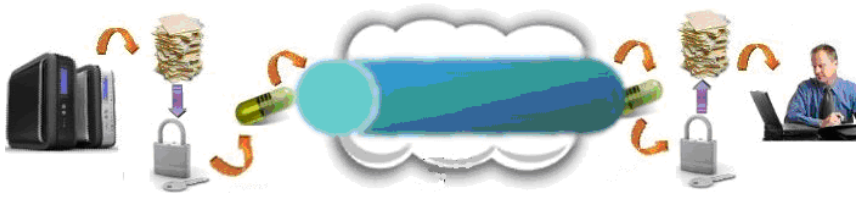

Figura 3.2 - Ilustração de uma conexão VPN [15]

Basicamente uma VPN pode ser feita de duas formas:

A primeira forma é um simples host em trânsito conectando-se em um provedor de Internet e através dessa conexão, se estabelece um túnel na rede remota. A Figura 3.3 ilustra essa forma.

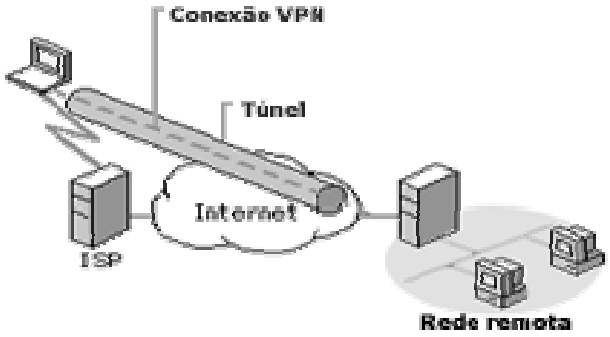

Figura 3.3 - Conexão entre 2 redes utilizando VPN [16]

Na segunda forma, duas redes se interligam através de hosts com link dedicado ou discado, via internet, formando assim um túnel entre as duas redes. A Figura 3.4 ilustra essa forma.

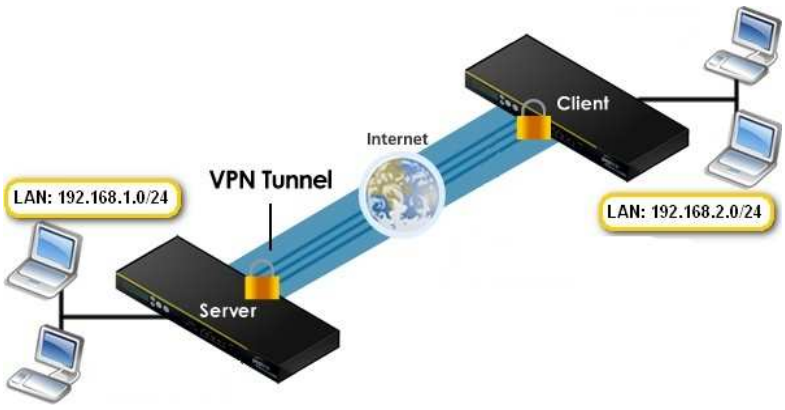

Figura 3.4 - Conexão entre 2 redes utilizando VPN [17]

Os protocolos utilizados no túnel virtual, são, Internet Protocol Security (IPSec), Layer 2 Tunneling Protocol (L2TP), Layer 2 Forwarding (L2F) e o Point-to-Point Tunneling Protocol (PPTP). O protocolo escolhido, será o responsável pela conexão e a criptografia entre os hosts da rede privada. Eles podem ser normalmente habilitados através de um servidor Firewall ou RAS que esteja trabalhando com um deles agregado.

3.2. Algoritmos de criptografia simétricos

3.2.1. AES

O algoritmo Advanced Encryption Standard (AES), anteriormente chamado de Rinjdael foi adotado pelo governo americano como protocolo padrão de criptografia. Ele também foi adotado como substituto oficial do DES pelo Nacional institute and Technology (NIST). [01]

O AES é um algoritmo rápido, tanto em software quanto em hardware e requer pouca memória. Ele é um cifrador de blocos com um tamanho de bloco fixo de 128 bits e com um tamanho de chave variável de 128, 192 e 256 bits. [01]

O AES opera sobre um arranjo bidimensional de bytes com 4x4 posições, denominado de estado. Para criptografar, cada etapa do AES (exceto a última), consiste de quatro etapas, sendo uma de permutação e três de substituição. [01]

1. SubBytes: Utiliza uma caixa-S para realizar uma substituição linear byte a byte do bloco;

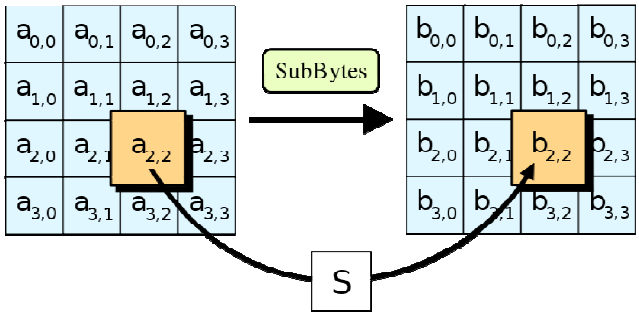

Figura 3.5 – Etapa SubBytes [18]

2. ShiftRows: É uma operação de permutação simples de transposição nas linhas do estado, ou seja, cada fileira do estado é deslocada de um determinado número de posições;

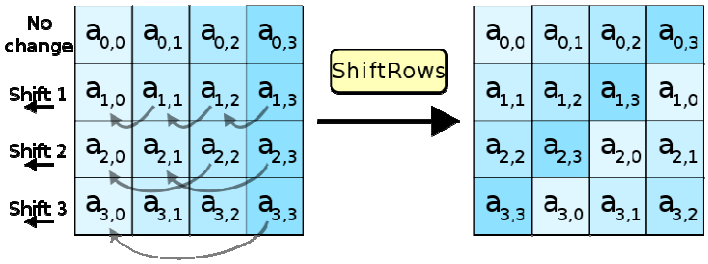

Figura 3.6 - Etapa ShitRows [18]

3. MixColumns: É uma operação nas colunas do estado. Faz-se uma mistura dos dados com cada coluna da matriz de estados.

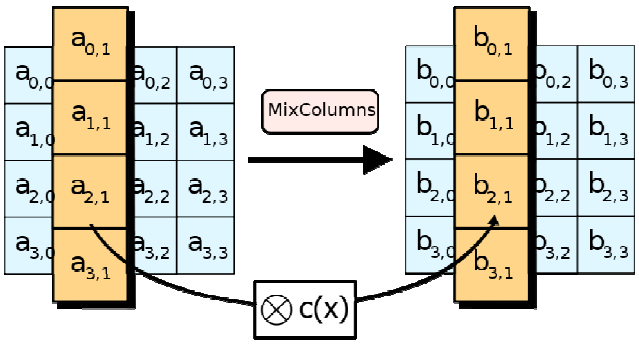

Figura 3.7 – Etapa MixColumns [18]

4. AddRoundKey: Cada byte do estado é combinado com a subchave própria do turno (RoundKey); cada subchave é derivada da chave principal usando o algoritmo de agendamento de chaves.

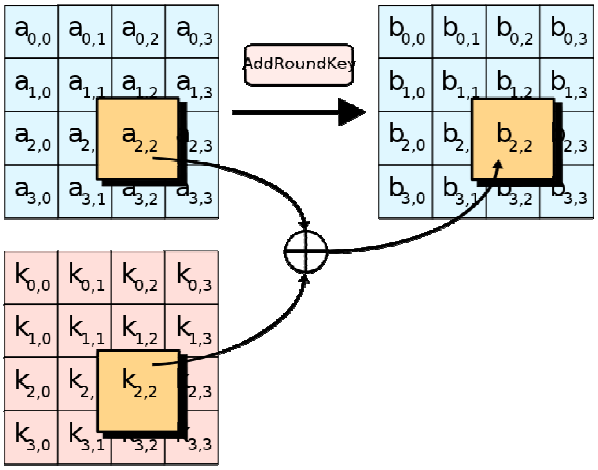

Figura 3.8 – Etapa SubBytes [18]

A Estrutura do AES, tanto para criptografar como decriptografar, a cifra começa pela etapa AddRoundKey, seguida por nove rodadas, onde cada uma das rodadas inclui as quatro etapas que foram especificadas acima. Por último é feito uma décima rodada com apenas três etapas.

Cada estágio pode ser reversível, no entanto o algoritmo de decriptografia não é idêntico ao algoritmo de criptografia. Isso é uma conseqüência da estrutura específica do AES.

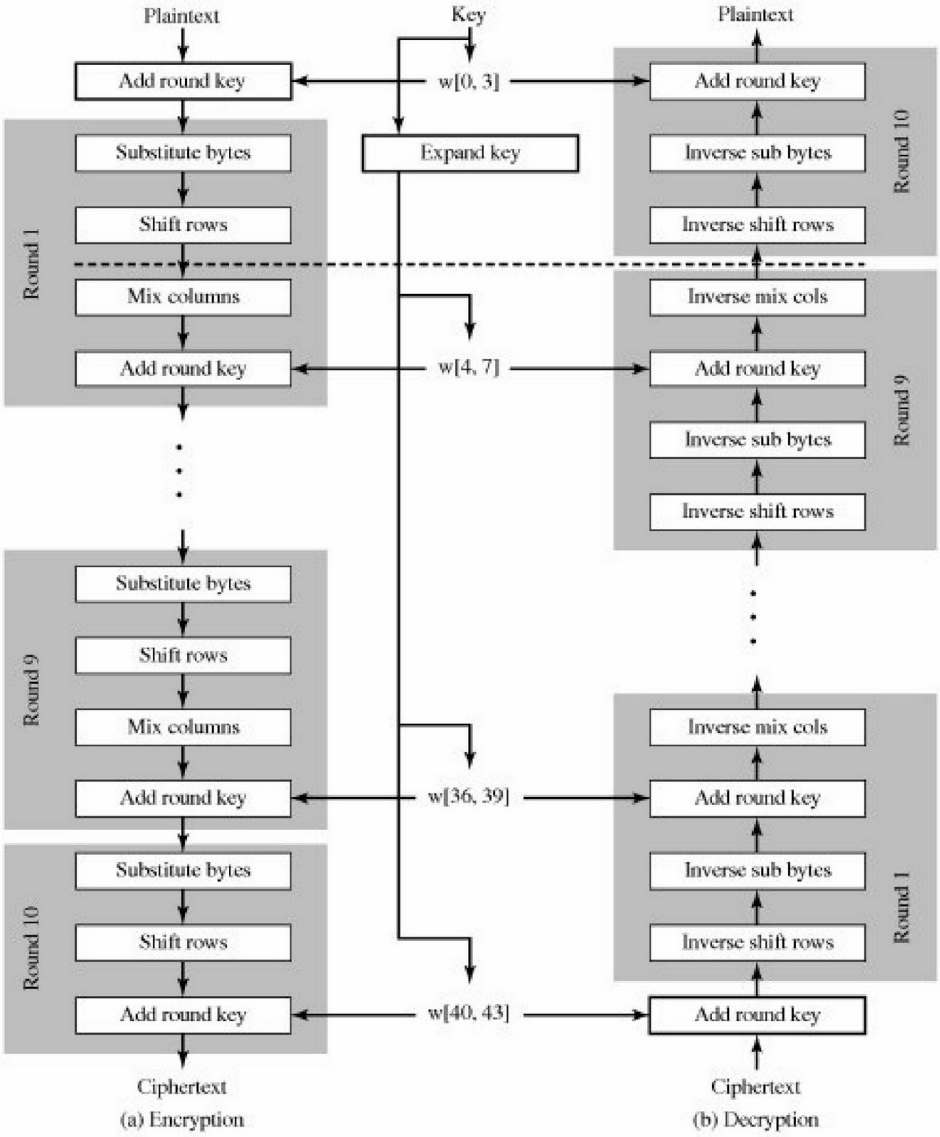

Figura 3.9 – Estrutura do AES [19]

3.2.2. Blowfish

O Blowfish é um algoritmo de cifra de bloco desenvolvido por Bruce Scneier. O bloco tem um tamanho de 64bits e a chave pode variar entre 32 à 448bits de comprimento. Esta flexibilidade é uma ótima vantagem, pois se optarmos dar prioridade à velocidade, podemos escolher uma chave pequena, por exemplo de

32bits. No entanto, se optarmos em utilizar chaves maiores até 448bits, o algoritmo deixará de ter sua vantagem na velocidade para ter mais segurança. O nome blowfish significa peixe-assopro, representado pelo peixe baiacu, que tem a propriedade de inchar e desinchar. É uma analogia com respeito à flexibilidade do ramanho da chave do algoritmo. [20]

Suas principais características são: [20]

a) Rapidez - ainda hoje, Blowfish é um dos algoritmos de cifra de bloco em uso mais rápidos;

b) Pequeno tamanho – Blowfish pode rodar em menos de 5K de memória;

c) Simplicidade - A estrutura de código do blowfish é simples;

d) Segurança variável – O comprimento da chave pode ser escolhido, podendo ser de 32 a 448 bits;

3.2.3. RC4

O RC4 é uma cifra de fluxo projetada em 1987 por Ron Rivest para a RSA Security. O RC4 é usado nos padrões Security Sokets Layer/Transport Layer Security (SSL/TLS), definido para a comunicação entre os navegadores Web e servidores. É também usado para proteção de tráfego na internet e nos protocolos Wired Equivalent Privacy (WEP) e WiFi Protected Access (WAP), que fazem parte do padrão de LAN sem fio IEEE 802.11. [01]

O RC4 é conhecido por ser rápido e eficiente. Ele pode ser escrito usando apenas algumas linhas de códigos e requer apenas 256 bytes de memória RAM. [01]

Este algoritmo é uma cifra de fluxo, ou seja, cada byte gerado é imediatamente utilizado, com tamanhos de chaves variados, normalmente entre 40 a 256 bits e operações orientadas a byte. [01]

De uma forma geral, o algoritmo consiste em utilizar um array que a cada ciclo tem os seus valores permutados e misturados com a chave, o que faz com que seja muito dependente desta. Esta chave, utilizada na inicialização do array, pode ter até 256 bytes (2048 bits). O algoritmo é mais eficiente quanto menor for a chave.. Esta característica torna RC4 fácil de implementar em dispositivo programável. O processo de pesquisa do RC4 está ilustrado na Figura 3.10. [01]

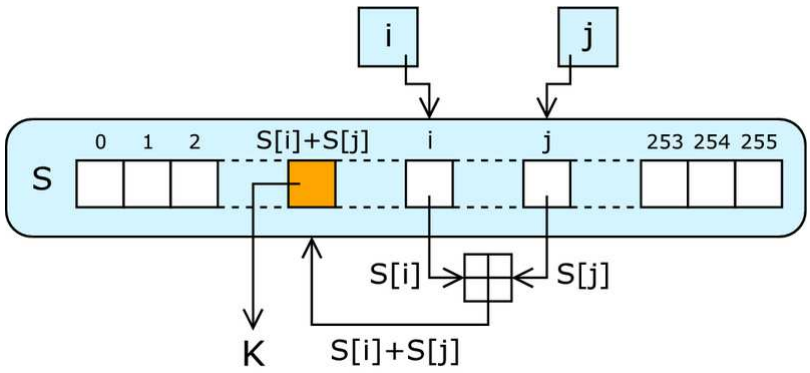

Figura 3.10 - Processo de pesquisa do RC4 [21]

Por isso, para aplicações que exigem criptografia/decriptografia de um fluxo de dados, como sobre um canal de comunicações ou um link de um navegador, essa cifra de fluxo pode ser a melhor alternativa. Porém, para aplicações

que lidam com bloco de dados, e-mail e transferências de arquivos, a cifra de blocos é a mais recomendável. No entanto, qualquer tipo de cifra pode ser usado em praticamente todas as aplicações. [01]

Quadro 3.1 - Comparação das velocidades de cifras simétricas num PentiumII [01]

Cifra	**Tamanho de chave**	**Velocidade (Mbps)**
DES	56	9
3DES	168	3
RC4	Variável	45

Conforme o especialista William Stallings, o algoritmo RC4 é usado para transmitir informações sobre um canal de comunicações, entre a comunicação de navegadores web e servidores, ou para proteção de tráfego na internet nos protocolos WEP e WAP, que fazem parte do padrão de LAN sem fio IEEE 802.11. [01]

Diante dessas análises, podemos dizer que o algoritmo RC4 é um excelente algoritmo, quando analisado dentro de certo contexto, como é o caso desse projeto. Isso porque utiliza a transmissão de informações por meio da tecnologia WiFi e, também, não menos importante, a quantidade de informações a serem transmitidas, é relativamente pequena, e o tempo em que o conteúdo criptografado estará em transito é curto. Vale ressaltar que o mesmo algoritmo e chave são utilizados tanto para criptografia como para decriptografia dos dados.

Assim. o nível de segurança que almejamos nesse projeto deve levar em conta a baixa capacidade de processamento do dispositivo móvel e as limitações da fonte de energia. Por fim, a complexidade dessa tecnologia e o custo financeiro se tornam menores quando comparado aos demais algoritmos expostos aqui, o que justifica a sua escolha.

3.3. Consumo de energia em dispositivos móveis

Os algoritmos de criptografia são conhecidos como sendo de computação intensiva. Eles consomem quantidade significativa de recursos computacionais, tais como: tempo de CPU, memória e energia. O dispositivo móvel, usualmente, tem recursos limitados principalmente no que se referem à capacidade da bateria. O objeto aqui abordado é o problema sobre o consumo de energia devido a execução do algoritmo de criptografia. Além disso, unidades de rede sem fio comumente transmitem e recebem dados num trecho de rede sem fio. Os dados são protegidos – confidencialmente – antes da transmissão, utilizando algoritmo de criptografia, para mantê-los seguros. A criptografia também é essencial para outros serviços de segurança tais como: autenticaçãos, integridade de dados e controle de acesso. Devido ao uso intensivo da computação inerente aos algoritmos de criptografia, estes tendem a consumir uma quantidade substancial de energia ou bateria. A bateria pode ser rapidamente descarregada devido à criptografia, principalmente para um pequeno dispositivo sem fio. [10]

3.4 Comparação de Cifras

3.4.1 Desempenho

A Figura 3.11 ilustra uma comparação entre as velocidades medidas de criptografia com diferentes tipos de algoritmos. O Quadro 3.2, apresenta um resumo do que foi esboçado na Figura 3.11. Os algoritmos de criptografia considerados são: o AES (com chaves de 128 e 256 bits), DES, Triple DES, RC4 (com chaves de 256 bits) e Blowfish (com chave de 256 bits). [28]

A Figura 3.11 ilustra o tempo necessário para criptografar blocos de 16 bytes de dados para cada um dos algoritmos mencionados.

Analisando o gráfico da Figura 3.11, verificamos que, dos algoritmos comparados, o RC4 com chave de 256 bits foi o que obteve melhor desempenho, ou seja, criptografou o maior número de blocos no menor tempo em relação aos demais.

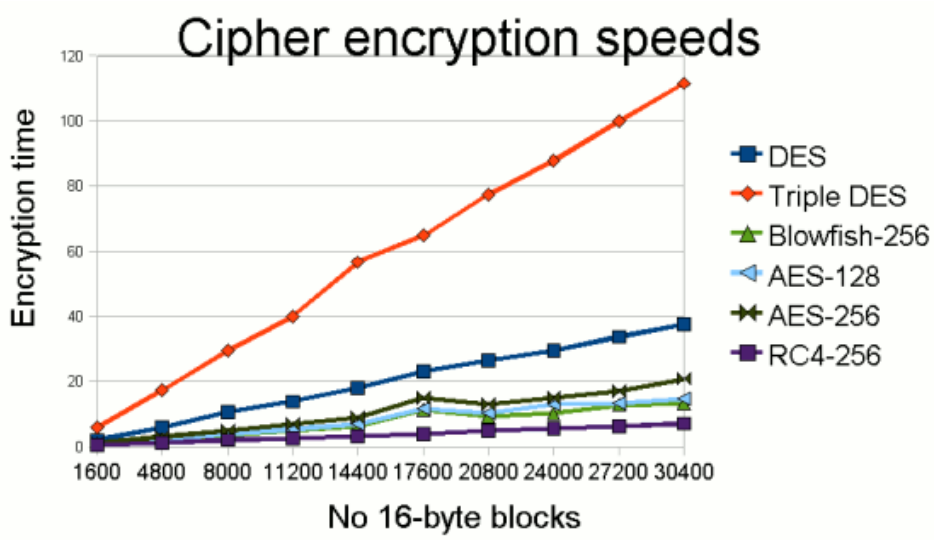

Figura 3.11 - Comparativo de algoritmos de criptografia simétrica numa escala de tempo [28]

No Quadro 3.2 é feito um resumo comparativo dos diversos algoritmos em termos de velocidade de processamento e tamanho da chave, assim quanto maior a velocidade, menor o uso dos recursos do palm.

Quadro 3.2 – Comparativo

Algoritmo	**Tamanho da chave**	**Velocidade**
RC4	40-1024	Muito Rápido
BlowFish	128-448	Rápido
AES	128, 192, 256	Rápido
DES	56	Lento
3DES	112/168	Muito lento

3.5 Análise comparativa entre os algoritmos RC4 e AES

3.5.1 Performance da Encriptação

A Figura 3.12, ilustra uma comparação entre as performances dos algoritmos de criptografia AES e RC4. As criptografias foram realizadas com diferentes tamanhos de pacotes de dados.

Analisando o gráfico podemos perceber que os resultados apresentados pelo AES mostram que não há diferenças significativas de performance à medida que se aumenta o tamanho dos blocos de dados, mesmo com a variação no tamanho das chaves. Já no algoritmo RC4, à medida que se aumenta o tamanho

dos pacotes, a performance fica muito melhor do que aquela verificada no AES, independente do tamanho das chaves. Isso mostra que o RC4 é mais eficiente que o AES à medida que se aumenta o tamanho dos blocos de dados. Mesmo utilizando-se diferentes tamanhos de chave, o AES apresenta pequenas diferenças na melhoria no desempenho, enquanto que, no RC4, o melhor desempenho independe do tamanho da chave. Portanto, considerando esse contexto e, também, que as chaves maiores proporcionam maiores segurança dos dados é preferível usar o RC4 com chave de comprimento maior para assegurar maior confiabilidade aos dados. [11]

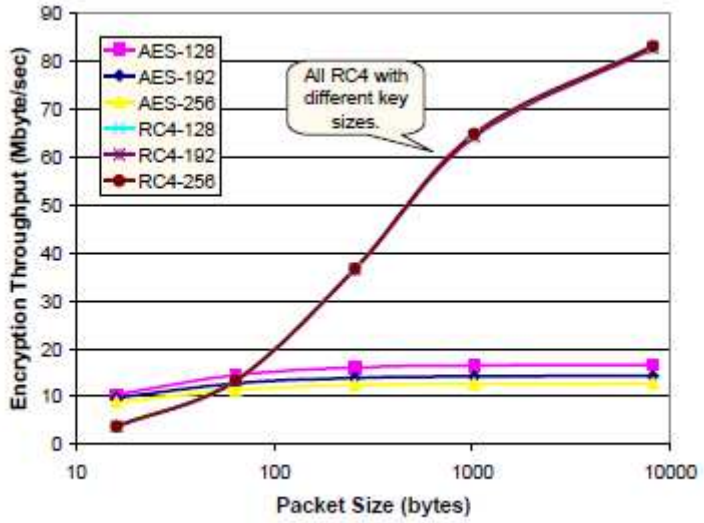

Figura 3.12 - Comparativo de criptografia utilizando os algoritmos AES e RC4 com diferentes tamanhos de chaves [10]

3.5.2 Utilização do processador

Na Figura 3.13, vê-se o desempenho dos algoritmos AES e do RC4 em termos de utilização do processador. O AES apresenta pequenas variações para

menos no tempo de processamento, à medida que se aumentam o tamanho dos pacotes de dados e diminui-se o tamanho das chaves utilizadas. Já o RC4 apresenta alto consumo de processador para pequenos blocos de dados – inferiores a 100 bytes. À medida que o tamanho dos blocos de dados aumentam verifica-se uma diminuição expressiva no consumo de processamento, apresentando melhor performance do que o AES a partir de tamanhos próximos e acima dos 100 bytes.

O RC4 opera com menos tempo de processamento para blocos de dados com tamanhos acima de 100 bytes, reduzindo o processamento quando codifica grande blocos de dados. [10]

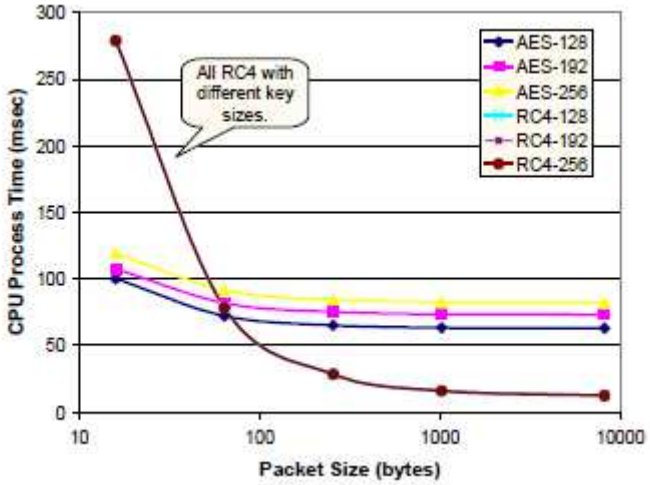

Figura 3.13 - Tempo de processamento da CPU, com diferentes tamanhos de chaves [10]

5.5.3 Consumo de Energia

A Figura 3.14, ilustra o consumo de energia de cada um dos algoritmos de criptografia para diferentes tamanhos de bloco. É mostrado que o AES consome tão pouco, cerca de três vezes menos energia do que a criptografia RC4 para pequenos blocos de dados. Já em contrapartida, o RC4 consome menos energia

que o AES para blocos de dados com tamanhos próximos ou acima de 100 bytes. [10]

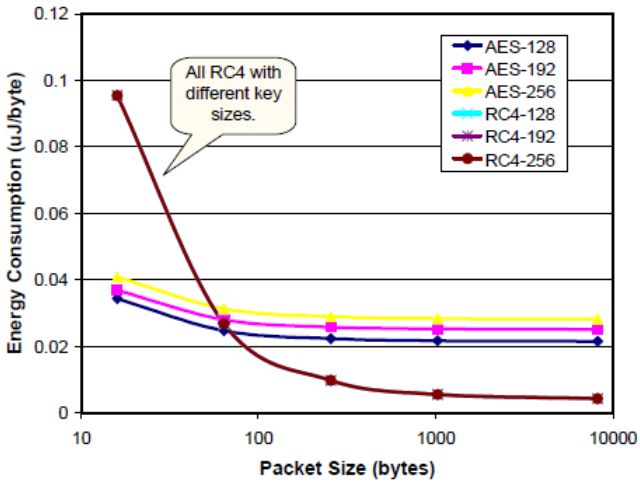

Figura 3.14 - O consumo de energia do AES e RC4 com diferentes tamanhos de chaves [10]

3.5.4 Criptografia com diferentes tamanhos de chaves

Na Figura 3.15, vê-se que o consumo de energia de cada um dos algoritmos em função dos diferentes tamanhos de chaves, que são utilizadas para encriptar pacotes com até 1 MBytes. Percebemos que, com o aumento no tamanho da chave utilizada pelo do AES, aumenta também o consumo de energia. Entretanto, a variação no tamanho da chave no RC4 não tem qualquer efeito sobre o consumo de energia. [10]

A análise do quadro sugere, como melhor opção, o uso do algoritmo RC4 com chave de 256 bits para criptografar blocos de dados com tamanho de 1024 bytes.

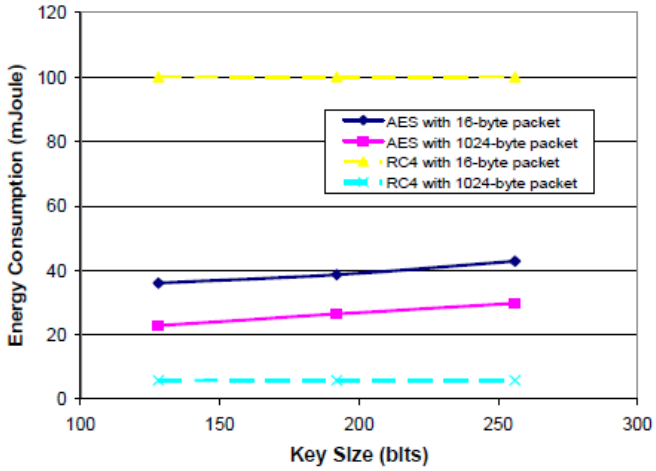

Figura 3.15 – Consumo de energia conforme os diferentes tamanhos de chave [10]

Os resultados das pesquisas acima foram obtidos utilizando-se um laptop com um processador Pentium III 700 MHz. Também foi utilizado o mesmo laptop, nessa experiência, para criptografar um arquivo com tamanho de 5,5 MBytes com os algoritmos de criptografia RC4 e o AES. [10]

Considerando as necessidades do negócio e as limitações de recursos, o RC4, pelas suas características, mostrou-se como a melhor opção dentre os demais algoritmos comparados.

Os tópicos abordados em 3.6, 3.7 e 3.8, a seguir, tratam das tecnologias de segurança, de otimização, de escalabilidade e de praticidade, que estão em uso na empresa citada no estudo de caso e, também, adotadas neste projeto como um meio de retratar uma situação, a mais próxima da realidade.

3.6 Segurança em Redes Wi Fi

O uso das redes sem fio (Wireless) vem aumentando significativamente, resultando em um impacto importante na vida das pessoas. Tanto em longa (telefones celulares), média (Wireless LAN) ou curta distâncias (Bluetooth), as redes sem fio facilitam a vida das pessoas. No entanto, existem novos riscos. Pois se nas redes com fio um hacker tinha de ter pelo menos o acesso a um ponto de rede para acessar os pacotes que trafegam por ela, com as redes sem fio, isso não é necessário. A ausência de um meio físico para conexão a uma rede pode ser um facilitador de acesso, pois uma vez dentro da área de cobertura de uma rede WiFi, o hacker, mais facilmente, poderá acessar pacotes que trafegam na rede. [14]

3.6.1. Wireless Local Access Network (WLAN)

As redes WLAN´s, além de serem utilizadas em empresas, hoje elas também podem ser vistas por toda parte, como, por exemplo: em hotéis, centro de convenções, aeroportos, bares e restaurantes, faculdades e escolas dentre outros locais. As WLAN´s, representam uma série de benefícios, tais como: [14]

1. Mobilidade dos usuários
2. Instalação rápida, sem necessidade de infraestrutura;
3. Flexibilidade: possibilidade de criar WLANS temporárias e específicas, como por exemplo, em eventos;
4. Escalabilidade.

3.6.2. Wired Equivalent Privacy (WEP)

O WEP é um padrão de criptografia de dados para redes wireless. Ele usa uma chave secreta que é compartilhada entre a estação wireless e o ponto de acesso. Todos os dados enviados e recebidos pela estação podem ser cifrados com essa chave compartilhada. O algoritmo de criptografia usado pelo WEP é o RC4, com chaves que podem variar entre 40 a 128 bits. [14]

O WEP possui algumas falhas de projeto que envolvem o uso de chave estática, falta de autenticação mútua, o uso de criptografia menos segura, dentre outras.

Diante das falhas de segurança do WEP, abaixo estão alguns cuidados que devem ser considerados na configuração de um ponto de acesso e também em relação ao uso de redes sem fio:

1. Prover limitação no local e segurança física para os pontos de acesso;
2. Desligar o ponto de acesso quando ele não estiver em uso como, por exemplo, finais de semana: [14]
3. Identificar quem pode utilizar a WLAN na organização;
4. Mudar constantemente a senha padrão de administrador do equipamento;
5. Usar a configuração de criptografia apropriada;
6. Usar o controle de acesso via MAC;
7. Mudar o canal padrão;

Então mesmo utilizando algum tipo de segurança, como exemplo a WEP, na rede WiFi interna da empresa, não é possível garantir que o GTO esteja utilizando alguma criptografia para transmitir os dados em conexões externas.

3.7 Firewall

Firewall pode ser definido como um sistema ou um conjunto de mecanismos e aplicações, podendo ser hardware e/ou software, que atua como defesa de uma rede, e reforça a política de segurança entre uma rede privada e a internet, controlando todo o tráfego de entrada e saída. O firewall também pode ser usado para modificar e monitorar o tráfego da rede, fazer NAT, redirecionamento de pacotes, marcação de pacotes, modificar a prioridade de pacotes que chegam e/ou saem da rede, contagem de bytes, dividir tráfego entre máquinas, fazer balanceamento de links de internet, criar proteções anti-spoofing e etc. A Figura 3.16 ilustra a representação de um Firewall. [14]

Basicamente existem dois tipos de firewalls: [14]

a) Nível de aplicação: este tipo de firewall analisa o conteúdo do pacote para tomar suas decisões de filtragem. Firewalls deste tipo são mais intrusivos (pois analisam o conteúdo de tudo que passa por ele) e permitem um controle relacionado ao conteúdo do tráfego. Servidores proxy, como o squid, são um exemplo deste tipo de firewall. [14]

b) Nível de pacote: este tipo de firewall toma as decisões baseadas nos parâmetros do pacote, como porta/endereço de origem/destino, estado da conexão, e outros

parâmetros do pacote. O firewall pode negar o pacote (DROP) ou deixar o pacote passar (ACCEPT). O iptables, ferramenta utilizada nesse projeto, é um excelente firewall que se encaixa nesta categoria. [14]

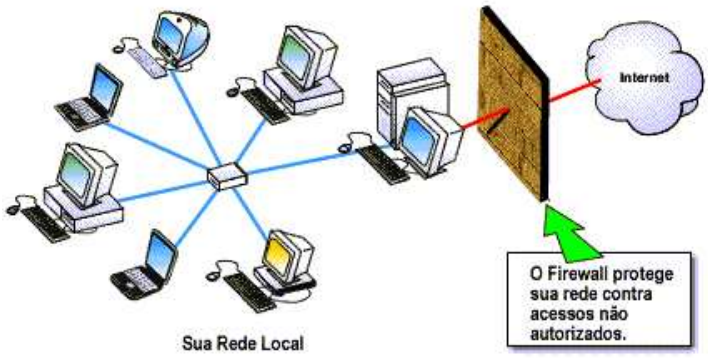

Figura 3.16 - Imagem ilustrativa representando um firewall na rede [22]

3.8 Virtualização

Virtualização é a capacidade de emularmos o hardware, ou seja, nos permite executar vários sistemas operacionais isolados, de diferentes plataformas, bem como infraestruturas de rede e de armazenamento em um único equipamento físico com uma perda mínima de performance, gerando vantagens como a economia de recursos e na redução de despesas, como: [23]

a) Ganho de espaço físico;

b) Menor gasto de energia;

c) Menor custo com manutenção de hardwares e sistemas operacionais;

d) Escalabilidade;

e) Disponibilidade/Contingência – No caso de falhas em um ambiente ou hardware, facilmente é possível transferir máquinas virtuais de um hardware para outro.

No mercado existem algumas ferramentas que possibilitam trabalhar com virtualização, sendo que as mais utilizadas são: VM Ware [24], Citrix[25] e Virtual Box. [26]

A Figura 3.17 ilustra um diagrama que exemplifica graficamente o layout da arquitetura de uma máquina virtual. Já a Figura 3.18 ilustra outro exemplo de arquitetura básica de virtualização de servidores com sistemas operacionais distintos.

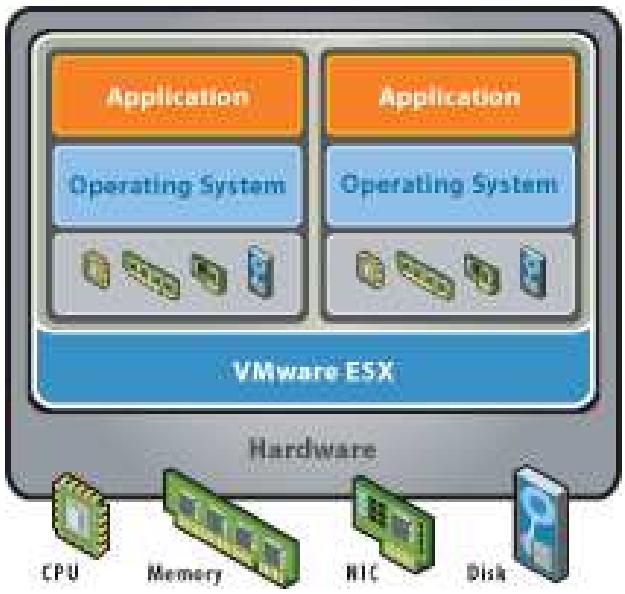

Figura 3.17 - Arquitetura de Virtualização. [24]

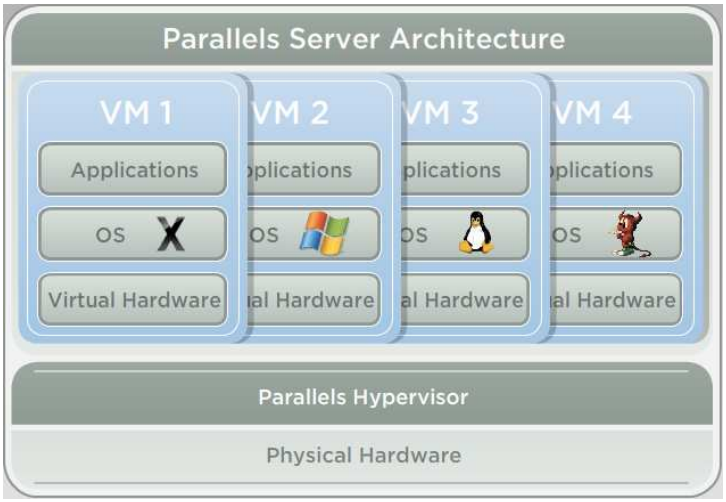

Figura 3.18 - Virtualização de servidores com diferentes sistemas operacionais. [27]

Vimos nesse capítulo alguns conceitos sobre criptografia, tais como: criptografia simétrica, assimétrica, VPN, Firewall, Virtualização e um estudo, um pouco mais aprofundado, dos algoritmos de criptografia simétrica: AES, Blowfish e RC4. A partir desse estudo foi possível concluir dentro do contexto exposto, que, um algoritmo de criptografia complexo pode sobrecarregar os recursos escassos de um dispositivo móvel. Devido aos baixos custos financeiros para implantar o RC4, sua simplicidade de implementação, por consumir menos energia dos dispositivos móveis e, por fim, por sua maior velocidade, ele foi escolhido para ser implementado.

4 - PROPOSTA DE SOLUÇÃO E MODELO

Este capítulo apresenta o protótipo desenvolvido neste projeto, tecnologia, hardware e software adotados.

4.1 Planejamento dos experimentos

Para analisar a transmissão de dados em uma rede Wifi utilizando PALM Top montou-se um ambiente, conforme ilustra a Figura 4.1, para reproduzir de maneira mais próxima à realidade da empresa do estudo de caso, considerando-se os custos financeiros, o contexto exposto na motivação e objetivos deste projeto. Foram realizados alguns comparativos teóricos quanto à velocidade do algoritmo, consumo de bateria e por fim, um teste com a aplicação desenvolvida.

4.2 Ambientes de testes

Para a realização dos testes, foram utilizados os recursos ilustrados nas Figuras 4.1 e 4.2, com o objetivo de demonstrar os benefícios da transmissão criptografada. Também, consta algumas comparações teóricas entre os algoritmos, com base em outras pesquisas, para avaliar o desempenho, segurança e consumo do dispositivo móvel, conforme apresentado no capítulo 3, Referencial teórico item 3.4 Comparação de cifras.

4.3 Transmissão criptografada

Para a transmissão dos dados entre o palm top e o servidor, foram utilizados os seguintes recursos: um notebook com sistema operacional Windows 7. No notebook, foi instalado o software VM Ware Workstation e, em seguida, criadas duas máquinas virtuais. Em cada uma delas foram instalados o sistema operacional e algumas aplicações, partes da solução.

Na máquina virtual com Windows 2003 Server foi instalado a aplicação Protheus 10 (ERP) da empresa Totvs e, também, parte da aplicação existente no palm. Também foi instalado um banco de dados SQL Server 2000 e as ferramentas para gerenciar o palm, como o Palm Desktop.

Na outra máquina virtual, foi instalado o sistema operacional Linux, distribuição CentOS versão 5.2 com o iptables e o firewall builder para controlar o tráfego de dados na rede.

Foi utilizado um roteador para fazer a conexão entre o notebook, as máquinas virtuais e o Palm top.

Foi utilizado um palm TX para efetuar o inventário e, em seguida, efetuar a transmissão criptografada das informações levantadas. No palm, existe uma aplicação desenvolvida para o controle de inventário de produtos e, também, uma aplicação para criptografar as informações que trafegam entre o palm e o servidor.

A aplicação instalada no Palm para criptografar os dados foi desenvolvida em C utilizando o algoritmo de criptografia RC4.

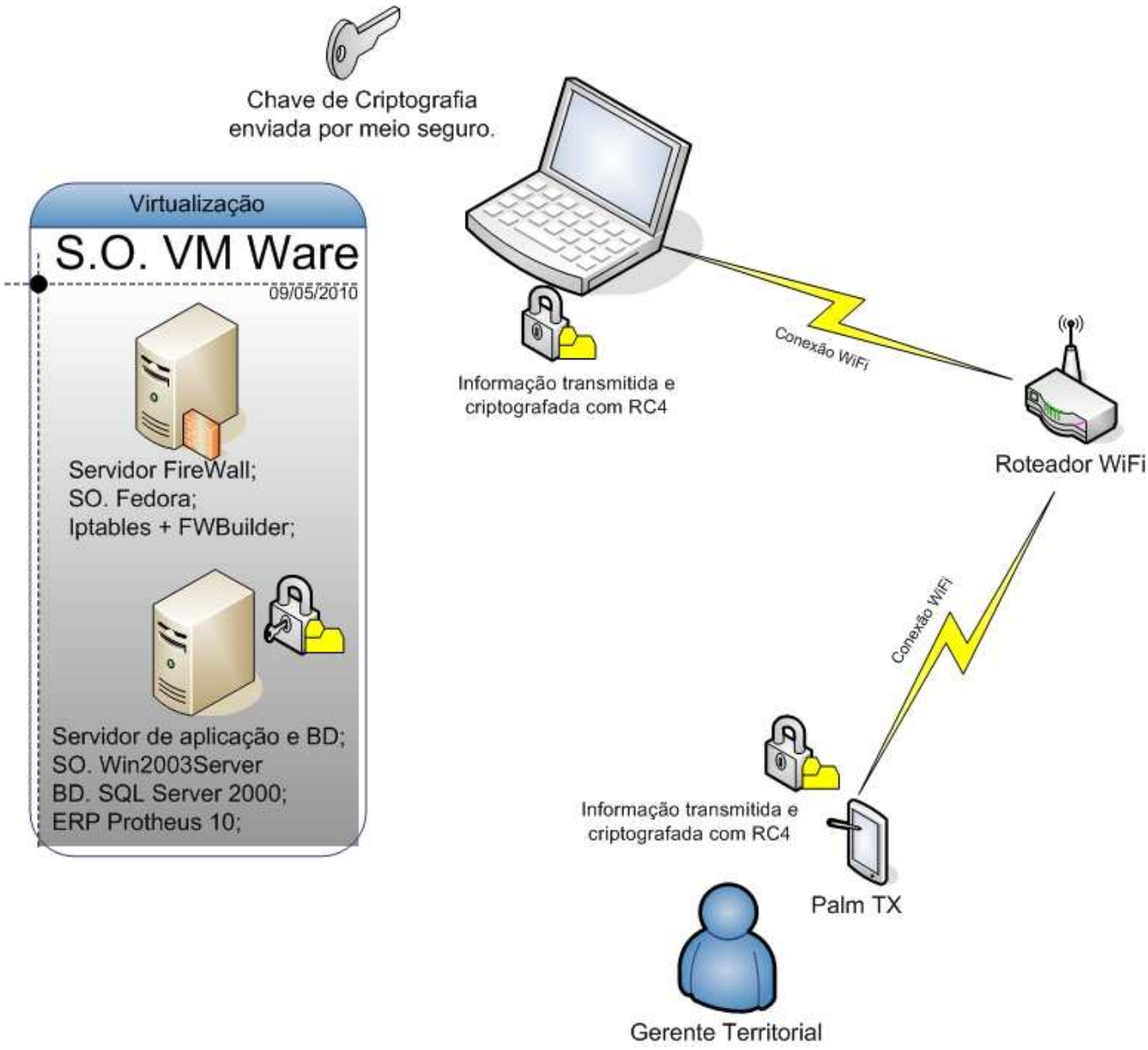

Figura 4.1 – Topologia do projeto

4.4 Hardware:

Para os testes de transmissão foram utilizados os seguintes equipamentos:

1. Um Notebook – Processador Intel Core 2 Dual T8100 2.1 GHz, 4 GB de memória RAM, HD SATA II com 500 GB 7200 RPM, Placa de rede com fio 10/100, Placa de rede sem fio G e sistema operacional Windows 7 Professional;
2. Um PALM TX - Processador Intel Intel de 312 MHz, 128MB (100MB de espaço real disponível), padrão de rede Wi-Fi 802.11b e Bluetooth 1.1 e sistema operacional Palm OS Garnet 5.4;
3. Um Roteador Wireless/Access Point Linksys, WRT54G2-BR – CISCO.

Figura 4.2 - Componentes utilizados no projeto

4.5 Software:

As informações transmitidas foram gerenciadas pelos seguintes softwares:

1. Protheus 10: Sistema de gestão integrada. Aplicação proprietária desenvolvida pela empresa TOTVS instalada na máquina virtual, juntamente com o sistema operacional Windows 2003 Server e o banco de dados SQL 2000 Server;
2. Palm Quick Install: Aplicativo da Palm responsável por instalar as aplicações no PalmTop;
3. Firewall Builder: Aplicação utilizada para gerenciar visualmente as regras do firewall em iptables;
4. Technical Force Automation (TFA): Aplicação instalada no Palm top para inventariar os estoques nas frentes de serviço;
5. Programa em C com algoritmo de criptografia RC4 para criptografar os dados transmitidos pela Wi Fi.

5 - IMPLEMENTAÇÃO DA ALTERNATIVA QUE UTILIZA O ALGORITMO RC4

Considerando as necessidades do negócio e as limitações de recursos, o RC4, pelas suas características, mostrou-se como a melhor opção dentre os demais algoritmos comparados, o que justifica a sua implementação.

Neste capítulo é dada a explicação do código fonte do algoritmo RC4, que implementa a alternativa escolhida de criptografia simétrica. Apresenta, também, as interfaces, na seqüência em que são chamados, e respectivos códigos fontes das aplicações utilizadas no palm, citadas como estudo de caso.

5.1 Exemplo de criptografia simétrica RC4

O código fonte a seguir e as interfaces ilustradas pelas Figuras de 5.1 a 5.7 referem-se ao exemplo que permite visualizar o resultado da criptografia, utilizando o algoritmo RC4, a partir da entrada de um texto qualquer e tendo como resposta, o resultado da criptografia. Vale ressaltar que o algoritmo de criptografia RC4 é o mesmo utilizado tanto para criptografar quanto para decriptografar o texto informado, pois ao entrar com um texto “em claro” o algoritmo retorna o texto criptografado; e, ao entrar com o texto criptografado retornará o texto “em claro” correspondente.

Código fonte do programa-exemplo:

```
#include "eADVPL.ch"

/*****************************************************************************/
/* Funcao: Crypt                                                             */
/* Programa para Demonstração do RC4                                         */
/*****************************************************************************/
Function Crypt()
Local oDlg
Local oMnu
Local oItem, oBtn
Local oGet, oGetKey
Local cGet := ""
Local cKey := ""

DEFINE DIALOG oDlg TITLE "Criptografia - RC4"
ADD MENUBAR oMnu CAPTION "Opções"                          OF oDlg
ADD MENUITEM oItem CAPTION "Sair" ACTION CloseDialog()     OF oMnu

@ 15,02 TO 158,158 CAPTION ""                              OF oDlg //"Controles Gráficos"
@ 50,10 SAY "Texto:"                                       OF oDlg //"Objetos Get:"
@ 50,50 GET oGet VAR cGet                                  OF oDlg
@ 100,10 SAY "Chave:"                                      OF oDlg //"Objetos Get:"
@ 100,50 GET oGetKey VAR cKey                              OF oDlg
@ 145,50 BUTTON oBtn CAPTION "Executar" ACTION MakeCrypt(cGet, cKey, oGet)      OF oDlg
//"Fechar a Janela"

ACTIVATE DIALOG oDlg

Return nil

Function MakeCrypt(cData, cKey, oGet)
Local cCrData := ""

If Empty(cData)
        MsgAlert("Favor preencher o campo Texto.")
EndIf
If Empty(cKey)
        MsgAlert("Favor preencher o campo Chave.")
EndIf
//Chamada da função para criptografia do texto
cCrData := CryptoData(cData, cKey)
SetText(oGet, cCrData)
Return .T.
```

5.2 Interfaces de interação do usuário com o programa-exemplo

A Figura 5.1 ilustra a tela inicial do programa-exemplo que faz a criptografia utilizando o algoritmo simétrico RC4. Na seqüência, a Figura 5.2 ilustra o texto informado, “Projeto CEUB 2010”, a ser criptografado. A Figura 5.3 ilustra a mesma interface anterior acrescida da chave de cifragem, “South*África-2010!”. A partir da Figura 5.3, ao clicar no botão executar, o programa-exemplo cifra o conteúdo informado e resulta na saída criptografada, conforme ilustra a Figura 5.4. E, por fim, a partir do que ilustra a Figura 5.4, se clicar no botão executar, o programa-exemplo retorna o texto original conforme ilustrado na Figura 5.5.

Figura 5.1 - Imagem da interface inicial do programa-exemplo

Figura 5.2 - Imagem da interface com o texto a ser criptografado

Figura 5.3 - Imagem da interface com o texto e chave informados

Figura 5.4 - Imagem da interface com o resultado da criptografia

Figura 5.5 - Imagem da interface com o retorno do texto original

5.3 Interfaces de interação do GTO com a aplicação de inventário

As Figuras de 5.6 a 5.19 ilustram o procedimento de inventário dos produtos num cliente pelo Gerente Territorial Operacional (GTO), que é realizado mensalmente. O código fonte dessa aplicação consta no Anexo II, que por sua vez, faz uso da função de criptografia descrita no Anexo I. Segue um breve relato dos passos executados pelo GTO para a realização do inventário. Com o uso do palm, o GTO se dirige ao cliente na sua área de cobertura e faz o levantamento das quantidades existentes em estoque dos produtos, inserindo-as no palm. Para a simplificação do trabalho, a aplicação lista, no palm, os clientes que estão sob a supervisão do GTO. Após a seleção de um cliente, a aplicação mostra a lista dos produtos do cliente selecionado para a realização do inventário. A partir desse momento, o GTO seleciona cada produto e confirma a quantidade inventariada. Ao final, o GTO aciona a função que gera o pedido a partir do inventário, o qual consiste da diferença entre as quantidades dos produtos contratadas e aquelas inventariadas. No ato da geração do pedido, a aplicação também realiza a sua criptografia. Por fim, o GTO aciona a função de sincronismo do palm com o servidor e transmite o conjunto de dados do pedido criptografado. Uma vez no servidor, o conjunto de dados é decriptografado utilizando o mesmo algoritmo descrito no anexo I, de modo análogo ao que é mostrado no programa-exemplo.

Figura 5.6 – Imagem da interface inicial da aplicação

Figura 5.7 – Imagem da interface de seleção da opção de inventário

Figura 5.8 – Imagem da interface de menu de clientes gerenciados, para seleção

Figura 5.9 – Imagem da interface da lista de produtos para inventário do cliente selecionado

Figura 5.10 – Imagem da interface para entrada da quantidade inventariada do produto

Figura 5.11 – Imagem da interface do produto inventariado

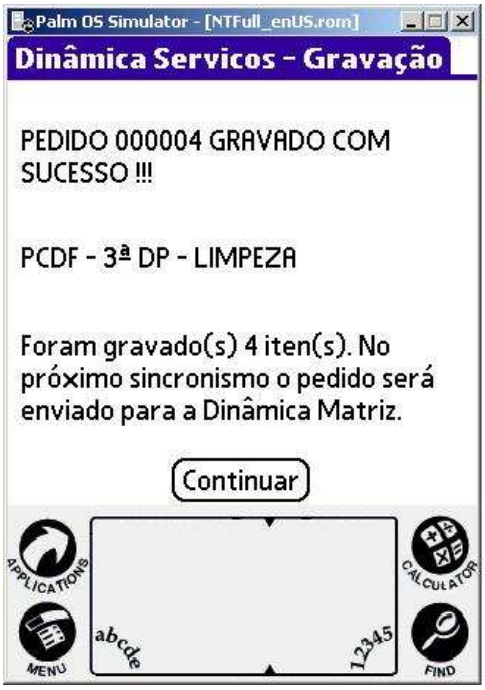

Figura 5.12 – Imagem da interface do inventário/pedido finalizado

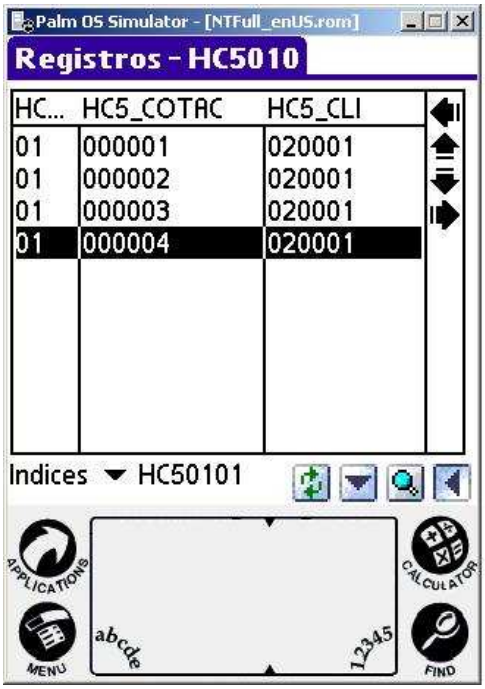

Figura 5.13 – Imagem da interface do registro do pedido 000004 salvo no palm

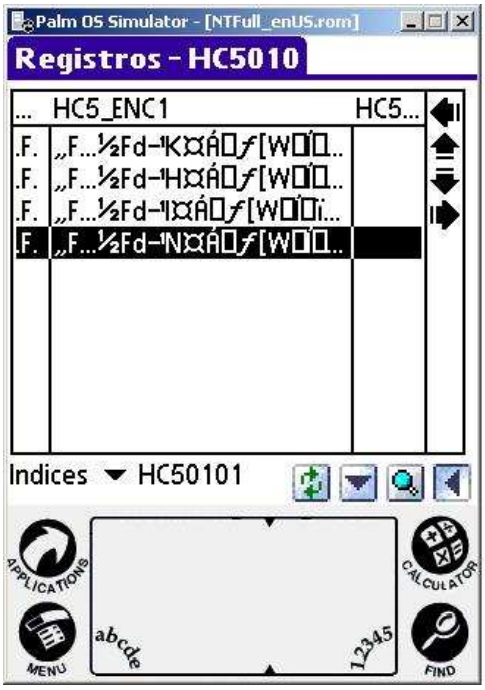

Figura 5.14 – Imagem da interface do pedido criptografado

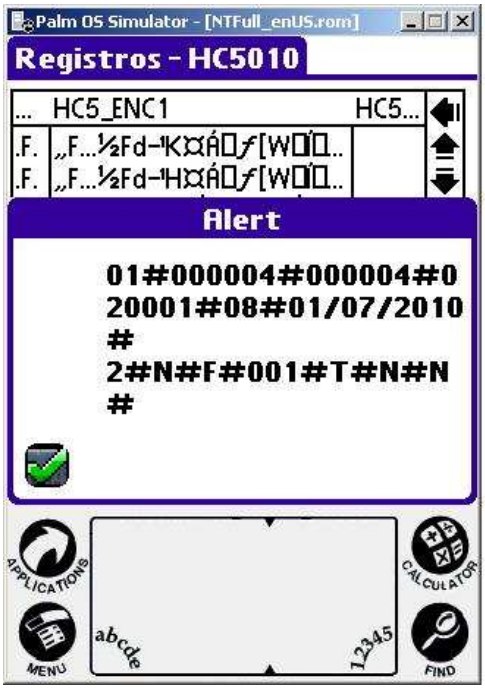

Figura 5.15 – Imagem da interface dos dados do pedido em claro

Figura 5.16 – Imagem da interface do menu para seleção da opção de sincronismo

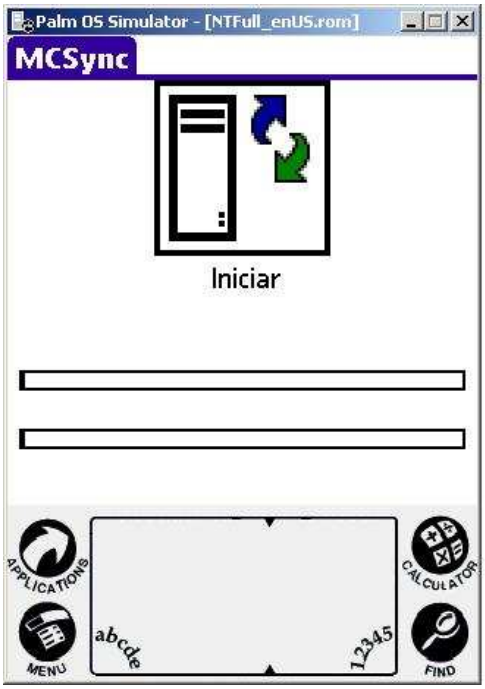

Figura 5.17 – Imagem da interface de sincronismo com o servidor

Figura 5.18 – Imagem da interface de inicio da transmissão

Figura 5.19 – Imagem da interface de transmissão concluída

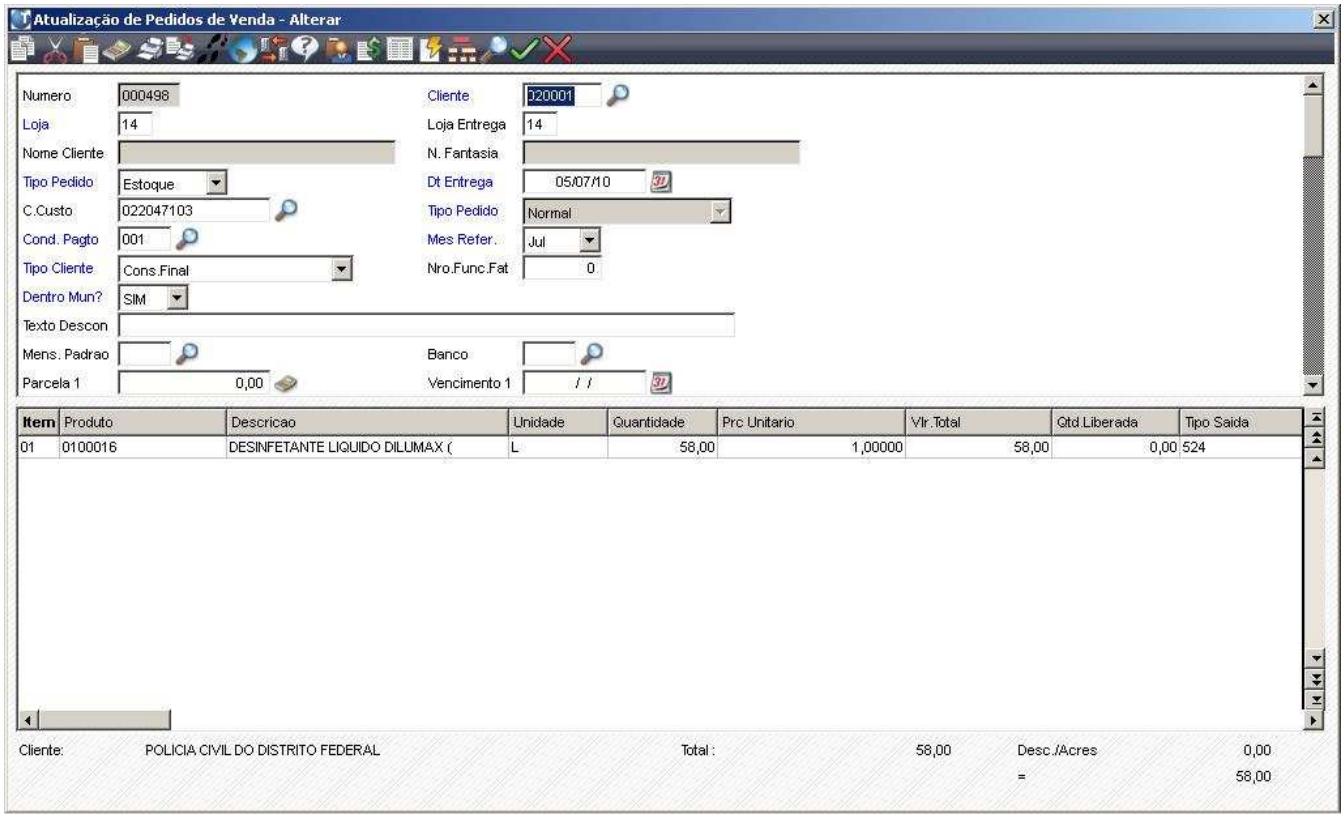

Figura 5.20 – Imagem do pedido gerado e transmitido pelo palm no servidor

5.4 Resultados Obtidos

A aplicação apresenta resultados satisfatórios atendendo as expectativas. O uso do algoritmo RC4 para cifrar o conteúdo tem bom desempenho e utiliza poucos recursos do dispositivo móvel. O resultado, para o usuário final, é transparente e faz com que o mesmo tenha uma produção similar quando comparada à utilização do palm sem criptografia, mas com segurança, a um bom custo e com código fonte aberto, o que torna a aplicação mais segura uma vez que soluções proprietárias podem, de maneira voluntária ou involuntária, conter brechas de segurança.

6 – CONCLUSÃO

A escolha do algoritmo de criptografia RC4, sobre as demais possibilidades apresentadas neste estudo de caso comparativo, mostrou ser a mais apropriada devido às condições favoráveis que se apresentaram, quais sejam: o uso com aproveitamento do algoritmo RC4 em situações em que mostra melhor desempenho, aliado às características do negócio em si, no qual está sendo aplicado. Essas condições podem ser descritas como sendo a melhor performance do algoritmo para transmissão de dados, cuja média do tamanho dos blocos por transação se situa na faixa de menor consumo de processamento do algoritmo permitindo o uso de chaves de tamanhos maiores, sem o comprometimento da performance, assegurando maior autenticidade e confiabilidade às transmissões, as quais são feitas a partir de dispositivos portáteis sem fio, que pelas suas características, possui limitações de memória, de capacidade de processamento e de armazenamento de energia. O baixo uso de processador nessas condições favorecem o menor consumo de energia e a transmissão mais segura dos dados pelo uso de criptografia utilizando tamanho de chaves maiores. Isso tudo permitiu conjugar os vários fatores de modo a aliar as virtudes do algoritmo na solução das funcionalidades necessárias à inovação do negócio, no que diz respeito à transmissão de dados para a reposição de estoques inventariados junto aos clientes da empresa.

Após avaliar o desempenho dos algoritmos RC4 e o AES, foi possível perceber que para grandes pacotes o RC4 mostrou ser mais rápido e também

apresentou menor consumo de energia. Já o AES, foi mais eficiente do que o RC4 para pacotes menores. A partir desses resultados, para futuros projetos, fica evidente que é possível economizar energia e garantir melhor performance para prover a criptografia de pacotes com qualquer tamanho utilizando um sistema de segurança híbrido com o RC4 e o AES.

BIBLIOGRAFIA

[01] STALLING, Willian, **Criptografia e Segurança de Redes**: Princípios E Práticas 4. Ed. Prentice Hall Brasil, 2007;

[02] Palm – Palm OS Product
Disponível em:
http://software.palm.com/us/html/display_palm_product.jsp?navCategoryId=cat90016&id=prod6502013
Acessado em: 07 de abril de 2010;

[03] Palm – Palm OS Product
Disponível em:
http://software.palm.com/us/html/display_palm_product.jsp?id=prod2430707
Acessado em: 08 de abril de 2010;

[04] Antha Soft - AnthaSoft Security on the move
Disponível em: http://www.anthasoft.com/anthavpn-virtual-private-network.php
Acessado em: 12 abril de 2010;

[05] MORENO, Edward David; PEREIRA, Fábio Dacêncio; BARROS, Rodolfo, **Criptografia em Software e Hardware** Ed. Novatec Editora Ltda, 2006.

[06] DUARTE, Otto, Redes de computadores, GTA / URFJ, 2005.
Disponível em: http://www.gta.ufrj.br/grad/05_2/
Acessado em: 18 de abril de 2010;

[07] COSTA, Celso José; SILVA Luiz Manoel, Apostila - **Curso de Criptografia e segurança em redes**, Unidade 2 - Universidade Federal Fluminense – UFF, 2006;

[08] ApConcursos
Disponível em: http://apconcursos.blogspot.com/2007_08_01_archive.html
Acessado em: 16 de abril de 2010;

[09] N. Ruangchaijatupon, P. Krishnamuerthy,
Encryption and Power Consumption in Wireless LANs, The Third IEEE Workshop on Wireless LANs, September 27-28, 2001, Newton, Massachusetts;

[10] P. Prasithsangaree and P. Krishnamurthy, **Analysis of Energy Consumption of RC4 and AES Algorithms in Wireless LANs**, Telecommunications Program. University of Pittsburgh;

[11] IEEE P802 working group, P802.11i Draft Supplement to Standard for Telecommunications and Information Exchange Between Systems- LAN/MAN Specific Requirements - Part 11: Wireless Medium Access Control (MAC) and physical layer (PHY) specifications: Specification for Enhanced Security, November 2002;

[12] PKI – Public key infrastructure
Disponível em: http://www.gta.ufrj.br/grad/07_2/delio/Criptografiasimtrica.html
Acessado em: maio de 2010;

[13] B. Schneier, **Applied Cryptography**, John Wiley & Sons, Inc., 1996;

[14] NAKAMURA, Emilio Tissato; GEUS, Paulo Lício de **Segurança de Redes:** em Ambientes Coorporativos – Ed. Novatec, 2007;

[15] Vaquelli
Disponível em: http://www.vaquelli.com.br/wp-content/uploads/2009/11/vpn.jpg
Acessado em: 24 de maio de 2010

[16] Softsys
Disponível em: http://www.softsys.com.br/images/vpn1.jpg
acessado em: 26 de maio de 2010;

[17] Peplink
Disponível em: http://www.peplink.com/image/knowledgebase_illustrations/vpn-diagram.jpg
Acessado em: 26 de maio 2010;

[18] Wikipedia – Imagem algoritmo AES
Disponível em: http://pt.wikipedia.org/wiki/Advanced_Encryption_Standard
Acessado em: 16 de maio de 2010;

[19] CodProjeect
Disponível em: http://www.codeproject.com/KB/cs/SecuringData/AES_structure.png
Acessado em:16 de maio de 2010;

[20] COSTA, Celso José; SILVA Luiz Manoel, Apostila - **Curso de Criptografia e segurança em redes**, Unidade 3 - Universidade Federal Fluminense – UFF, 2006;

[21] Wikipedia – Imagem Algoritmo RC4
Disponível em: http://pt.wikipedia.org/wiki/RC4
Acessado em: 12 de maio de 2010;

[22] GTA / URFJ
Disponível em:http://www.gta.ufrj.br/grad/08_1/firewall/firewall_1.jpg
Acessado em: 23 de abril de 2010;

[23] MATTOS, Diogo Menezes Ferrazani, GTA/POLI – UFRJ
Disponível em: http://www.gta.ufrj.br/grad/08_1/virtual/artigo.pdf
Acessado em: 19 de maio de 2010;

[24] VM Ware
Disponível em: http://www.vmware.com/br
Acessado em: 28 de abril de 2010;

[25] Citrix
Disponível em: http://www.citrix.com.br
Acessado em: 28 de abril de 2010;

[26] Virtual Box
Disponível em: http://www.virtualbox.org
Acessado em: 28 de abril de 2010;

[27] Macmagazine
Disponível em: http://macmagazine.uol.com.br/wp-content/uploads/2010/02/26-paralles-server.png
Acessado em: 19 de junho de 2010;

[28] Javamex
Disponível em: http://www.javamex.com/tutorials/cryptography/ciphers.shtml
Acessado em: 23 de junho de 2010;

[29] A.K. Lenstra; E. R. Verheul, **Selecting Cryptographic Key sizes**, Journal of Cryptology, vol. 14, no. 4, pp. 255-293, 2001;

[30] B. Schneier; D. Whiting, **Fast Software Encryption: Designing Encryption Algorithms for Optimal Software Speed on the {Intel Pentium} Processor**, Lecture Notes in Computer Science, vol. 1267, pages 242-259, 1997;

Anexo I
Código fonte do RC4

```
/* Declaração de variáveis */

unsigned char S[256];
unsigned int i, j;

 /* KSA */

/* Prepara a chave que será utilizada na criptografia. A função rc4_init recebe como parâmetros duas variáveis: a chave que é utilizada para criptografia e o seu tamanho. E, faz:
1. Preenche o array S, com tamanho de 256 posições, atribuindo a cada posição o número correspondente à sua posição relativa, iniciando em 0 e terminando em 255.
2. Permuta os valores do array e mistura-os com a chave que foi passada por parâmetro. (Vide Fig. 3.10)

void rc4_init(unsigned char *key, unsigned int key_length){
        for (i = 0; i < 256; i++)
                S[i] = i;
        for (i = j = 0; i < 256; i++) {
                unsigned char temp;
                j = (j + key[i % key_length] + S[i]) & 255;
                temp = S[i];
                S[i] = S[j];
                S[j] = temp;
        }
        i = j = 0;
}

 /* PRGA */

/*A função rc4_output carrega o array S com o resultado da operação de criptografia ou decriptografia sobre o dado informado via parâmetro. */

unsigned char rc4_output() {
        unsigned char temp;

        i = (i + 1) & 255;
        j = (j + S[i]) & 255;

        temp = S[j];
```

```
        S[j] = S[i];
        S[i] = temp;

        return S[(temp + S[j]) & 255];
}

/* A função EncRC4 aciona a função rc4_init para iniciar a operação, efetivamente realiza a
criptografica e aciona a função rc4_output para gerar o resultado */

int EncRC4(unsigned char* cData, unsigned char* cKey, char* strenc){
        char ch[3];
        int y;
        int keylen = xstrlen((char*)cKey);
        int datalen = xstrlen((char*)cData);
        rc4_init(cKey, keylen);
        for (y = 0; y < datalen; y++)
        {
                xsprintf(ch, "%c", cData[y] ^ rc4_output());
                xstrcat(strenc, ch);
        }
        return 0;
}
```

ANEXO II

Código fonte do programa de inventário no palm

```
#INCLUDE "eADVPL.ch"
/*-----------------------------------------------------------------------------------
  |Funçao | Dinam
  -----------------------------------------------------------------------------------
  |Descr. | Função principal
  -----------------------------------------------------------------------------------
  |Param. | Nao de aplica
  -----------------------------------------------------------------------------------*/
Function Dinam()
Local oMnu, oItem, oSayFile, oMeterFiles
Local cMeter      := ""
Local cSerial     := ""
Local nMeterFiles := 0
Local aEmp                := {}
Local oDlg_Main
Local oLogo
Local oBtnExit
Local cCliente := ""
Local cProduto := ""

// Variaveis Públicas
Public cEmpresa
Public cFilial
Public cSufixo
Public cSupervisor
Public cCodSuperv
Public cProPed
Public dDataBase := Date()

// Conficuracoes de Ambiente
SET CENTURY ON
SET DATE BRITISH
SET DELETED ON

// Definicao do Dialogo Principal
DEFINE DIALOG oDlg_Main       TITLE    "Dinâmica Servicos" COLOR CLR_WHITE, CLR_HBLUE
ADD MENUBAR           oMnu            CAPTION "Menu Principal"
            OF oDlg_Main
ADD MENUITEM          oItem           CAPTION "Consulta Clientes" ACTION Clientes() OF oMnu
ADD MENUITEM          oItem           CAPTION "Inventario"              ACTION CabContr() OF oMnu
ADD MENUITEM          oItem           CAPTION "Sync"                                    ACTION
InitSync(oSayFile,oMeterFiles, nMeterFiles)      OF oMnu
```

```
ADD MENUITEM oItem CAPTION "Sair" ACTION
CloseDialog() OF oMnu

// Logotipo
@ 20,10 BUTTON oLogo CAPTION DINAMICALOGO SYMBOL OF oDlg_Main

@ 120,20 GET oSayFile VAR cMeter READONLY NO UNDERLINE SIZE 120,15 OF oDlg_Main
@ 135,20 METER oMeterFiles SIZE 120, 5 FROM 0 TO 100 OF oDlg_Main

@ 143,010 BUTTON oBtnExit CAPTION BTN_BITMAP_EXIT SYMBOL ACTION CloseDialog() OF
oDlg_Main
@ 150,112 SAY "Versão: " + GetVer() OF oDlg_Main

// Verifica arquivo de Empresas
If !OpenEmp(aEmp)
        InitSync()
        Return .F.
EndIf

// Abre arquivos de Dados
OpenFiles(oSayFile, oMeterFiles, nMeterFiles)
ACTIVATE DIALOG oDlg_Main

Return

/*-----------------------------------------------------------------------------
  |Funçao | CabContr
  -----------------------------------------------------------------------------
  |Descr. | Rotina de selecao do Contrato (Cliente/Loja) que sera Inventariado
  -----------------------------------------------------------------------------
  |Param. | Nao de aplica
  -----------------------------------------------------------------------------*/
Function CabContr()
Local oDlg_Cab
Local oBrwCli, oCol
Local oMnu, oItem, oBtn
Local oCliente
Local aItemCli := {}
Local cCliente := ""
Local cHA3Tbl := "HA3" + cEmpresa + "0"

// Verifica se existe a Tabela
/*
If !File(cHA3Tbl)
        MsgStop("Tabela de Vendedores do HandHeld " + cHA3Tbl + " não encontrada!","Aviso")
        Return Nil
EndIf
*/
// Abrir a Tabela de Cantratos de Vendedor
```

```
If Select("HA3") = 0
	MsgStop("Tabela de vendedor (HA3) não está aberta.","Aviso")
	Return Nil
EndIf

dbSetOrder(1)
HA3->(dbGoTop())
cSupervisor := HA3->HA3_NREDUZ
cCodSuperv  := HA3->HA3_COD
cProPed     := HA3->HA3_PROPED

// Seleciona os Contratos (Clientes/Loja) do Supervisor
dbSelectArea("HDA")
dbSetOrder(1)
HDA->(dbGoTop())
If HDA->(Reccount()) = 0
	MsgStop("Nenhum contrato encontrado.","Aviso")
	Return Nil
EndIf

DEFINE DIALOG oDlg_Cab TITLE "Clientes"

ADD MENUBAR  oMnu  CAPTION "Opções" OF oDlg_Cab
ADD MENUITEM oItem CAPTION "Sair" ACTION CloseDialog() OF oMnu

@ 000,050 SAY "Supervisor: " + cSupervisor
                        OF oDlg_Cab

@ 015,005 GET oCliente VAR cCliente READONLY MULTILINE NO UNDERLINE SIZE 155,25 OF
oDlg_Cab

@ 040,005 BROWSE oBrwCli SIZE 150,100 ON CLICK Show_Cli(oBrwCli,aItemCli,oCliente) OF
oDlg_Cab

SET BROWSE oBrwCli ARRAY aItemCli
ADD COLUMN oCol TO oBrwCli ARRAY ELEMENT 1 HEADER "Cliente"     WIDTH 150 //Descricao do
Cliente

@ 143,005 BUTTON oBtn CAPTION "Visualizar Produtos" ACTION ItemContr(@oBrwCli,@aItemCli)
	SIZE 85,13 OF oDlg_Cab
@ 143,115 BUTTON oBtn CAPTION "Retornar"                    ACTION CloseDialog(oDlg_Cab)
                                              SIZE 45,13 OF oDlg_Cab  //CloseDialog(oBrwCli)

LoadContratos(aItemCli, oBrwCli)

// Inicia com o primeiro cliente
cCliente := aItemCli[1,1]
```

```
ACTIVATE DIALOG oDlg_Cab

Return nil

Function LoadContratos(aItemCli, oBrwCli)
While !HDA->(EOF())
	HA1->(dbSetOrder(1))
	If !HA1->(dbSeek(RetFilial("HA1")+HDA->HDA_CODCLI+HDA->HDA_LOJCLI))
		MsgAlert("Cliente nao encontrado.", RetFilial("HA1")+HDA->HDA_CODCLI+HDA-
>HDA_LOJCLI)
		HDA->(dbSkip())
		Loop
	EndIf
	aAdd(aItemCli,{AllTrim(HA1->HA1_NREDUZ),HDA->HDA_NUMCTR,HDA->HDA_CODCLI,HDA-
>HDA_LOJCLI,HDA->HDA_DTVIS, .F.})
	HC5->(dbSetOrder(2))
	HC5->(dbSeek(RetFilial("HC5")+HDA->HDA_CODCLI+HDA->HDA_LOJCLI))
	While HC5->(!Eof()) .And. HDA->HDA_CODCLI = HC5->HC5_CLI .And. HDA->HDA_LOJCLI =
HC5->HC5_LOJA
		If HC5->(IsDirty())
			aItemCli[Len(aItemCli),6] := .T.
			GridSetCellColor(oBrwCli,Len(aItemCli),1,CLR_YELLOW ,CLR_BLACK)
		EndIf
		HC5->(dbSkip())
	EndDo
	HDA->(dbSkip())
EndDo
Return Nil

/*-----------------------------------------------------------------------------------
  |Funçao | ItemContr
  -----------------------------------------------------------------------------------
  |Descr. | Exibe os Itens do Contrato (Produtos) para Inventario
  -----------------------------------------------------------------------------------
  |Param. | oBrwCli : Browse com lista de contratos
			aItemCli: Array com contratos do supervisor
  -----------------------------------------------------------------------------------*/
Function ItemContr(oBrwCli,aItemCli)
Local oDlgProd, oBrwProd, oColProd, oMnuProd, oItemProd, oBtnProd, oProduto
Local aItemProd := {}
Local cTot     := 5
Local nLinha    := 0
Local lOpenHDB          := .F.
Local lOpenHC5:= .F.
Local cId      := ""
Local cNumCtr
Local cCodCli
Local cLojCli
Local nDiasVis
```

```
Local dDtMax
Local dDtMin
Local cCliente := ""
Local cProduto := ""
Local dDataVis := dDataBase
Local lHasContrato := .F.
// Selectiona a Linha do Browse e Armazena as informacoes do CLIENTE para o PEDIDO DE VENDAS
nLinha   := GridRow(oBrwCli)
cCliente := aItemCli[nLinha,1]
cNumCtr  :=      aItemCli[nLinha,2]
cCodCli  := aItemCli[nLinha,3]
cLojCli  := aItemCli[nLinha,4]
dDataVis := aItemCli[nLinha,5]
lHasContrato := aItemCli[nLinha,6]

// Define a tolerancia de dias para a visita no Cliente para mais ou para menos
dbSelectArea("HCF")
HCF->(dbSetOrder(1))

// Parametro com a quantidade de dias de tolerancia da data de visita estipulada no cronograma
If HCF->(dbSeek("MV_DIASVIS"))
        nDiasVis := AllTrim(HCF->CF_VALOR)
Else
        nDiasVis := 2
EndIf

If Empty(dDataVis)
        dDataVis := dDataBase
EndIf

//dDtMax := (dDataVis + nDiasVis)
dDtMax := (dDataVis)
//dDtMin := (dDataVis - nDiasVis)
dDtMin := (dDataVis)
//If dDataBase < dDtMin .Or. dDataBase > dDtMax
        //MsgStop("A visita esta fora da data programada para este Cliente","Inventario nao
permitido")
        //Return Nil
//EndIf

// Verifica se ja existe o Pedido de Vendas para o Cliente Selecionado
If lHasContrato
        MsgStop("Inventario ja existente para este Cliente","Inventario Existente")
        Return Nil
EndIf

//ClearStatus()

// Seleciona os Itens dos Contratos (Produtos)
```

```
dbSelectArea("HDB")
dbSetOrder(1)
HDB->(dbGoTop())
If Reccount() = 0
	MsgStop("A tabela HDB010 esta VAZIA","Aviso")
	Return Nil
EndIf

If !dbseek(RetFilial("HDB")+cNumCtr)
	MsgStop("Contrato - Filial:"+HDB->HDB_FILIAL+" - "+	HDB->HDB_NUMCTR+"nao
encontrado","Aviso")
	Return Nil
EndIf

While !HDB->(EOF()) .And. HDB->HDB_NUMCTR = cNumCtr
	aAdd(aItemProd		,{.F. ,HDB->HDB_DESCPR,HDB->HDB_UM		,0 ,HDB-
>HDB_CODPRO,HDB->HDB_QUANT})
	HDB->(dbSkip())
EndDo

// Inicia com o Primeiro produto
cProduto  := aItemProd[1,2]

DEFINE DIALOG oDlgProd TITLE "Produtos"

ADD MENUBAR  oMnuProd  CAPTION "Opcoes"                                                    OF
oDlgProd
ADD MENUITEM oItemProd CAPTION "Sair"	ACTION CloseDialog() OF oMnuProd

@ 000,055 SAY "Supervisor: "+cSupervisor OF oDlgProd

@ 015,005 SAY oProduto		PROMPT cProduto  OF oDlgProd

@ 030,005 BROWSE oBrwProd SIZE 150,105 ON CLICK Show_Prod(@oBrwProd,aItemProd,oProduto)
	OF oDlgProd
SET BROWSE oBrwProd ARRAY aItemProd
ADD COLUMN oColProd TO oBrwProd ARRAY ELEMENT 1 HEADER ""		WIDTH 10 MARK
ADD COLUMN oColProd TO oBrwProd ARRAY ELEMENT 2 HEADER "Produto" WIDTH 60
ADD COLUMN oColProd TO oBrwProd ARRAY ELEMENT 3 HEADER "UM"		WIDTH 20
ADD COLUMN oColProd TO oBrwProd ARRAY ELEMENT 4 HEADER "Qtd"		WIDTH 30

@ 145,005 BUTTON oBtnProd CAPTION "Gerar Ped."	ACTION GeraPV(oBrwProd
,aItemProd,oBrwCli,aItemCli,.F.)		SIZE 45,13 OF oDlgProd
@ 145,057 BUTTON oBtnProd CAPTION "Inventariar"	ACTION QtdInv(@oBrwProd,@aItemProd)
                                                        SIZE 50,13 OF oDlgProd
@ 145,115 BUTTON oBtnProd CAPTION "Retornar"	ACTION ConfRet()
                                                        SIZE 50,13 OF oDlgProd

ACTIVATE DIALOG oDlgProd
```

```
Return nil

/*-------------------------------------------------------------------------------------
  |Funçao | GeraPV
  -------------------------------------------------------------------------------------
  |Descr. | Gera Pedido de Vendas de Acordo com o Inventario e com o Cadastro de Contratos
  |                    |        de Parceria
  -------------------------------------------------------------------------------------
  |Param. | oBrwProd : Browse com lista de produto do contrato
                              aItemProd: Array com produtos do contrato
                              aItemCli: Array com contratos do supervisor
  -------------------------------------------------------------------------------------*/
Function GeraPV(oBrwProd,aItemProd,oBrwCli,aItemCli,lExtra)
Local oBtn
Local oDlgGravado
Local oGet
Local i                     := 0
Local y                     := 0
Local lItensNoInv   := .T.
Local nItensNoInv   := 0
Local nTotItens     := 0

// Variaveis do Browse de Contratos (Cliente/Loja)
Local nLinCli  := 0
Local cNumCtr  := ""
Local cProduto := ""
Local cCliente := ""
Local cCodCli  := ""
Local cLojCli  := ""

// Variaveis do Browse de Produtos do Contrato
Local cItem   := ""
Local cDescr  := ""
Local cUm           := ""
Local nQtdVen := 0
Local nQtdInv := 0
Local nPrUnit := 0
Local nPrcVen := 0

//Variaveis referente as Datas
Local nItem       := 0
Local nQtdPdr   := 0 // Quantidade Padrao
Local cMsg      := ""
Local cMsgItens := ""

Local cKey      := "MONOGRAFIA"
Local cData                 := ""
Local cDtEncry  := ""
```

```
// Dados do Cliente
nLinCli         :=      GridRow(oBrwCli)
cCliente        :=      aItemCli[nLinCli,1]
cNumCtr         :=      aItemCli[nLinCli,2]
cCodCli         :=      aItemCli[nLinCli,3]
cLojCli         :=      aItemCli[nLinCli,4]


// Verifica se existem itens nao inventariados
For i := 1 to Len(aItemProd)
        If aItemProd[i,1] = .F.
                lItensNoInv := .T.
                nItensNoInv++
        EndIf
Next

nTotItens := Len(aItemProd)-nItensNoInv

If !lExtra
        // Verifica se existe algum item para gerar o Pedido
        If nTotItens = 0
                MsgStop("Nenhum item inventariado, o pedido não pode ser gerado.","Inventário
Inválido")
                Return Nil
        EndIf


        // Aviso de Itens Zerados no Inventario
        If lItensNoInv
                If !MsgYesOrNo("Existem "+Str(nItensNoInv,3)+" itens não inventariados. Deseja
emitir o Pedido de Materiais?","Gerar Pedido")
                        Return Nil
                EndIf
        EndIf
EndIf

dbSelectArea("HC5")
dbSetOrder(1)
cProPed := GetProxPed()

HC5->(dbAppend())
HC5->HC5_FILIAL  := RetFilial("HC5") //"01" //"02"
cData += HC5->HC5_FILIAL + "#"
HC5->HC5_NUM    :=    cProPed
cData += HC5->HC5_NUM + "#"
HC5->HC5_COTAC          :=  cProPed
cData += HC5->HC5_COTAC + "#"
HC5->HC5_CLI    :=      cCodCli
```

```
cData += HC5->HC5_CLI + "#"
HC5->HC5_LOJA    :=     cLojCli
cData += HC5->HC5_LOJA + "#"
//HC5->HC5_VALOR :=  nValPed //Valor igual a quantidade total de itens do Pedido
HC5->HC5_EMISS          :=        Date()
cData += DTOC(HC5->HC5_EMISS) + "#"
HC5->HC5_QTDITE         := nTotItens
cData += Str(HC5->HC5_QTDITE,3,0) + "#"
HC5->HC5_TIPO    :=     "N"
cData += HC5->HC5_TIPO + "#"
HC5->HC5_TIPOCL  :=    "F"
cData += HC5->HC5_TIPOCL + "#"
//HC5->HC5_VEND1 := cCodSuperv
HC5->HC5_COND           := '001'
cData += HC5->HC5_COND + "#"
HC5->HC5_HH             := .T.
cData += If(HC5->HC5_HH, "T", "N") + "#"
HC5->HC5_EXTRA          := If(lExtra,.T.,.F.)
cData += If(HC5->HC5_EXTRA, "T", "N") + "#"
HC5->HC5_STATUS  := "N"
cData += HC5->HC5_STATUS + "#"
```

//Chamada da função para criptografia do inventário informado

```
cDtEncry := CryptoData(cData, cKey)

HC5->HC5_ENC1 := If(Len(cDtEncry) > 250, SubStr(cDtEncry,1,250), cDtEncry)
HC5->HC5_ENC2 := If(Len(cDtEncry) > 500, SubStr(cDtEncry,251,250), "")
HC5->HC5_ENC3 := If(Len(cDtEncry) > 750, SubStr(cDtEncry,751,Len(cDtEncry)), "")
HC5->(dbCommit())


dbSelectArea("HC6")
dbSetOrder(1)
For y := 1 to Len(aItemProd)

        If !lExtra
                // Ignora os Itens nao inventariados
                If aItemProd[y,1] == .F.
                        Loop
                EndIf
        Else
                // Ignora os Itens zerados
                If aItemProd[y,4] = 0
                        Loop
                EndIf
        EndIf

        nItem++
        cItem := StrZero(nItem,3)
```

```
        // Dados do Produto
        nQtdPdr  := aItemProd[y,6]
        cProduto := aItemProd[y,5]
        cDescr   := aItemProd[y,2]
        cUm              := aItemProd[y,3]
        nQtdInv          := aItemProd[y,4]
        nQtdVen  := If(!lExtra, (nQtdPdr - nQtdInv), nQtdInv)   // Qtd padrao menos a Qtd
Inventariada
        nPrcVen  := 1
        nPrUnit  := 1

        HC6->(dbAppend())
        HC6->HC6_FILIAL := RetFilial("HC6") //"01" //"02"
        HC6->HC6_ITEM       := cItem
        HC6->HC6_NUM        := cProPed
        HC6->HC6_COTAC      := cProPed
        HC6->HC6_PROD       := cProduto
        HC6->HC6_DESCRI     := cDescr
        HC6->HC6_UM         := cUm
        HC6->HC6_QTDVEN     := nQtdVen
        HC6->HC6_PRCVEN     := 1
        HC6->HC6_PRUNIT     := 1
        HC6->HC6_VALOR      := nQtdVen
        HC6->HC6_TES  := "524"
        HC6->HC6_LOCAL      := "01"
        HC6->HC6_ENTREG     := Date()
        HC6->HC6_STATUS := "N"
        HC6->(dbCommit())
Next

// Mensagens de Gravacao
cMsg := "PEDIDO "
If lExtra
        cMsg += "EXTRA "
EndIf
cMsg += cProPed + " GRAVADO COM SUCESSO !!!"

cMsgItens := "Foram gravado(s) "
cMsgItens += Alltrim(Str(nTotItens,3,0))
cMsgItens += " iten(s). "
cMsgItens += "No próximo sincronismo o pedido será enviado para a Dinâmica Matriz."

// Atualiza array indicando que invetário foi realizado
aItemCli[nLinCli,6] := .T.
GridSetCellColor(oBrwCli,nLinCli,1,CLR_YELLOW ,CLR_BLACK)

// Definicao do Dialogo Principal
```

```
DEFINE DIALOG oDlgGravado TITLE        "Dinâmica Servicos - Gravação" COLOR CLR_WHITE,
CLR_BLUE

@ 030,05 GET oGet VAR cMsg READONLY MULTILINE NO UNDERLINE COLOR CLR_WHITE, CLR_HRED
SIZE 150,50 OF oDlgGravado
@ 070,05 GET oGet VAR cCliente READONLY MULTILINE NO UNDERLINE SIZE 150,40 OF oDlgGravado
@ 100,05 GET oGet VAR cMsgItens READONLY MULTILINE NO UNDERLINE SIZE 150,40 OF
oDlgGravado
If !lExtra
        @ 145,057 BUTTON oBtn CAPTION "Continuar" ACTION InitExtra(aItemCli,oBrwCli) SIZE 45,13
OF oDlgGravado
Else
        @ 145,057 BUTTON oBtn CAPTION "Continuar" ACTION CloseDialog() SIZE 45,13 OF
oDlgGravado
EndIf
ACTIVATE DIALOG oDlgGravado
Return

/*-------------------------------------------------------------------------------------
  |Funçao | InitExtra
  -------------------------------------------------------------------------------------
  |Descr. | Inicia Pedido de Vendas extra
  -------------------------------------------------------------------------------------
  |Param. | oBrwCli : Browse com lista de produto do contrato
                      aItemCli: Array com contratos do supervisor
  -------------------------------------------------------------------------------------*/
Function InitExtra(aItemCli,oBrwCli)

If !MsgYesOrNo("Deseja gerar Pedido Extra para esse Cliente?","Pedido Extra")
        CloseDialog()
        CloseDialog()
Else
        PedExtra(aItemCli,oBrwCli)
        CloseDialog() // Fecha tela de Confirmacao do Pedido
        CloseDialog()
EndIf
Return

/*-------------------------------------------------------------------------------------
  |Funçao | QtdInv
  -------------------------------------------------------------------------------------
  |Descr. | Exibe tela para digitação da quantidade inventariada de um produto
  -------------------------------------------------------------------------------------
  |Param. | oBrwProd : Browse com lista de produto do contrato
                      aItemProd: Array com produtos do contrato
  -------------------------------------------------------------------------------------*/
Function QtdInv(oBrwProd,aItemProd)
Local oGet, oGetQtd
Local oBtn, oCalc
```

```
Local oDlg
Local nQuant   := 0
Local nLinha   := 0
Local cTitle   := "Digite a Quantidade Inventariada para o Produto: "
Local cProduto := ""

// Selectiona a Linha do Browse
nLinha   :=       GridRow(oBrwProd)
cProduto :=       aItemProd[nLinha,2] //Descricao do Produto
nQuant   :=       aItemProd[nLinha,4] // Quantidade

DEFINE DIALOG oDlg TITLE "Inventário"

@ 00,060 SAY "Superior: " + cSupervisor OF oDlg

@ 45,05 GET oGet VAR cTitle READONLY NO UNDERLINE MULTILINE SIZE 150, 30 OF oDlg

@ 70,05 GET oGet VAR cProduto READONLY NO UNDERLINE MULTILINE SIZE 150, 30 OF oDlg

@ 90,50  GET oGetQtd VAR nQuant PICTURE "@E 999,999.99"  SIZE 50,20 OF oDlg
@ 90,120 BUTTON oCalc CAPTION CALC SYMBOL ACTION PVQTde(oGetQtd) OF oDlg

@ 135,050 BUTTON oBtn CAPTION "Confirma" ACTION GravaBrow(nQuant,oBrwProd,@aItemProd)
SIZE 45,13 OF oDlg
@ 135,105 BUTTON oBtn CAPTION "Cancelar" ACTION CloseDialog() SIZE 45,13 OF oDlg

SetFocus(oGetQtd)

ACTIVATE DIALOG oDlg
Return .T.

/*-----------------------------------------------------------------------------------
  |Funçao | QtdInv
  -----------------------------------------------------------------------------------
  |Descr. | Atualiza informações no browse de itens inventariados do contratro
  -----------------------------------------------------------------------------------
  |Param. | nQuant   : Quantidade digitada
                       oBrwProd : Browse com lista de produto do contrato
                       aItemProd: Array com produtos do contrato
  -----------------------------------------------------------------------------------*/
Function GravaBrow(nQuant,oBrwProd,aItemProd)
Local nRow    := GridRow(oBrwProd)
Local nQtdPdr := aItemProd[nRow,6]
Local cDesProd:= AllTrim(aItemProd[nRow,2])
Local cCodProd:= AllTrim(aItemProd[nRow,5])
// Verificar quantidade digitada
If nQuant > nQtdPdr
        //MsgStop("A quantidade inventariada ("+ AllTrim(Str(nQuant,5,0))+") não pode ser maior do
que o KIT padrão (" + Str(nQtdPdr,5,0) + ").")
```

```
        MsgStop("A quantidade inventariada ("+Str(nQuant,5,0)+") inválida para o produto " +
cDesProd + ".")
        Return .F.
EndIf

///If nQuant > 0
        aItemProd[nRow,1] := .T.
        aItemProd[nRow,4] := nQuant

        SetArray(oBrwProd, aItemProd)
        GridSetCellColor(oBrwProd,nRow,1,CLR_HRED ,CLR_WHITE)
        CloseDialog()
//EndIf
Return .T.

/*-----------------------------------------------------------------------------
  |Funçao | InitSync
  -----------------------------------------------------------------------------
  |Descr. | Inicia processo de sincronismo
  -----------------------------------------------------------------------------
  |Param. | oSayFile    : Objeto de status na abertura de arquivos
                          oMeterFiles : Objeto de progresso na abertura de arquivos
                          nMeterFiles : Objeto de controle no progresso da abertura de arquivos
  -----------------------------------------------------------------------------*/
Function InitSync(oSayFile, oMeterFiles, nMeterFiles)
Local aEmp := {}

dbCloseAll()
DoSync()

// Verifica arquivo de Empresas
If OpenEmp(aEmp)
        // Abre arquivos de Dados
        OpenFiles(oSayFile, oMeterFiles, nMeterFiles)
EndIf

Return Nil

/*-----------------------------------------------------------------------------
  |Funçao | Show_Cli
  -----------------------------------------------------------------------------
  |Descr. | Atualiza dados do cliente na tela de contratos
  -----------------------------------------------------------------------------
  |Param. | oBrwCli : Browse com lista de contratos
                          aItemCli: Array com contratos do supervisor
                          oCliente: Objeto de exibição de dados do cliente
  -----------------------------------------------------------------------------*/
Function Show_Cli(oBrwCli,aItemCli,oCliente)
Local nLinha   := 0
```

```
Local cCliente := ""

nLinha   := GridRow(oBrwCli)
cCliente := aItemCli[nLinha,1]
SetText(oCliente,cCliente)
Return

/*-----------------------------------------------------------------------------
  |Funçao | Show_Prod
  -----------------------------------------------------------------------------
  |Descr. | Atualiza dados do produto na tela de inventário
  -----------------------------------------------------------------------------
  |Param. | oBrwProd : Browse com lista de produto do contrato
                       aItemProd: Array com produtos do contrato
                       oProduto : Objeto de exibição de dados do produto
  -----------------------------------------------------------------------------*/
Function Show_Prod(oBrwProd,aItemProd,oProduto)
Local nLinha   := 0
Local cProduto := ""

nLinha := GridRow(oBrwProd)
cProduto := aItemProd[nLinha,2]
SetText(oProduto,cProduto)

Return

/*-----------------------------------------------------------------------------
  |Funçao | ConfRet
  -----------------------------------------------------------------------------
  |Descr. | Função para cancelar inventário
  -----------------------------------------------------------------------------
  |Param. | Nao de aplica
  -----------------------------------------------------------------------------*/
Function ConfRet()
Local lRet := .F.
If MsgYesOrNo("Abandona Inventario?","Confirmação de Retorno")
        lRet := .T.
        CloseDialog()
EndIf
Return lRet
```

/* Função CryptoData no palm que aciona a função EncRC4 para realizar a criptografia do inventário antes do envio */

```
pObj_Node __CryptoData(pObj_Node cText, pObj_Node cChave)
{
      char *strenc = NULL;
```

```
    pObj_Node res;
    if(cText->Length > 0 && cChave->Length > 0)


    {
        strenc = (char*)xmalloc(TAM_MAX_BUFF_TEMP);
        xmemset(strenc, 0x00, TAM_MAX_BUFF_TEMP);
        EncRC4((unsigned char*)__GET_STRING(cText), (unsigned char*)__GET_STRING(cChave),
strenc);
    }
    res = __INIT_CHARPTR(strenc);
    if(strenc)
        xfree(strenc);
    return res;
}
```