THATIANE CRISTINA DOS SANTOS DE CARVALHO RIBEIRO

SEGURANÇA DA INFORMAÇÃO PARA APLICAÇÕES INTERATIVAS NO SISTEMA BRASILEIRO DE TELEVISÃO DIGITAL

UMA METODOLOGIA BASEADA EM ACESSO A WEB SERVICE EM APLICAÇÕES INTERATIVAS

CAMPINAS
2014

Universidade Estadual de Campinas
Faculdade de Engenharia Elétrica e de Computação

THATIANE CRISTINA DOS SANTOS DE CARVALHO RIBEIRO

SEGURANÇA DA INFORMAÇÃO PARA APLICAÇÕES INTERATIVAS NO SISTEMA BRASILEIRO DE TELEVISÃO DIGITAL

UMA METODOLOGIA BASEADA EM ACESSO A WEB SERVICE EM APLICAÇÕES INTERATIVAS

Orientador: Prof. Dr. Yuzo Iano
Coorientador: Prof. Dr. Vicente Idalberto Becerra Sablón

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas para obtenção do título de Doutora em Engenharia Elétrica, na área de Telecomunicações e Telemática (AG).

Este exemplar corresponde à versão final dissertação/tese defendida pela aluna Thatiane Cristina dos Santos de Carvalho Ribeiro e orientada pelo prof. Dr. Yuzo Iano.

_____

CAMPINAS
2014

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca da Área de Engenharia e Arquitetura
Rose Meire da Silva - CRB 8/5974

R354s

Ribeiro, Thatiane Cristina dos Santos de Carvalho, 1982-
Segurança da informação para aplicações interativas no sistema brasileiro de televisão digital - uma metologia baseada em acesso Web Service em aplicações interativas / Thatiane Cristina dos Santos de Carvalho Ribeiro. – Campinas, SP : [s.n.], 2014.

Orientador: Yuzo Iano.
Coorientador: Vicente Idalberto Becerra Sablon.
Tese (doutorado) – Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Televisão digital. 2. Tecnologia da informação - Segurança. 3. Sistema de televisão. I. Iano, Yuzo,1950-. II. Becerra Sablon, Vicente Idalberto. III. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. IV. Título.

Informações para Biblioteca Digital

Título em outro idioma: Information security for interactive applications in brazilian system of digital television
Palavras-chave em inglês:
Digital television
Information technology - Security
Television system
Área de concentração: Telecomunicações e Telemática
Titulação: Doutora em Engenharia Elétrica
Banca examinadora:
Yuzo Iano [Orientador]
Adão Boava
Marcelo Augusto Costa Fernandes
Luiz César Martini
Geraldo Peres Caixeta
Data de defesa: 31-01-2014
Programa de Pós-Graduação: Engenharia Elétrica

COMISSÃO JULGADORA - TESE DE DOUTORADO

Candidata: Thatiane Cristina dos Santos

Data da Defesa: 31 de janeiro de 2014

Título da Tese: "Segurança da Informação para Aplicações Interativas no Sistema Brasileiro de Televisão Digital - Uma Metodologia Baseada em Acesso a Web Service em Aplicações Interativas"

Prof. Dr. Yuzo Iano (Presidente): _____

Prof. Dr. Adao Boava: _____

Dr. Marcelo Augusto Costa Fernandes: _____

Prof. Dr. Luiz César Martini: _____

Prof. Dr. Geraldo Peres Caixeta: _____

RESUMO

Apresenta-se neste trabalho uma metodologia que permite transmitir com segurança as informações no SBTVD – Sistema brasileiro de Televisão digital. O modelo baseia-se no acesso a web service em aplicações de TVD interativas. A proposta têm como base a proteção da informação, via mecanismos de controle, contra possíveis ameaças – seja por ação intencional, mau uso do aplicativo, defeitos ou falhas na programação – que ocorram onde a informação estiver sendo criada, processada, armazenada ou transmitida. Uma implementação real dos serviços propostos serve como prova do conceito da eficácia no uso da metodologia apresentada. Na avaliação dos resultados realizada foi possível obter uma visão geral da situação atual em que se encontra a gestão da segurança da informação da organização, bem como verificar os pontos que estão de acordo com a normatização e daqueles que necessitam aprimoramentos no SBTVD.

Palavras-chave: Televisão Digital, Tecnologia da Informação - Segurança, Sistema de Televisão.

ABSTRACT

This paper presents a methodology to securely transmit information in the SBTVD – Brazilian Television Digital System. The model builds on noted access to web service in interactive digital TV applications. The proposal promotes the protection of information, mechanisms for control against possible threats - whether by intentional action, misuse of the application, defects or failures in programming - that occur where information is being created, processed, stored or transmitted. A real implementation of the proposed services serves as proof of concept of the effectiveness in the use of the methodology presented. In the evaluation of the results was performed can get an overview of the current situation that is managing the organization's information security, as well as verification of the points that conform to standards and those that need enhancements in the– Brazilian Television Digital System.

Keywords*: Digital Television, Technology of Information - Security, TV System.*

SUMÁRIO

Dedico este trabalho ao meu marido pelo amor, empenho e dedicação nesta jornada.

"Porque a tua benignidade é melhor do que a vida, os meus lábios te louvarão. Assim eu te bendirei enquanto viver; em teu nome levantarei as minhas mãos."

Salmo 63 -3:4

AGRADECIMENTOS

Agradeço a Deus por estar sempre comigo, por me auxiliar e nunca me desamparar. Ao meu orientador Professor Dr. Yuzo Iano. Ao Prof. Dr. Vicente Idalberto Becerra Sablón, por sua dedicação na elaboração e conclusão do trabalho. Ao meu marido por ser sempre meu porto seguro em todos os momentos. A todos que de alguma forma contribuíram para o desenvolvimento do trabalho. Nossos agradecimentos a CAPES (Coordenação de Aperfeiçoamento Pessoal de Nível Superior), a CAPES RH-TVD, ao CNPq, a FAEPEX/UNICAMP, a RNP/CTIC (Rede Nacional de Pesquisa/Centro de Pesquisa e Desenvolvimento em Tecnologias Digitais para Informação e Comunicação) e a PRP/Unicamp pelo incentivo para a realização deste trabalho.

AGRADECIMENTOS

Agradeço ao programa CAPES RH-TVD da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior tanto pelo apoio financeiro quanto pelo incentivo acadêmico para que este trabalho pudesse ser realizado.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

3G	Terceira Geração
AAC	*Advanced Audio Coding*
ABNT	Associação Brasileira de Normas Técnicas
AVC	*Advanced Video Coding*
BP	*Basic Profile*
BST	*Band Segmented Transmission*
Capes	Coordenação de Aperfeiçoamento de Pessoal de Nível Superior
CPG	Comissão de Pós-Graduação
Decom	Departamento de Comunicações
DoS	*Denial-of-Service*
EPG	*Electronic Program Guide*
EULA	*End User Licence Agreement*
HEAAC	*High-Efficiency Advanced Audio Coding*
HP	*High profile*
HTML	*HyperText Markup Language*
HTTP	*Hypertext Transfer Protocol*
IEEE	*Institute of Electrical and Electronics Engineers*
IM	*Instant Message*
IP	*Internet Protocol*
IrDA	*Infrared Data Association*
JSP	*Java Server Pages*
MMS	*multimedia messaging service*
MPEG4	*Moving Picture Experts Group*
NCL	*Nested Context Language*
OFDM	*Orthogonal Frequency Division Multiplexing*
SBTVD	Sistema Brasileiro de Televisão Digital
SBTVD-T	Sistema Brasileiro de Televisão Digital Terrestre
SSL	*Secure Sockets Layer*
TCP	*Transmission Control Protocol*
TLS	*Transport Layer Security*
TS	*Transport Stream*
TV	Televisão
UMTS	*Universal Mobile Telecommunication System*
Unicamp	Universidade Estadual de Campinas
VPN	*Virtual Private Network*
Wi-Fi	*Wireless Fidelity*
WLAN	*Wireless Local Area Network*

PUBLICAÇÕES REALIZADAS

SANTOS, T. C.; SABLON, V. I.B.; COVRE, J.P.; IANO, Y.. *Authentication Applications for the Brazilian System of Digital Television* – SBTVD. Revista del IEEE America Latina. ISSN: 1548-0992. Submetido em Novembro de 2013.

SANTOS, T. C.; SABLON, V. I.B.; COVRE, J.P.; IANO, Y.. *Controle de Segurança da Informação nas Aplicações Interativas do Sistema Brasileiro de Televisão Digital*. Revista de Ciências Gerenciais. Sistema Anhanguera de Revistas Eletrônicas. ISSN 1415-6571 e ISSN 2178-6909. Submetido em Setembro de 2013.

SANTOS, T. C.; SABLON, V. I.B.; IANO, Y.. *Segurança da Informação para Aplicações Interativas no Sistema Brasileiro de Televisão Digital.* In: Eleventh LACCEI Latin American and Caribbean Conference for Engineering and Technology (LACCEI'2013).

MAKLUF, C. A.; SANTOS, T. C.; REUIZ, J. L. ; Yuzo Iano ; ARTHUR, R. . *A Study on the Technical Viability for Structuration of the Brazilian Digital Television s Return Channel using 3G Technologies.* in Proceedings of LatinDisplay 2010/IDRC 2010, 2010 ISSN: 1946-3871

MAKLUF, C. A. ; SANTOS, T. C.; REUIZ, J. L. ; IANO, Y. ; ARTHUR, R., *Análise de desempenho de tecnologias 3g visando à estruturação do canal de retorno da TV digital,* Rev. Ciência e Tecnol., vol. 15, no. 26, pp. 9–15, 2012.

SANTOS, Thatiane Cristina dos; IANO, Y. ; Omar Branquinho. *Análise de Desempenho da Transmissão de Vídeo em Redes IEEE 802.11 visando à estruturação do canal de retorno para TV Digital*. Revista Ciência e Tecnologia, v. 18, p. 29-38, 2008.

xxx

1 INTRODUÇÃO

A TV Digital oferece para o usuário não apenas melhor qualidade de imagem e som, mas também uma gama de novos serviços e aplicações de entretenimento e de informações. Assim sendo, a adoção da TV Digital no Brasil aliada ao avanço da tecnologia, permite que serviços e aplicações sejam disponibilizados mesmo em localidades remotas, contribuindo para a universalização e democratização de informações e serviços eletrônicos, permitindo a inclusão social de uma parcela maior da população brasileira.

A interatividade possibilita a utilização da televisão digital para fazer transações pessoais, comerciais, entre outras. Isso gera a necessidade de mecanismos de seguranças capazes de identificar ameaças durante a transição de informações através das aplicações interativas.

O *Middleware* é a camada de *software* intermediário, entre o hardware/Sistema Operacional e as aplicações, no SBTVD (Sistema Brasileiro de Televisão Digital), é conhecido como Ginga. O *middleware* proporciona um ambiente onde as transações de informações em aplicações são realizadas. Nessas transações os dados pessoais dos usuários devem ser protegidos e a proteção das aplicações deve manter a disponibilidade, confiabilidade e a integridade dos dados. [1]

O estudo objetiva manter a segurança da informação para aplicações do Sistema Brasileiro de Televisão Digital. As aplicações precisam ter uma rotina que determina se estas foram desenvolvidas de maneira que assegura que os dados não serão violados.

Na construção de modelos de segurança encontram-se dificuldades para manter o sistema protegido em todas as situações, principalmente quando é preciso ter mecanismos de interação com o usuário, o sistema deve ser o mais robusto e autoexplicativo. O usuário ao utilizar a aplicação recebe as informações de permissão e negação em determinada ação, pode instalar um componente ou mesmo permitir um acesso aos seus dados. Os pilares da segurança da informação, integridade, confiabilidade e disponibilidade serão assegurados a partir das tomadas de decisão do usuário.

Mecanismos de autenticação de aplicativos, como a utilização de protocolos de segurança e requisitos de segurança no uso do canal de interatividade para aplicações são formas de manter a segurança no Sistema SBTVD.

1.1 OBJETIVOS

Estudar a situação atual da gestão da segurança da informação, bem como verificar dos pontos que estão de acordo com a normatização e daqueles que necessitam aprimoramentos no SBTVD.

Criar uma metodologia baseada em acesso *web service* em aplicações de televisão digital interativas, sendo fundamentada em resultados alcançados para sistema de telefonia celular, por serem sistemas análogos, em comparação às técnicas de ataque, as técnicas de segurança e proteção.

1.2 MOTIVAÇÃO

A TV Digital no Brasil aliada ao avanço da tecnologia, poderá permitir que serviços e aplicações sejam disponibilizados mesmo em localidades remotas, contribuindo para a universalização e democratização de informações e serviços eletrônicos, permitindo a inclusão social de uma parcela maior da população brasileira.

A disponibilidade de serviços para pessoas em qualquer lugar, a proteção dos seus dados, as políticas de privacidade, segurança das empresas fornecedoras dos aplicativos e o desenvolvimento de uma metodologia para Segurança da informação para aplicações interativas compatíveis com o *middleware* de referência proposto para o SBTVD constitui o motivador deste trabalho.

1.3 METODOLOGIA

A metodologia do trabalho consiste em desenvolver um método para as certificações das aplicações de televisão digital, baseada em acesso *web service*. A segurança da informação nos aplicativos promove a proteção da informação, via mecanismos de controle, contra possíveis ameaças

existentes – seja por ação intencional, mau uso do aplicativo, defeitos ou falhas na programação – que ocorram onde a informação estiver sendo criada, processada, armazenada ou transmitida. O protocolo de certificação SSL é testado em uma aplicação que utiliza a interface HTTP para transmissão dos dados do usuário. Os dados recebidos pela camada de aplicação são comprimidos, assinados e criptografados. A construção de aplicações com interatividade realiza acesso ao canal de retorno por meio de protocolos padronizados. A autenticidade, a disponibilidade e confiabilidade são asseguradas na aplicação segura quando constantemente verificada, validada e monitorada as informações do usuário.

1.4 ESTRUTURA DO TRABALHO

Este trabalho está organizado da seguinte forma. No capítulo 2 são apresentados os padrões utilizados no SBTVD, o canal de interatividade e a arquitetura do *middleware* Ginga.

No capítulo 3 apresenta-se um estudo sobre a segurança da informação, vulnerabilidades, ameaças, riscos e necessidades de segurança. Na seção 3.1 e apresentada as técnicas de segurança em serviços da televisão digital interativa. A seção 3.2 mostra a segurança do ponto de vista do usuário. A seção 3.3 apresenta um estudo sobre as ameaças e vulnerabilidades.

No capítulo 4 apresenta-se a metodologia proposta e os mecanismos de segurança. Neste Capitulo é apresentada arquitetura, rotas de infecção e os protocolos de segurança.

No Capítulo 5 os testes e resultados são descritos, descreve-se também a aplicação testada, o módulo de segurança e o acesso seguro. No capítulo 6 são discutidos os resultados obtidos.

No capítulo 7 são apresentadas as conclusão e as sugestões para trabalhos futuros.

2 O SISTEMA BRASILEIRO DE TELEVISÃO DIGITAL - SBTVD

Neste capítulo, o Sistema Brasileiro de Televisão Digital – SBTVD será abordado. A infraestrutura do sistema, os padrões adotados para a difusão e o acesso e para o terminal de acesso são detalhados.

O governo brasileiro determinou os requisitos básicos para o Padrão do Sistema de TV Digital, como o baixo custo e robustez na recepção, flexibilidade e capacidade de evolução, interatividade e novos serviços, visando promover a inclusão digital [2].

A arquitetura proposta baseia-se no modelo de referência sugerida pela União Internacional de Telecomunicações - UTI [2]. No projeto brasileiro optou-se por representar de forma única as funções de Multiplexação e Transporte, agrupadas na camada de transporte. De forma análoga, a codificação de canal, modulação e transmissão estão representadas em um único módulo. Por fim, o receptor digital foi expandido, para que fosse possível dar uma ênfase maior à sua arquitetura.

O sistema é definido como uma plataforma multimídia capaz de transmitir sinais de áudio e vídeo de alta qualidade, bem como dados, utilizando o sinal de radiodifusão. A capacidade de transmissão de dados, que podem estar vinculados ou não à programação, possibilita o desenvolvimento de novos serviços e aplicações digitais [3].

O sistema atua como uma plataforma de comunicação entre a emissora de conteúdo e os usuários finais, fazendo uso das aplicações interativas, e está dividido em duas entidades complementares: a difusão e acesso e o terminal de acesso, como mostra a Figura 2.1.

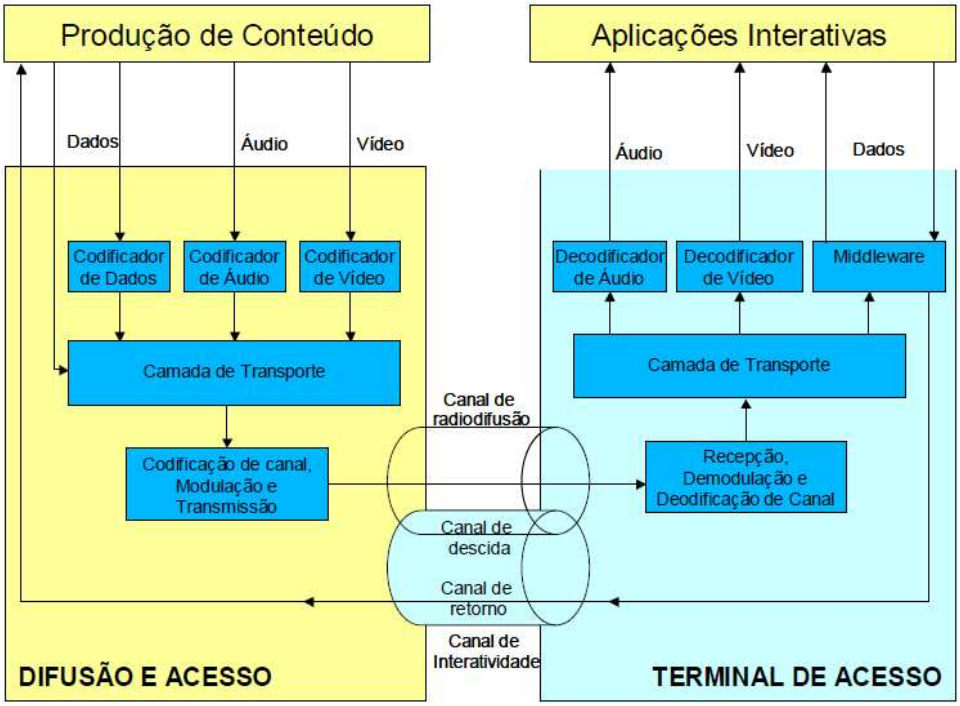

Figura 2.1 - Diagrama de Fluxo de Informação. [3]

Os processos de Difusão e Acesso são constituídos pelos módulos de codificação e empacotamento das informações a serem transmitidas para os receptores. Para que os sinais de áudio, vídeo e os dados, originados na Produção de Conteúdo, possam ser transmitidos, estes precisam ser codificados, o que inclui a sua compactação e a inserção de informações que permitam, posteriormente, a sua decodificação, pelos Codificadores de Áudio, de Vídeo e de Dados, respectivamente, conforme se apresenta na Figura 2.1 [3]. Uma vez codificados, os sinais são processados pela Camada de Transporte, que os empacota e transmite em um único sinal de transporte, ou fluxo de transporte TS – (*Transport Stream*), acrescentando-lhes informações auxiliares de controle. Na etapa seguinte, o sinal gerado na Camada de Transporte passa por um processamento adicional no módulo de Codificação de Canal, Modulação e Transmissão, por onde é transmitido.

O Terminal de Acesso é composto pelos módulos necessários para efetuar o processamento reverso ao da Difusão e Acesso, reconstituindo as informações originais de áudio, vídeo e dados. O sinal recebido pelo Terminal de Acesso, através de antenas receptoras, no módulo de Recepção, Demodulação e Decodificação de Canal, passa por um processo de demodulação e de decodificação de canal, de onde resulta o sinal de transporte que será enviado à etapa de demultiplexação,

no módulo da Camada de Transporte. Os sinais codificados de Áudio, Vídeo e Dados, que são então submetidos aos Decodificadores de Áudio, de Vídeo e ao *Middleware*, respectivamente. Os Decodificadores de Áudio e Vídeo reconstituem os sinais originais, para que possam ser corretamente exibidos. O *Middleware*, por outro lado, além de decodificar os dados recebidos, é responsável por tratar as instruções, funcionando como uma plataforma de execução de *software*. Como resultado final, têm-se as Aplicações Interativas sendo utilizadas pelos usuários.

O sistema possui ainda um Canal de Interatividade, composto de um Canal de Descida e um Canal de Retorno, que possibilita a interação do usuário final com a Produção de Conteúdo, permitindo- lhe receber ou enviar solicitações e informações, como apresentado na Figura 2.1.

2.1 PADRÕES ADOTADOS PELO SISTEMA BRASILEIRO DE TV DIGITAL

O sistema de TV digital é composto por um conjunto de padrões que regulam cada uma das etapas já descritas. O Sistema Brasileiro de TV Digital adotou os seguintes padrões:

- Para codificação de áudio – o padrão *Moving Picture Expert Group* – parte 4 – MPEG4 com 2 níveis de perfil para receptores fixos e móveis (*Advanced Audio Coding* – AAC@L4 – para multicanal 5.1 e *High Efficiency Advanced Audio Coding* – HEAAC v1@L4 – para estéreo) e 1 nível de perfil para receptores portáteis (HEAAC v2@L3 – dois canais).
- Para codificação de vídeo – o padrão MPEG4- (AVC/H.264) com o nível de perfil (alto), *High Profile* – HP@L4.0 para receptores fixos e móveis e o nível de perfil (básico) – *Basic Profile* – BP@L1.3 para receptores portáteis.
- Para o sistema de transporte (multiplexação e demultiplexação) – o padrão MPEG-2 Systems. (H.262 - ISO/IEC 13818-2).
- Para o processo de modulação foi adotado o padrão *Band Segmented Transmission Orthogonal Frequency Division Multiplexing* – BST – OFDM/SBTVD-T.
- Para a camada de *middleware* – o padrão GINGA [3].

Na Figura 2.2 apresenta-se os padrões de referência e as interações entre eles.

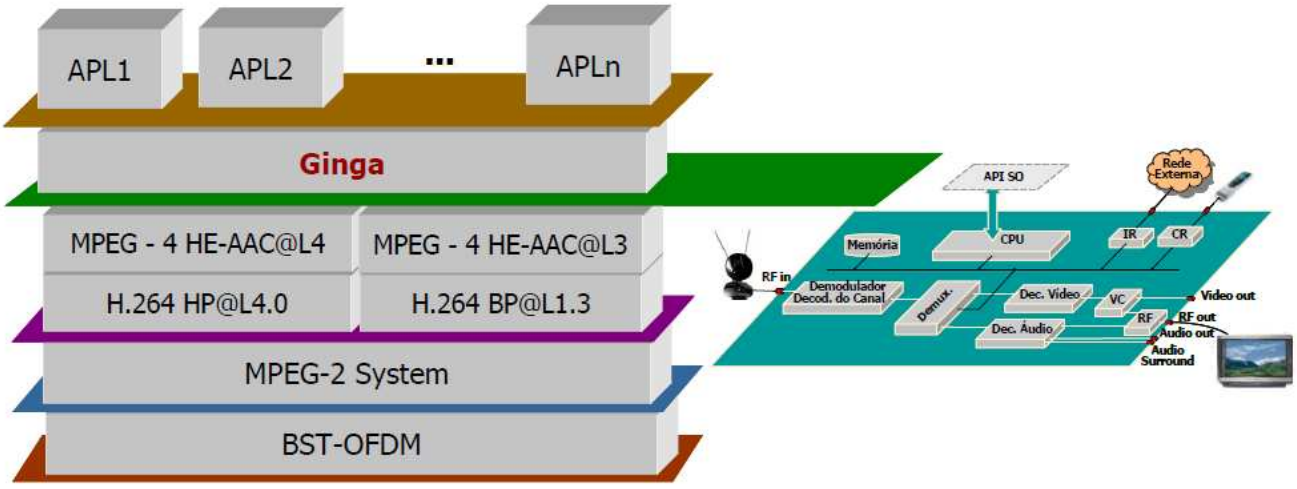

Figura 2.2 - Padrões de Referência do Sistema Brasileiro de TV Digital Terrestre. [3]

O Sistema de Televisão pode ser dividido em dois subsistemas simétricos: um referente à transmissão e outro referente ao usuário, sendo esta formada pela recepção e pelo tráfego de informação. Os serviços, aplicações e conteúdos estão relacionados com os dois subsistemas. A codificação de sinais na difusão e decodificação está no terminal de acesso do usuário. A multiplexação na difusão e demultiplexação são inerentes à camada de transporte. A transmissão, modulação e codificação do canal de difusão e recepção, demodulação e decodificação de canal estão no terminal de acesso. A Figura 2.3 apresenta o receptor do SBTVD. [3]

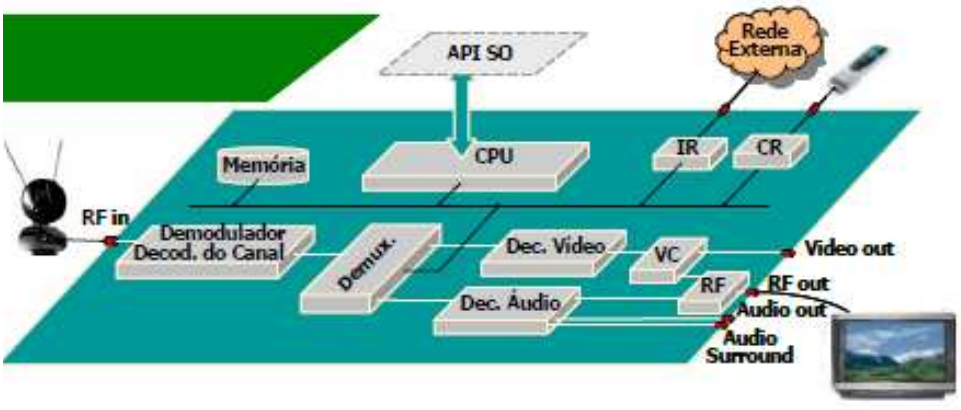

Figura 2.3 – Receptor do Sistema Brasileiro de TV Digital Terrestre. [3]

2.2 CANAL DE INTERATIVIDADE

O canal de interatividade é um meio que possibilita ao usuário, individualmente, interagir encaminhando ou recebendo informações e solicitações das emissoras/programadoras como: provedor de conteúdo, provedor de serviço/aplicações, provedor de interatividade, provedor de rede, programador, distribuidor, outros usuários, resultando em dois canais de comunicação: canal de descida e o canal de retorno [4]. A Figura 2.4 apresenta o diagrama simplificado dos canais.

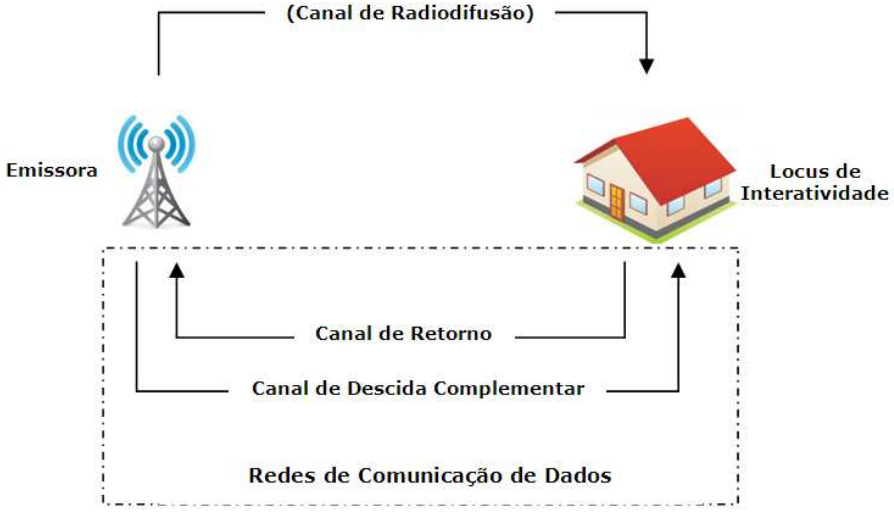

Figura 2.4 - Diagrama Simplificado do Sistema Canal de Retorno. [5]

O canal de descida estabelece a comunicação no sentido emissoras/programadoras para os usuários, sendo constituído pelos canais de radiodifusão, podendo ter uma comunicação *broadcast* (ponto-multiponto), aberta e disponível a todos os usuários ou uma comunicação *unicast* (ponto a ponto) individualizada. O canal de retorno estabelece a comunicação no sentido dos usuários para as emissoras/programadoras, e é composto por redes de acesso para o estabelecimento da ligação entre usuário e emissoras/programadoras. Desse modo, pode ocorrer a transferência e troca de dados, de ambos os lados, permitindo assim a interatividade [6]. O grau de interação do usuário com as aplicações, serviços e conteúdos interativos podem ser divididos em três categorias: local, intermitente e permanente [7].

A interatividade local é a mais básica das três categorias, o difusor é composto pelo provedor de serviço de difusão, que gera o sinal dos programas de televisão para que o canal de difusão transmita os fluxos de áudio e vídeo para o receptor doméstico de forma unidirecional. Já na interatividade intermitente ou remota unidirecional, algumas mudanças significativas são realizadas, de forma que, nessa categoria, a comunicação do usuário em direção ao difusor seja possível. O difusor apresenta, além do provedor do serviço de difusão, outro provedor denominado provedor de serviço de interação. A interatividade permanente ou remota bidirecional é considerada uma evolução da interatividade intermitente, na qual a comunicação dos dados no canal de interação deixa de ser unidirecional para se tornar bidirecional, existindo para isso um canal de retorno dedicado no receptor digital [7] [8].

A Figura 2.5 mostra o canal de interatividade apresentando seus subsistemas, no qual a emissora/ programadora de conteúdos recebe os dados do usuário através de uma aplicação interativa.

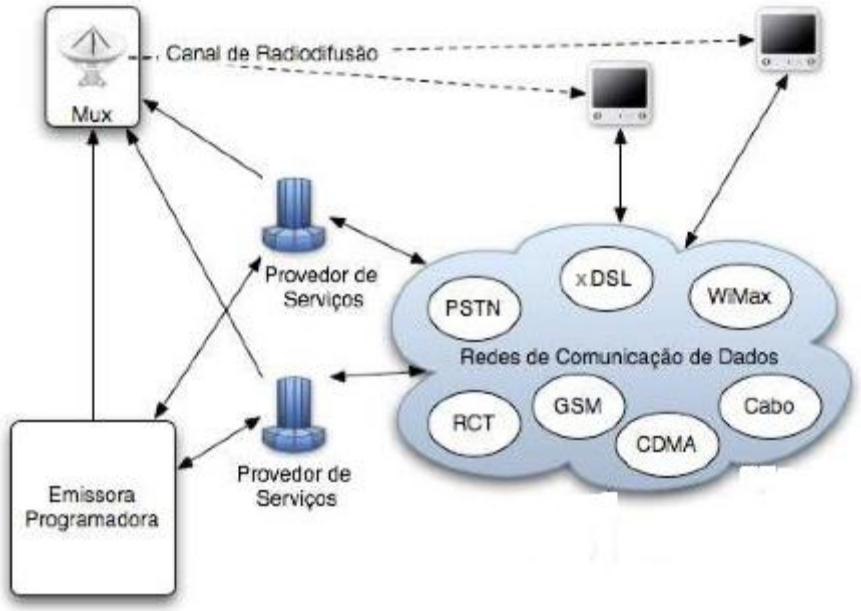

Figura 2.5 – Arquitetura do Canal de Interatividade. [7]

Uma característica importante é a diversidade de alternativas tecnológicas para a implementação do acesso ao canal de retorno. Essa diversidade é importante porque oferece soluções para dificuldades técnicas, otimização de recursos e exigências distintas em adequação aos diferentes ce-

nários geográficos, populacionais, socioeconômico e de redes de comunicação. Essas tecnologias não são excludentes, admitindo, inclusive, que novas tecnologias sejam agregadas em complementaridade [8].

2.2.1 Aplicações Interativas

Para o desenvolvimento, estudo, especificações e protótipos de aplicações interativas compatíveis com o padrão de *middleware* do padrão brasileiro de Televisão Digital compreende um conjunto de serviços e aplicações interativas disponibilizadas através de um televisor e um decodificador, chamado *Set-top-Box* (STB). A TV interativa permite que o telespectador através dos aplicativos interaja com a programação, como por exemplo, escolhendo a câmera (ângulo) em um jogo de futebol, participando de votações e jogos de auditório, escolhendo suas preferências em aplicativos interativos como previsão de tempo, bolsas de valores, notícias de última hora, etc. [11]

A TV digital interativa permitem a navegação do usuário pelas informações disponibilizadas através das aplicações. Aquelas aplicações que são mais avançadas permitem o envio de dados ao provedor do conteúdo utilizando possivelmente a própria infraestrutura da *internet*, caracterizando a utilização do canal de retorno. [12]

As aplicações para TV digital são classificadas em:

- **Aplicações *servicebound*:** apresentam informações ou interatividade contextuais de acordo com o conteúdo exibido de um determinado serviço. Estas aplicações são carregadas no STB toda vez que são executadas.
- **Aplicações *unbound*:** não são contextuais, podem ser acessadas a qualquer momento e não possuem um propósito específico. Guias de programação eletrônica, home banking, jogos, etc.
- **Aplicações armazenadas**: são mais complexas e possuem funções bem mais genéricas. Geralmente são pagas e são armazenadas em um dispositivo de memória para o usuário executá-la durante o período de validade dela.
- **Aplicações embarcadas**: são aplicações nativas do STB, como um guia de programação eletrônica, serviços de alerta de catástrofes, etc.

Cada uma dessas classes de aplicações pode apresentar diferentes níveis de interatividade, conforme foi explicado na seção anterior. [15]

Um dos serviços importantes que se espera disponibilizar com a TV digital interativa é o chamado *t-govern*, são serviços governamentais pela TV, tornando o acesso a eles mais fácil, evitando deslocamentos a cartórios, prefeituras ou postos de informação, levando a todos os cidadãos conhecimento, informação e serviços diversos relativos ao governo, sendo uma forma eficiente de inclusão. Essa ferramenta do governo também busca estimular à democracia e diminuir a burocracia, já que propiciará a transparência da gestão, informações de projetos e oportunidades, programas educativos e culturais e até mesmo transferências financeiras para fins tributários. A Figura 2.6 mostra um exemplo de um aplicativo interativo de televisão digital, no qual ilustra aplicações contextuais de acordo com a programação [11].

Figura 2.6 – Exemplo de Aplicativo Interativo para Televisão Digital. [11]

Uma importante aplicação é conhecida como o *t-learning*, que nada mais é que a educação à distância pela TV. Considerando que poucas pessoas têm acesso a computadores, em vista que cerca de 90% da população brasileira têm acesso aos televisores, busca-se com isso permitir que os telespectadores tenham o mesmo acesso que teriam se estivessem conectados em um computador com *Internet*. Permite ao estudante construir e até mesmo criar o conhecimento, aumentando as habilidades intelectuais [11] [12].

Serviços com o *t-banking* onde o telespectador poderá fazer consultas, transferências, pagamentos e outras operações bancárias pela TV, a qualquer hora, sem sair de casa. Outros serviços: troca de e-mails, comércio eletrônico, EPG (*Electronic Program Guide*), previsão do tempo, acesso a *internet*, *chat*, videoconferência, também serão serviços disponibilizados.

2.3 O MIDDLEWARE DO SISTEMA BRASILEIRO DE TELEVISÃO DIGITAL - GINGA

2.3.1 Arquitetura do *Middleware* Ginga

O *middleware* do SBTVD – Ginga – é uma camada de *software* posicionada entre os códigos das aplicações e a infraestrutura de execução (plataforma de *hardware* e sistema operacional). O Ginga deve ter acesso ao áudio, vídeo, aplicações de dados, entre outros recursos de mídia, que devem ser transmitidos por cabo, ar, satélite ou redes IP (*Internet Protocol*). Dessa forma, as informações são processadas e transmitidas aos usuários, conforme mostra a Figura 2.7. [13]

Figura 2.7 – Contexto do *Middleware* Ginga. [13]

O *middleware* no padrão brasileiro consiste de máquinas de execução das linguagens oferecidas, e bibliotecas de funções, que permitem o desenvolvimento rápido e fácil de aplicações. O universo das aplicações pode ser dividido em três módulos principais: Ginga-CC, Ginga-NCL e Ginga-J, sendo que os dois últimos compõem a camada de serviços específicos e estão separados pelo

tipo de aplicações que são responsáveis. O Ginga-NCL é responsável pelas Aplicações Declarativas e o Ginga-J pelas Aplicações Interativas.

A máquina de apresentação declarativa é baseada na linguagem NCL (*Nested Context Language)*, uma linguagem declarativa para autoria de documentos multimídia e trata de interpretação semântica do modelo NCM (*Nested Context Model)*. Este último é um modelo conceitual e descreve as estruturas de dados, os eventos e o relacionamento entre estes dados, e também definem as regras de estruturação e operações sobre os dados para a manipulação e atualização das estruturas. A Figura 2.8 mostra a Arquitetura do *Middleware* do Ginga, o sistema operacional, os blocos do Ginga-CC, a interface da Máquina de Execução Ginga-J e a Máquina de Apresentação Ginga-NCL.

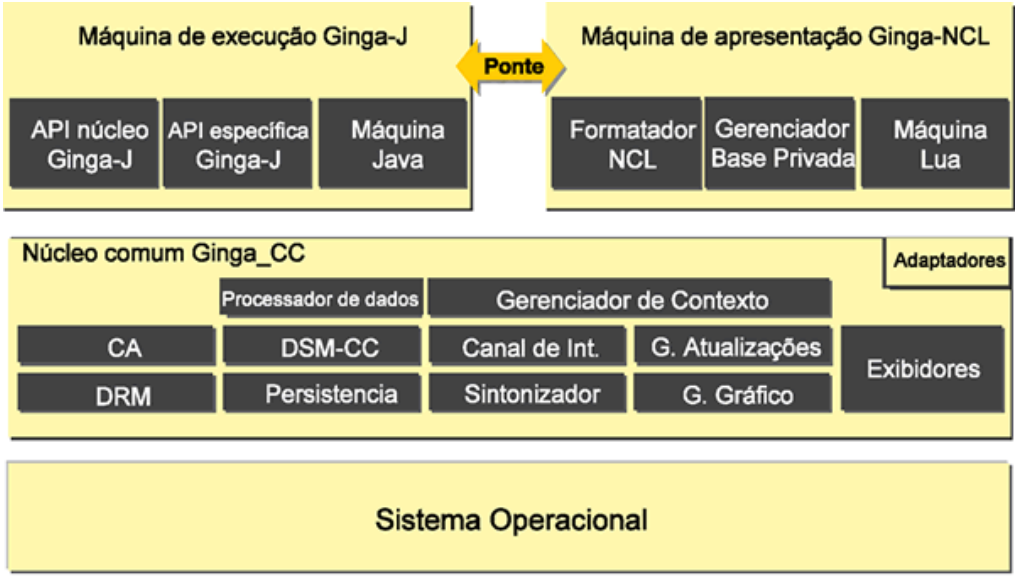

Figura 2.8 – Arquitetura do *Middleware* Ginga. [13]

O *middleware* apresenta duas máquinas de aplicações declarativas e imperativas nos receptores fixos e móveis. Há dois tipos de aplicações, as chamadas declarativas e as procedurais. Um conteúdo declarativo deve ser baseado em uma linguagem declarativa, isto é, em uma linguagem que enfatiza a descrição declarativa do problema, ao invés da sua decomposição em uma implementação algorítmica. Um conteúdo procedural deve ser baseado em uma linguagem não declarativa. [9]

Linguagens não declarativas podem ser linguagens baseadas em módulos, orientadas a objetos. Na literatura especializada usa-se o termo procedural para representar todas as linguagens que não são declarativas [10]. Nessa linguagem o programador possui um maior poder sobre o código, estabelecendo o fluxo de controle e execução de seu programa. [11]

A Figura 2.8 apresenta um ambiente de execução comum às propostas de middleware para TV digital. O Ambiente apresenta uma Máquina de Execução para aplicações procedurais, uma Máquina de Apresentação para aplicações declarativas e Elementos-ponte, que fazem o mapeamento bidirecional entre objetos procedurais e declarativos.

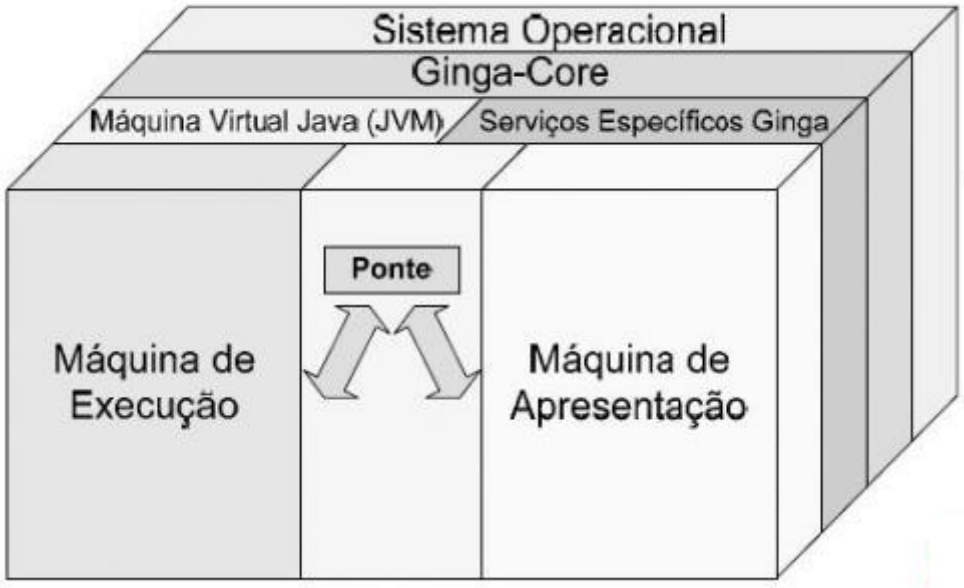

Figura 2.9 - Arquitetura em Alto Nível do *Middleware* Ginga. [11]

Um componente-chave do Ginga-NCL é o mecanismo de decodificação do conteúdo informativo (NCL *formatter*), e o mecanismo LUA, que é responsável pela interpretação dos *scripts* LUA, programação procedural com poderosas construções para descrição de dados baseadas em tabelas associativas e semânticas extensíveis. [12][13]

2.3.2 Ginga NCL - *Nested Context Language*

Ginga-NCL, ambiente obrigatório para receptores portáteis e fixos, é um subsistema lógico do sistema Ginga, responsável pelo processamento de documentos NCL. Um componente-chave do Ginga-NCL é a máquina de interpretação do conteúdo declarativo (formatador NCL). Outros módulos importantes são o exibidor (*user agent*) XHTML e a máquina de apresentação Lua, que é responsável pela interpretação dos scripts Lua.

NCL é uma das principais linguagens existentes para a definição do sincronismo temporal. Como vantagem adicional, e imprescindível em um sistema de TV digital, NCL também provê suporte a variáveis, que podem ser manipuladas através de código procedural, entre eles o de sua linguagem de script Lua, quando há necessidade de alterações na lógica de programação.

NCL foi concebida de forma modular. Módulos agrupam de modo coerente, elementos e atributos XML que possuam alguma relação semântica entre si. Essa estruturação permite que as funcionalidades da linguagem sejam reunidas de acordo com as necessidades de uma determinada aplicação. Sendo assim, um subconjunto das funcionalidades de NCL foi reunido para compor um perfil apropriado ao desenvolvimento de programas de TV não-lineares.

Composer é um ambiente de autoria voltado para a criação de programas NCL para TV digital interativa. Nessa ferramenta, as abstrações são definidas em diversos tipos de visões que permitem simular um tipo específico de edição (estrutural, temporal, leiaute e textual). Essas visões funcionam de maneira sincronizada, a fim de oferecer um ambiente integrado de autoria.[10]

2.3.3 Ginga – Java

Ginga-J é uma especificação de *middleware* distribuído, que reside em um dispositivo Ginga (dispositivo que embarque o *middleware* Ginga – um receptor de televisão digital), com possibilidade de possuir componentes de software nos dispositivos de interação (celulares, PDA etc). Um dos principais objetivos do Ginga é a interação com dispositivos portáteis. Mais do que apenas transmitir para esses dispositivos o Ginga-J deve também ser capaz de receber e interpretar os dados dos celulares, PDAs, controles, etc, para que haja interação com o usuário.

O Ginga-J é responsável pela máquina da apresentação imperativa e utiliza a linguagem Java. O Ginga-J está dividido em três módulos: a máquina virtual Java; o núcleo e suas API; e o módulo responsável pelo suporte às API específicas do Ginga-J.

A definição Ginga-J é composta por API (Interfaces de Programação de Aplicativos) projetadas para suprir todas as funcionalidades necessárias para a implementação de aplicativos para televisão digital, desde a manipulação de dados multimídia até protocolos de acesso.

2.3.4 Ginga – Common Core

O núcleo comum Ginga (*Ginga Common Core*) concentra serviços necessários tanto para a máquina de apresentação (declarativo) quanto para a máquina de execução (procedural). Esse subsistema faz a interface direta com o sistema operacional, fazendo uma ponte estreita com o hardware. É nele onde é feito o acesso ao sintonizador de canal, ao sistema de arquivos, terminal gráfico, dentre outros.

É composto pelos decodificadores de conteúdo comuns e por procedimentos para obter conteúdos transportados em fluxos de transporte (*transport streams*) MPEG- 2 e através do canal de interatividade. Decodificadores de conteúdo comuns servem tanto às aplicações procedurais quanto às declarativas que necessitam decodificar e apresentar tipos comuns de conteúdo como PNG, JPEG, MPEG e outros formatos. O núcleo comum Ginga também deve obrigatoriamente suportar o modelo conceitual de exibição, conforme descrito na ABNT NBR 15606-1. [10]

O *middleware* Ginga através das linguagens oferecidas, e bibliotecas de funções, serão utilizados para os testes da seção 5. Na seção 4 os requisitos de segurança são detalhados para o envio de informações com segurança.

3 SEGURANÇA DA INFORMAÇÃO

O termo Segurança da Informação significa proteger uma informação ou um sistema de informação do acesso, uso, divulgação, modificação ou destruição não autorizada; provendo integridade, confidencialidade e disponibilidade. A garantia de integridade significa proteger o conteúdo contra modificação ou destruição imprópria do conteúdo, inclui ainda a garantia da autenticidade da fonte da informação e o não repúdio pelo seu expedidor. A confidencialidade é relacionada à privacidade, restringindo o acesso e divulgação das informações. Já a característica de disponibilidade garante acesso à informação quando esta for necessária de maneira rápida e precisa. A segurança na televisão digital terrestre, nada mais é do que a aplicação destes conceitos aos casos de uso da televisão digital [14].

3.1 REQUISITOS DA SEGURANÇA DA INFORMAÇÃO

Os principais objetivos são assegurar a proteção da informação, via mecanismos de controle, contra possíveis ameaças existentes – ataques (ação intencional), acidentes (mau uso), defeitos ou falhas – que ocorram onde a informação estiver sendo criada, processada, armazenada ou transmitida. A Segurança da Informação também possibilita saber o quanto o sistema é resiliente, ou seja, o poder de recuperar-se após uma falha ou um ataque.

Os requisitos da segurança da informação são:

- **Integridade:** tem como objetivo principal a proteção à exatidão e complexidade da informação e dos métodos de processamento. Esse requisito protege a informação contra qualquer alteração não autorizada. A violação da integridade pode ser o primeiro passo de um possível ataque e ainda alterar a confiabilidade do dado e/ou sistema;
- **Confidencialidade:** o objetivo desse requisito é assegurar que a informação esteja acessível somente às pessoas autorizadas, refere-se proteção aos dados e sistemas para não serem expostos aos usuários não autorizados. O impacto da violação da confi-

dencialidade das informações ou sistemas podem expor publicamente dados pessoais, corporativos e de governo indevidamente;

- **Disponibilidade:** assegurar aos usuários autorizados o acesso à informação e aos ativos associados. Este requisito se refere ao fato do dado e ou sistema estar liberado para ser acessado pelo usuário no tempo correto;
- **Autenticidade:** é a garantia de que a identidade alegada ou atribuída ao usuário da informação seja verdadeira. A correta identificação do usuário ou ponto de origem (dispositivo) também retrata o requisito autenticidade;
- **Responsabilidade:** permite que as ações de uma determinada entidade em questão sejam rastreadas (imputabilidade) com impossibilidade de sua repudiação, negação ou retratação;
- **Privacidade:** é o direito do usuário em restringir o conhecimento e o acesso aos seus dados pessoais. Uma das formas de preservar a privacidade nas suas transações eletrônicas e ou aplicações é assegurar o anonimato do usuário. A privacidade é considerada como um aspecto de sigilo ou confidencialidade. [14]

3.2 A SEGURANÇA EM SERVIÇOS DA TELEVISÃO DIGITAL INTERATIVA

O usuário pode receber um sinal de áudio e vídeo com qualidade superior, possibilitando que ele tenha a interatividade com diversos serviços interativos, associados ou não a programação. São exemplos de aplicações interativas: distribuição de imagem em eventos esportivos, placar em tempo real, legendas e áudios em vários idiomas, previsão do tempo, indicativos financeiros, entre outras.

Para que seja assegurada a proteção do receptor/conversor alguns aspectos devem ser tratados, como em outros sistemas embarcados, a segurança nas plataformas móveis e segurança em transações em aplicações onde há a troca de informações pessoais.

Na Figura 3.1 é possível verificar que o receptor de TVDi recebe o canal de radiodifusão de diversas emissoras/programas, que contém dados de interatividade do Canal de Descida no feixe de transporte (TS). Para a troca de informações com segurança os dados do usuário devem ser protegidos antes de serem enviados para a emissora. [15]

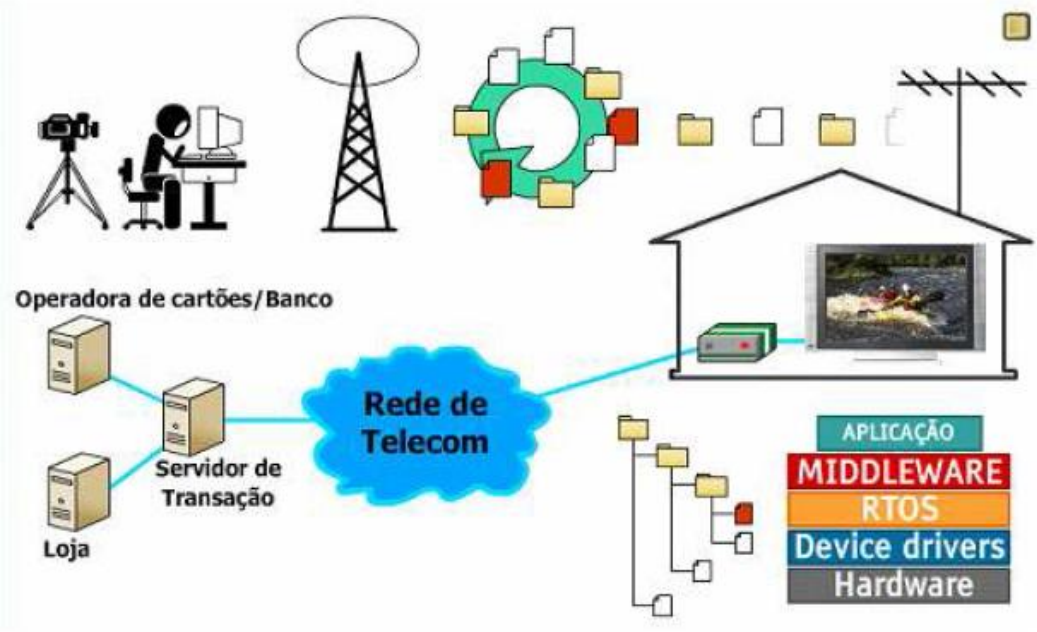

Figura 3.1 – Difusão de Aplicações em Televisão Digital. [14]

As relações comerciais entre empresas por meio eletrônico foram potencializadas pelos cartões de crédito, *internet* e a globalização. A segurança da troca de informações não é mais preocupação apenas das empresas, mas do cliente ou consumidor. A segurança da informação é parte estratégica do negócio ou serviço podendo traduzir em lucro, aumento de competitividade e fator de redução de perdas.

3.3 SEGURANÇA DO PONTO DE VISTA DO USUÁRIO

Para o usuário ou telespectador, a segurança é visto como um item que não causa muita preocupação, pois a televisão pode ser considerada um meio que não oferece tantos riscos como o computador. Mas os fatores de riscos a que a televisão digital estará exposta serão mais frequentes do que se pode imaginar, como está ocorrendo com os telefones celulares que suportam tecnologia *Android*.[17]

O usuário comum não tem um vasto conhecimento sobre segurança e mesmo que a sua consciência ao longo do tempo tenha sido aumentada pelas tantas ocorrências de invasões de *worms* e

vírus, ainda fica a pergunta se o usuário é capaz de diferenciar tantos produtos de segurança e se sabe utilizá-los da forma correta.

A importância de se tratar da segurança nos aplicativos de televisão digital fica evidente quando se compara os valores dos televisores e set-top-box. As soluções de segurança devem estar incorporadas de forma a facilitar a gestão dos aplicativos, facilitando a escolha e possibilitando que o usuário seja responsável pelo *download* e execução de aplicativos não certificados pelo Sistema.

A segurança estando incorporada ao *middleware* facilitaria a execução dos aplicativos tanto para os usuários que precisam de um sistema seguro tanto para os outros que não queiram perder tempo com esta questão.

3.3.1 Sistemas de verificação de vulnerabilidades

Sistemas operacionais e demais programas de suporte, tais como o navegador da *Internet*, máquina Java, frameworks e players, devem ser mantidos sempre atualizados. Normalmente eles possuem algum tipo de controle interno, que pode ser centralizado, informando ou até mesmo atualizando de vez o programa. Incluem-se nesta categoria os programas de proteção, tais como antivírus e *antispywares*. Existem ainda sistemas capazes de verificar se alguma porta do *hardware* está desprotegida e se as configurações do sistema operacional e da rede estão compatíveis com os requisitos de segurança até então conhecidos [17]. A execução destes programas deve ser feita por quem entende as operações por eles propostas. A integração entre os próprios sistemas da organização podem prover maior segurança ao conjunto.

Os ataques contra a confidencialidade resultar na liberação de informação não autorizada para fins de divulgação ou fraude. Ataques contra a integridade irão contra a confiabilidade da informação. E ataques contra a disponibilidade irão contra o suporte ao serviço ou a destruição da informação. Seja como for, com certeza as maiores ameaças estão dentro da própria organização, de forma que um sistema de *firewall* pode não ser suficiente [17].

A Tabela 3.1 mostra as partes da cadeia de valor da Televisão Digital, as vulnerabilidades e as ações de seguranças pertinentes a cada uma delas:

Tabela 3.1 – Necessidade de Segurança no Sistema de Televisão Digital. [17]

Vulnerabilidade	Parte do Sistema de Televisão Digital	Necessidade de Segurança
Transação Fraudulenta Perda/Roubo de Conteúdo	Usuário Final	Privacidade e integridade dos dados pessoais e execução segura de *software* baixado e instalado
Pirataria de Conteúdo	Provedor de Conteúdo	Proteção de Conteúdo e Gestão de direitos autorais
Falsificação, violação ou corrupção das aplicações	Provedor de Aplicações	Comunicação segura fim-a-fim Irretratabilidade e autenticidade
Uso ilegítimo do serviço	Provedor de Serviço	Acesso seguro a rede
Pirataria de *Software* e clonagem de Hardware	Fabricante/Integrador de Hardware e *Software*	Proteção da propriedade Intelectual
Instruções Maliciosas	Informação viaja através de redes sem fios, que podem ser menos seguras que as redes físicas.	Interceptação de informação resultando em uma captura de dados, comprometendo a integridade do usuário e ações legais.
Dispositivos que usam várias redes	Portabilidade do dispositivo cria o abandono da segurança imposta pela empresa e expõe os seus protocolos de segurança	Propagação de *malware* pode resultar em perda de informações, corrupção de dados e indisponibilidade.
Falta de política de acesso aos dispositivos	Quando a empresa não controla o dispositivo e a gestão é feita pelo usuário	Propagação de *malwares* e perda de informações.
Instalações de Aplicativos de terceiros	O dispositivo permite a instalação de aplicativos, a empresa não controla, a gestão é feita pelo usuário.	Propagação de *malwares*, perda de informações e invasão na rede da empresa.

Cada participante da cadeia de valor da Televisão digital tem a sua necessidade de segurança, desde a produção de *hardware*, o produtor de conteúdo de televisão até a oferta de serviços interativos ao usuário final. Em cada parte há uma necessidade de segurança, como em qualquer sistema embarcado. O sistema proposto na seção 4 destaca como o sistema pode ficar vulnerável e as possíveis formas de ataque que devem ser levadas em consideração para manter o sistema seguro.

4 SISTEMA PROPOSTO

No padrão televisão digital brasileiro, a metodologia é implementada a partir do modelo *top-down* e permite a interação de cinco componentes genéricos do sistema cliente, o provedor, o controlador, o *backend* e o servidor de aplicações. Protocolos de segurança são implementados para a autenticação da interação entre os componentes criando um ambiente seguro para a transação de informações, detalhes como essa interação é estabelecida será

4.1 ARQUITETURA

A arquitetura proposta na metodologia segue o modelo *top-down*, apresentado na Figura 4.1 que envolve os vetores de ataque, definição das falhas e do controle de segurança, e a análise dos impactos. O modelo que permite a identificação de riscos mediante a análise das falhas, vulnerabilidade e ameaças ao sistema.

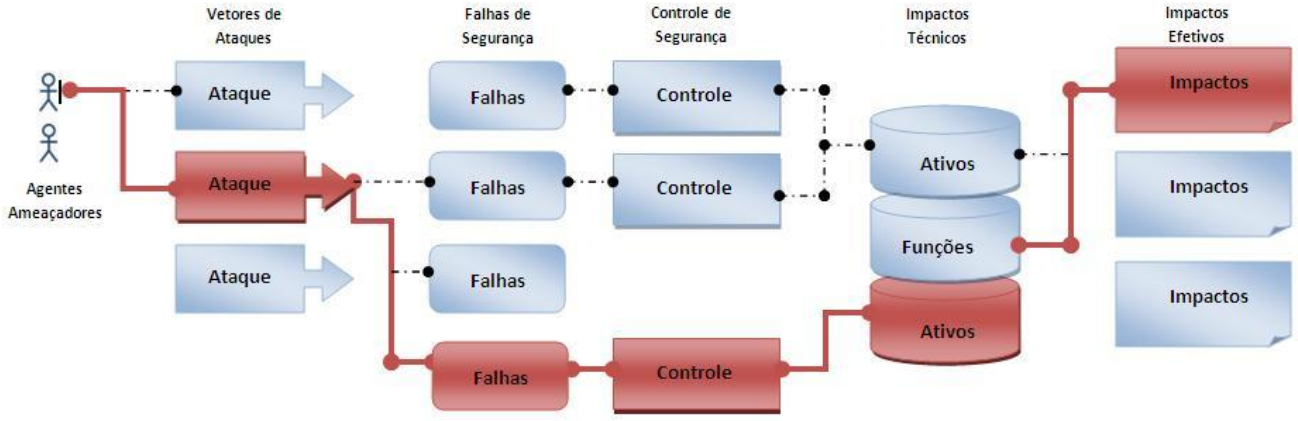

Figura 4.1 – Metodologia de Identificação de Riscos. [19]

As ameaças ou vetores de Ataque podem potencialmente usar muitos caminhos diferentes através de seu aplicativo para causar falhas na segurança. Cada um desses caminhos gera falhas ou danos à segurança que pode, ou não ser grave o suficiente para justificar a atenção. O controle de segurança é usado para encontrar e explorar os caminhos utilizados pelos vetores de ataque e podem ser extremamente difíceis. Para determinar o risco para a aplicação, pode-se avaliar a probabili-

dade associada a cada agente de ameaça, vetor de ataque, e a fraqueza de segurança e combiná-lo com uma estimativa do impacto técnico e de negócios efetivos. [19]

Ataques são as técnicas que os vetores de ataque usam para explorar as vulnerabilidades das aplicações. Os ataques podem ser facilmente confundidos com as vulnerabilidades, mas estes não são fraquezas do sistema, mas tentativas de explorar estas fraquezas.

Para a criação das aplicações interativas, o *middleware* oferece uma linguagem de programação, que foi testada usando vulnerabilidades conhecidas de outros sistemas, possibilitando a identificação de falhas e pontos onde o sistema se torna inseguro, colocando em risco as informações do usuário. As aplicações foram expostas as vulnerabilidades e identificou-se uma rotina para que a programação seja feita de forma segura.

O Ginga é o *middleware* que possibilita o desenvolvimento de aplicações interativas para o Sistema Brasileiro de TV Digital Terrestre (SBTVD). As aplicações para TV digital podem ser divididas em dois conjuntos de acordo com o tipo de conteúdo inicial: declarativas e procedurais. Ginga provê suporte para ambas as modalidades sendo a primeira através do módulo Ginga-NCL [19] e a segunda através do módulo Ginga-J [20].

Uma aplicação que use o carregamento de páginas JSP (*Java Server Page*) pode ser utilizado como objeto de estudo para a avaliação dos parâmetros de segurança da informação para um aplicativo de TV digital. Linguagens declarativas provêem um maior nível de abstração para o desenvolvedor, porém por enfatizar um domínio restrito, são mais restritas do que as procedimentais. É possível a construção de aplicações híbridas em que os dois tipos de conteúdo coexistem e se referenciam. [20]

Apesar de a tecnologia JSP ser comumente usada para gerar conteúdo em HTML, ela pode gerar também qualquer tipo de documento textual, inclusive *scripts* NCL.

4.2 VIAS DE INFECÇÃO PARA O SISTEMA DE TELEVISÃO DIGITAL

As pesquisas com os sistemas para computadores pessoais e *smartphones* quanto à segurança mostra que os incidentes relacionados aos receptores de TVDi aumentarão significativamente. A semelhança entre os aspectos de segurança de *smartphones* e receptores de TVDi vão desde a comparação das técnicas de ataque, as técnicas de segurança e proteção. [21]

A segurança para estes sistemas pode ser dividida em três pilares. Primeiro, a entrega segura de aplicações, verifica-se a integridade e a autenticidade de origem de uma aplicação a ser instalada. Segundo, níveis de confiança que determinam graus de segurança e privilégios e são implementados por mecanismos de controle de acesso. Terceiro, o isolamento de aplicações e do Sistema Operacional, como uma forma de prevenção contra o comprometimento de outras aplicações ou o próprio Sistema Operacional. Todas as vias de riscos possíveis são descritas na Figura 4.2. [21]

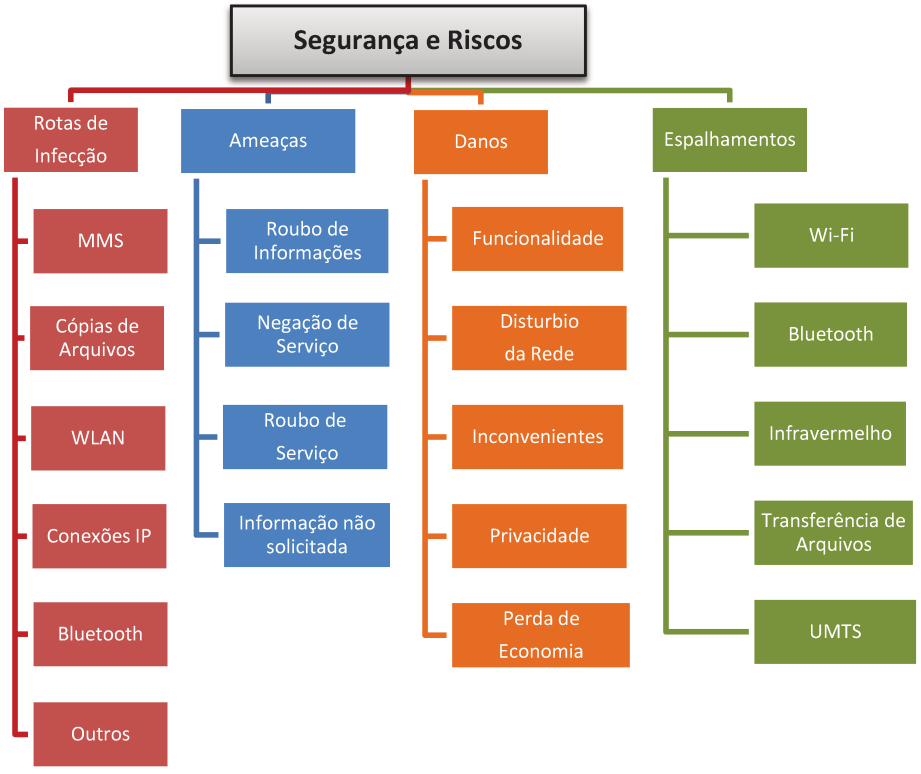

Figura 4.2 – Vias de Infecção, Ameaças, Danos e Vias de Espalhamento de Vírus para o SBTVD [21]

De modo equivalente ao oferecido pelas plataformas de aplicativos para dispositivos móveis, um receptor interativo, com *middleware* Ginga, aderente a norma brasileira, poderá possuir no mínimo as seguintes características de segurança: autenticação de usuário, segurança no canal de inte-

ratividade com SSL/TLS e suporte à autenticação de aplicativos com o mecanismo de controle de acesso correspondente.

4.3 AMEAÇAS E VULNERABILIDADES

As vulnerabilidades são pontos em que o sistema é suscetível a ataques. Considera-se, além das fragilidades do sistema de TVD, erros que nele existam. A identificação das vulnerabilidades técnicas nem sempre é trivial, requerendo, em geral, profundo conhecimento de Tecnologia da Informação e de Comunicação.

Os tipos de ameaças e vulnerabilidades irão variar conforme o ambiente interno e externo. A infraestrutura de dados e de comunicação utilizada, a organização dos processos, a cultura de segurança dos usuários, o apoio da direção à política de segurança da informação, a competitividade do mercado, a visibilidade da organização, estes são fatores a serem considerados.

Ao detectar as ameaças e vulnerabilidades relacionadas aos sistemas de informação pode-se identificar o impacto que as perdas de confidencialidade, integridade e disponibilidade podem causar aos usuários ou as organizações.

Contudo, diversas são as vulnerabilidades associadas a procedimentos ou ao comportamento humano. A questão das senhas é um bom exemplo. É fácil entender que, quanto mais complexas as senhas, mais difícil se torna a descoberta delas. Porém, na mesma proporção, mais difícil se torna decorá-las. [15]

4.3.1 *Vetores de Ataque*

Os vírus para novos dispositivos são semelhantes aos vírus anteriormente desenvolvidos para computadores. Analogamente aos vírus para dispositivos móveis pode-se agrupar os tipos de ameaças e vulnerabilidade como:

Ad/*Spyware* – este tipo de programa executa varias ações, sem o conhecimento do usuário. Por ter uma licença chamada EULA (*End User Licence Agreement*), onde o perfil do usuário é verificado, obtendo informações pessoais e suas preferências.

- ***Bluebug*** – O Bluetooth é o alvo uma vez que explora a sua vulnerabilidade na segurança podem usá-lo para fazer ligações de valor agregado e também pode usufruir da internet do usuário.
- ***Buffer-overflow*** – quando um buffer ultrapassa a sua capacidade máxima de armazenamento, os programas serão armazenados em regiões próximas de memória corrompendo dados e danificando o programa.
- **Força Bruta** – Cria uma lista de possíveis soluções que possa satisfazer o problema, verificando cada uma delas, são mais usuais os ataques a login e senha de acesso.
- ***Denial-of-Service*** (DoS) – usado para prejudicar e/ou negar acesso seja este a um dispositivo, serviço ou mesmo da rede.
- ***Exploit*** – explora uma vulnerabilidade do sistema permitindo a execução de ações indesejadas.
- ***Hacking Defaults*** – utiliza acessar dispositivos por meio de software no qual se usa a senha padrão permitindo que se alterem as configurações originais do sistema.
- ***Mobile Malware*** – destinado a afetar dispositivos móveis e software.
- ***Rogue Software*** – usado para induzir o usuário a comprar um software com a finalidade de proteger e/ou eliminar vírus em potencial. Depois da sua execução, erros em resultado após analises do sistema, cria mensagens que forçam o usuário a comprar um produto sem a necessidade.
- ***Payload*** – outro tipo de ameaça pode ser usado para a instalação do código malicioso, um Trojan pode ser hospedeiro de um Rogue Software, onde este é um payload do ataque.
- ***Trojan*** – software que se disfarça em algo que não é, mas não se replica.
- **Vírus** – software que infecta um arquivo contaminando o sistema.
- ***Worm*** – software malicioso que cria cópia de si mesmo. [16]

4.4 ROTAS DE INFECÇÃO

- **MMS (*Multimedia Message System)*** – Software malicioso pode se espalhar mensagens MMS, anexando uma cópia de si mesmo para uma mensagem e enviando para qualquer outro dispositivo capaz de receber MMS. Um bom exemplo é um *worm* chamado Commwarrior, que pode procurar a lista de telefone e depois se espalhar por mensagens MMS. [21]
- ***Bluetooth*** – Os primeiros vírus de celular se espalharam através de *Bluetooth*. Se o vírus pode ser uma parte de um arquivo, e este for trocado via *Bluetooth* pode espalhar de forma bastante eficiente.
- **Conexões IP através de UMTS– (*Universal Mobile Telecommunication System*)** – Normalmente as conexões de internet feitas por celulares. As ligações são temporárias e a conectividade é utilizada apenas para um pedido explícito. Uma conexão IP é aberta apenas quando é necessário. No entanto, isso pode mudar no futuro, se as ligações WLAN e Serviços 3G.
- **Cópia de arquivos** – Muitos vírus se espalham para outros arquivos como uma infecção. Esses arquivos podem ser copiados para outros dispositivos, espalhando o vírus.
- **Mídia removível** – Também é possível que um vírus se espalhe sobre uma mídia removível explicitamente como acontecia antigamente com os disquetes.
- **E-mail aplicações** – Aplicativos de e-mail em sua plataforma, sendo que um autor de vírus provavelmente um malware móvel que utiliza anexos ao e-mail para transmitir-se os dispositivos sem fios.
- ***Instant Message* (IM)** – O sequestro de listas de nomes de mensagens instantâneas e o envio de links para destinatários para direcioná-los para sites maliciosos. Vírus pode também enviar mensagens instantâneas com o código malicioso anexado.
- **Navegação em páginas** – Quando a página infectada é acessada usando browser, o código malicioso escondido na página web pode ser executado. Este código malicioso pode penetrar no hardware e causar alguns danos. [21]

4.5 AMEAÇAS

Ameaça não só classifica os ataques conhecidos, mas também ajuda a antecipar o que tipos de ameaças podem vir. No entanto, detecção e sistemas de prevenção para ataques estão ainda no seu início. Portanto, pesquisa de segurança em novas redes será focada em grande parte, em problemas de roteamento, e no protocolo de segurança. Como os vetores de ataque trabalham em seu caminho até a pilha de protocolos para explorar as aplicações, a análise de risco e ameaças deve ser significativa. Alguns exemplos de ataques e objetivos de seguranças são apresentados na Tabela 4.1. [22]

Tabela 4.1 – Exemplos de Ataques. [22]

Objetivo de Segurança	Exemplos de Ataques
Confiabilidade	Roubo de dados, *bluebugging* (é uma forma de *Bluetooth* ataque muitas vezes causada por uma falta de consciência), *bluesnarfing* (acesso não autorizado de informações a partir de um dispositivo sem fio através de um *Bluetooth* de conexão)
Integridade	Roubo do telefone
Disponibilidade	Ataques por Negação de Serviço, drenagem de bateria

- **Roubo de informações** – Hackers atacam frequentemente dispositivos para obter informações. Informação transitória inclui a localização do dispositivo, a sua utilização de energia, e outros dados, normalmente, não graváveis. O *Bluesnarfing* e ataque *Bluebugging* são exemplos. Atualmente, esses ataques dependem em grande parte de configuração do dispositivo *Bluetooth* e outras instalações padrão inseguras. [23]
- **Informação não solicitada** – Os ataques de informação também podem funcionar no sentido oposto. Por exemplo, os atacantes podem direcionar informações aos usuários como publicidade, mensagens e informações não solicitadas. *Bluejacking*, por exemplo, é nada mais do que a criação de um nome de dispositivo *Bluetooth* para anunciar alguma mensagem e deixar que outros usuários descubram.

- **Roubo de serviço** – Alguns *malwares* podem tentar usar os recursos do dispositivo da vítima. Sequestro de recursos do dispositivo feitos por *malware* utilizam recursos das vítimas por um bom tempo.
- **Negação de serviço** – Tentativa de invadir o dispositivo e tentar roubar informações e privilégios são dois tipos de ataques. Estes ataques que esgotam a bateria, mantendo o dispositivo ativo o tempo todo. Um ataque de exaustão de bateria cuidadosamente elaborada leva o usuário a acreditar que a bateria está com defeito. Outro ataque dessa forma é utilizar a bateria até seu esgotamento como um ataque secundário para ampliar o impacto de outro ataque, deixando o sistema mais vulnerável.

4.6 FALHAS OU DANOS

Um vírus móvel pode danificar os seus dispositivos de maneiras diferentes. Na melhor das hipóteses ele pode ocupar espaço de memória e afetar o desempenho de seus dispositivos, causando um comportamento imprevisível, como continuamente o envio de SMS ou MMS, a desativação de funcionalidades básicas, etc. Na pior das hipóteses ele pode alterar os seus dados, apagar arquivos importantes e executar funcionalidades sem permissão no aplicativo. Pode causar ao usuário final prejuízos de tempo e dinheiro para eliminar o problema.

Alguns danos causados por *malware* móvel estão incluídos na Tabela 4.2.

Tabela 4.2 – Exemplos de Falhas. [23]

Tipo de dano	Exemplos
Funcionalidade	Drenagem da Bateria Desabilita os produtos antivírus Impede o acesso a serviços de mensagens Substitui utilitários normais do dispositivo Modifica tela do dispositivo móvel Funcionalidade básica do telefone está desativada

	O desempenho do sistema diminui Arquivos corrompidos causam falha de inicialização
Perda de Economia	Envia mensagens para números de diferentes localidades Continuamente enviar SMS ou MMS Exclui os arquivos de privacidade do usuário
Inconvenientes	Bloqueio do cartão multimídia do dispositivo
Distúrbio da Rede	Ataques de negação de serviço Perda de largura de banda de rede
Privacidade	Roubo de dados Perda de informações confidenciais Dispositivo sequestrado Modificação de dados pessoais Exclusão de arquivos valiosos Maior risco de responsabilidade legal

4.7 PROPAGAÇÃO DE AMEAÇAS

As modalidades de propagação de vírus são inúmeras e pode variar. A utilização do *Bluetooth* pode facilitar a propagação do vírus. Outra forma de propagação pode ser através do envio de mensagens infectadas, abrindo conexões TCP/IP diretamente dos aplicativos e oferecendo, assim, maiores oportunidades para o *malware* a se espalhar. Alguns tipos de epidemias encontrados podem-se redefinir vírus de dispositivos como: um programa que se espalha entre dispositivos inteligentes pela interface de comunicação e podem influenciar no uso de aparelho ou vazar dados pessoais do usuário. [24]

4.8 PROTOCOLOS DE SEGURANÇA

O *middleware* Ginga, análogo as plataformas de aplicativos para dispositivos móveis, deve ter as seguintes características de segurança: autenticação de usuário, segurança no canal de interatividade com SSL (*Secure Sockets Layers*) e o TLS (*Transport Layer Security*), e suporte à autenticação de aplicativos com o mecanismo de controle de acesso correspondente.

O Protocolo TLS oferece serviços de segurança da origem ao destino para aplicações que usam um protocolo de camada de transporte confiável, como o TCP, proporcionando assim, segurança nas transações realizadas na internet. Esse protocolo oferece segurança para transações na *internet*.

O SSL foi desenvolvido para fornecer serviços de segurança e de compressão de dados gerados pela camada de aplicação. O SSL pode receber dados de qualquer protocolo da camada de aplicação, mas é usado principalmente pelo HTTP (*HyperText Transfer Protocol*). Depois de comprimidos, assinados e criptografados os dados são enviados para a camada de transporte, como mostra a Figura 4.3. [25]

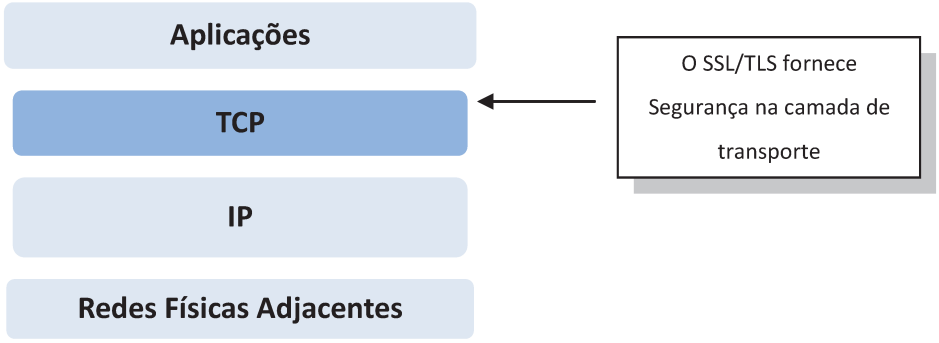

Figura 4.3 – Posição do Protocolo SSL e TSL no Modelo *Internet.* [25].

Quando há uma transação na *internet* esperam-se os seguintes requisitos de segurança:

- **Autenticação de entidades** – o cliente precisa estar seguro de que o servidor pertence ao verdadeiro fornecedor. Da mesma forma, o fornecedor precisa autenticar o cliente.

- **Integridade de mensagem** – o cliente e o fornecedor precisam ter certeza que o conteúdo da mensagem não será modificado durante a transação.
- **Confidencialidade** – o cliente e o fornecedor têm que ter certeza que não haverá interceptação de informações confidenciais, como dados pessoais do cliente e do fornecedor.

4.8.1 SSL (*Secure Sockets Layers*)

Os dados recebidos pela camada de aplicação são comprimidos, assinados e criptografados. O SSL pode receber dados de qualquer protocolo da camada de aplicação, mas o protocolo HTTP é o mais comum.

As fases para tratamento da segurança utilizando SSL são:

- **Fragmentação** – Primeiro, o SSL divide os dados em blocos de 214 *bytes* ou menos.
- **Compressão** – Cada fragmento de dados é comprimido com um método de compressão sem perdas, entre o cliente e servidor, este serviço é opcional.
- **Integridade de Mensagens** – O SSL usa uma função *hash* com chaves para criar um MAC, preservando a integridade dos dados.
- ***Framming*** – Um cabeçalho é adicionado ao *payload* criptografado, e este é passado para um protocolo de camada de transporte confiável.
- **Confiabilidade** – Os dados originais e MAC são criptografados utilizando a criptografia de chave simétrica. [25]

4.8.2 Parâmetros de Segurança

A assinatura digital obtida através do uso da criptografia assimétrica ou de chave pública infelizmente não pode ser empregada, na prática, de forma isolada, do modo como foi didaticamente descrito no item anterior. Um mecanismo para o uso adequado da assinatura digital é a função *Hashing*. Sua utilização como componente de assinaturas digitais se faz necessário devido à lentidão dos algoritmos assimétricos, em geral cerca de mil vezes mais lentos do que os simétricos.

Assim, na prática é inviável e não produtivo utilizar algoritmos puros de chave pública para assinaturas digitais, principalmente quando se deseja assinar grandes mensagens, que podem levar preciosos minutos ou mesmo horas para serem integralmente "cifradas" com a chave do usuário. Ao invés disso, é empregada uma função *Hashing*, que gera um valor pequeno, de tamanho fixo, derivado da mensagem que se pretende assinar, de qualquer tamanho. Assim, a função *Hashing* oferece agilidade nas assinaturas digitais, além de integridade confiável.

Também denominada *Message Digest, One-Way Hash Function,* Função de Condensação ou Função de Espalhamento Unidirecional, a função *Hashing* funciona como uma impressão digital de uma mensagem gerando, a partir de uma entrada de tamanho variável, um valor fixo pequeno: o *digest* ou valor *hash*.

Este valor está para o conteúdo da mensagem assim como o dígito verificador de uma conta-corrente está para o número da conta ou o *check sum* está para os valores que valida. Serve, portanto, para garantir a integridade do conteúdo da mensagem que representa. Assim, após o valor *hash* de uma mensagem ter sido calculado através do emprego de uma função *hashing*, qualquer modificação em seu conteúdo - mesmo em apenas um bit da mensagem - será detectada, pois um novo cálculo do valor *hash* sobre o conteúdo modificado resultará em um valor *hash* bastante distinto.

A chave secreta é para criar segredos de comprimento variável aplicando o mesmo conjunto de funções *hash* e pré-anexando constantes diferentes.

Existe ainda um algoritmo que produz a função **MD5** é uma função de espalhamento unidirecional, a sigla MD significa *Message Digest.* Este algoritmo produz um valor *hash* de 128 bits, para uma mensagem de entrada de tamanho arbitrário. O algoritmo foi projetado para ser rápido simples e seguro. Seus detalhes são públicos, e têm sido analisados pela comunidade de criptografia. Entretanto, o fato dele produzir um valor *hash* de somente 128 bits é o que causa maior preocupação; é preferível uma função *Hashing* que produza um valor maior, aumentando a confiabilidade do sistema. Essa fraqueza não afetou a segurança global do algoritmo. [25]

4.8.3 Protocolos SSL

De acordo com a Figura 4.4, o protocolo SSL define quatro protocolos em duas camadas. Descreve-se a seguir os quatro protocolos.

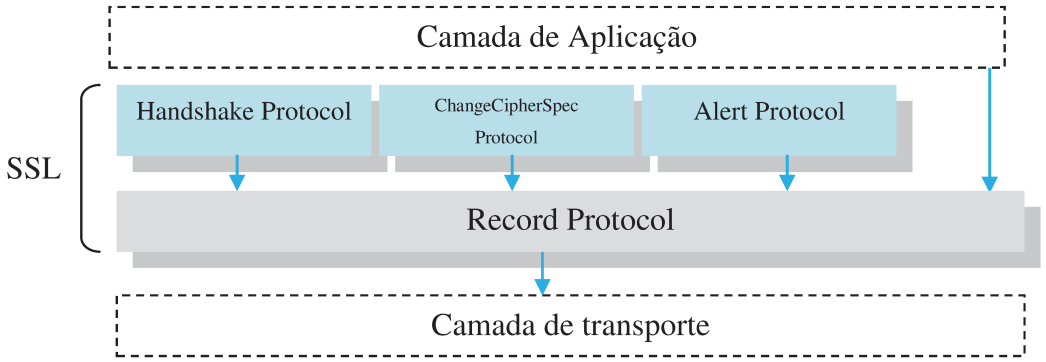

Figura 4.4 – Protocolos SSL[25].

Handshake Protocol – fornece parâmetros de segurança, usados para negociar o conjunto de cifras, a fim de autenticar o servidor perante o cliente e o cliente perante o servidor. A troca de informações visa à formação de segredos criptográficos. A Figura 4.5 mostra a Estrutura do Protocolo de *Handshanking*.

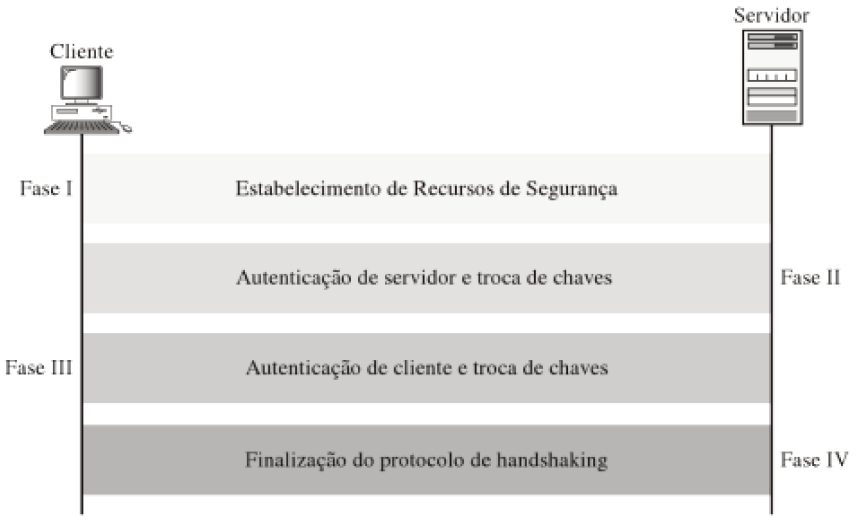

Figura 4.5 – Protocolo de *Handshanking*. [25]

ChangeCipherSpec Protocol – sinaliza com prontidão os segredos criptográficos.

Alert Protocol – informa a ocorrência de erros e condições anormais, descreve o problema e seu nível (aviso ou fatal).

Record Protocol – transporta mensagens de camada superior e é comprimido usando o protocolo *hash* negociado. O fragmento comprimido e o MAC são criptografados. O cabeçalho SSL é acrescentado à mensagem e enviado pelo emissor. No receptor o processo é o inverso. [25]

4.9 PROTOCOLOS DE SEGURANÇA PARA A LINGUAGEM LUA

Como ocorrem com as outras vulnerabilidades de programação insegura, as vulnerabilidades associadas ao mau uso de criptografia não são detectadas por ferramentas de varredura externas de vulnerabilidades. Ferramentas de análise estática de código fonte podem detectar o uso de APIs criptográficas conhecidas, mas não podem detectar se a API criptográfica está sendo usada de forma adequada.
No caso de Lua, a melhor maneira de verificar se uma aplicação cifra adequadamente os dados sensíveis e tem um bom mecanismo e gerenciamento de chaves é a analise dos códigos de programação das funções e bibliotecas.
Para a programação LUA há três bibliotecas criptográficas que estão disponíveis publicamente, LuaCripto, LuaSec e LuaMD5. [17]

4.9.1 LuaCripto

LuaCripto é uma biblioteca Lua para as funções criptográficas do OpenSSL. Isto pode significar que o LuaCripto não seria mais seguro que o OpenSSL subjacente. Porém ele pode ser mais inseguro, ao fazer uso de partes inseguras e eliminar partes seguras [17].

4.9.2 LuaSec

LuaSec é um encapsulamento das rotinas OpenSSL para comunicação segura com SSL/TLS. Assim como no caso anterior, isto pode significar que o LuaSec não seria mais seguro que o OpenSSL subjacente. Porém ele pode ser mais inseguro, ao fazer uso de partes inseguras e eliminar partes seguras [17] É preciso uma conexão TCP já estabelecidas e cria uma sessão segura entre os seus pares. LuaSec precisa de um conjunto de informações (tais como protocolo, chave, certificado, etc) para quebrar a conexão TCP.

4.9.3 LuaMD5

O LuaMD5 é uma biblioteca criptográfica simples para scripts Lua. A Comunidade de Usuários Lua (Lua-users.org) contém algumas recomendações curtas e genéricas sobre o uso de APIs Lua consideradas perigosas e sobre o uso de sand-boxing como estratégia de contenção de código.
O MD5 oferece instalações básicas de criptografia para Lua 5.1: um *hash* (digerir) função, um par cripta / descriptografar com base em MD5 e CFB, e um para cripta / descriptografar com base no DES com chaves de 56 bits.

4.10 INTERAÇÕES

Para a criação das aplicações interativas, no padrão brasileiro de TVD, inicialmente o conteúdo da aplicação é serializado na forma de um carrossel de objetos no padrão DSM-CC (*Digital Storage Media Command and Control*), onde os arquivos e pastas da aplicação são codificados em sessões e encapsulados em um fluxo (*stream* MPEG2-TS) (ITU 2000). Após a codificação dos dados, propriedades da aplicação como nome, tipo, classe principal e outras características são definidas e estruturadas na forma da tabela AIT (*Application Information Table*) e encapsulados em pacotes TS. Terminada a preparação dos dados, ocorre a configuração da tabela PMT (*Program Map Table*) com o PID utilizado pelo TS de dados (*Object Carousel*) e o PID da AIT, além

da adição dos descritores necessários para identificar a existência de um fluxo de dados para um determinado programa ou serviço. Por fim, o *stream* é multiplexado com outros fluxos de áudio, vídeo e dados para então ser transmitido em broadcast pela emissora de TV. [26]

No padrão televisão digital brasileiro, a metodologia é implementada a partir da interação de cinco componentes genéricos: Cliente, o Provedor, o Controlador, o *Backend* e o Servidor de Aplicações, como se mostra na Figura 4.6.

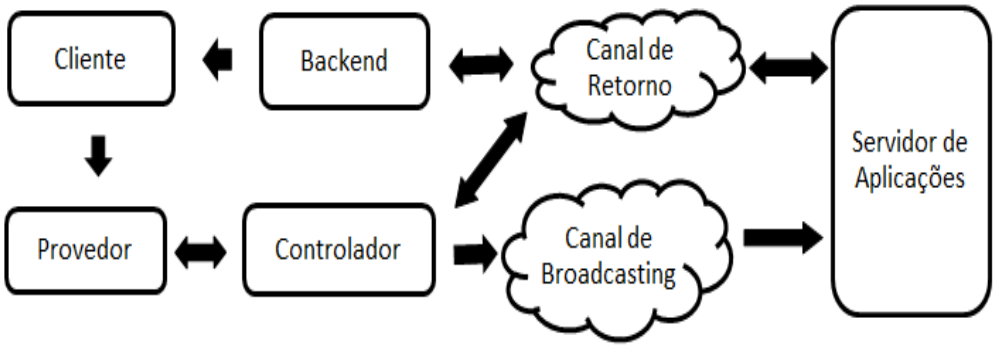

Figura 4.6 – Interações para Aplicações Interativas. [26]

Os dados transmitidos entre o cliente e o provedor requerem a autenticação do cliente pelo provedor e a confiabilidade na comunicação entre eles, de forma a proteger os dados de execução e os dados enviados para o provedor.

Os dados transmitidos entre o provedor e controlador precisam também passar pela autenticação e garantir a confiabilidade na troca de mensagens de controle.

A aplicação requisitada pelo cliente precisa de confiabilidade para estabelecer comunicações seguras entre o controlador e o *backend*, respectivamente. Por isso, todas as partes do sistema devem ser devidamente identificadas.

Como o receptor de televisão digital é um componente volátil e não conhecido previamente, a sua autenticação deve ser tratada de forma diferenciada, não podendo aplicar as formas de autenticação mais tradicionais.

Os testes foram realizados no *middleware* e utilizam vulnerabilidades conhecidas de outros sistemas, possibilitando a identificação de falhas e pontos onde o sistema se torna inseguro, colocando em risco as informações do usuário. Os protocolos de certificação SSL (*Secure Socket Layer*) são utilizados para a certificação da aplicação e entrega dos pacotes de forma segura na camada de aplicação.

O protocolo de segurança SSL será implementado na seção 5 em uma aplicação de TVDi e testada no *middleware* Ginga.

5 TESTES E RESULTADOS

Para os testes foram utilizados o protocolo de segurança SSL, descrito na seção anterior em uma aplicação de TVDi.

O NCLua HTTP implementa alguns dos principais recursos do protocolo HTTP/1.0. Ele é um módulo escrito inteiramente em linguagem Lua para ser utilizado em *scripts* NCLua. Pela simplicidade do protocolo HTTP, o módulo possui apenas algumas funções que permitem a geração de requisições e tratamento de respostas.

Na Figura 5.1 apresenta-se o modelo proposto para os testes com realização de uma conexão TCP no Ginga-NCL. A aplicação estabelece uma conexão a um servidor, envia uma requisição e fica aguardando o retorno.

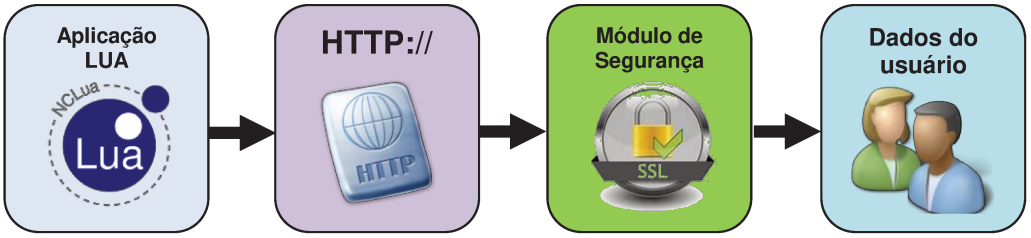

Figura 5.1 – Esquema de Implementação da Aplicação NCLua-HTTP com Segurança da Informação.

As implementações agilizam a construção de aplicações com interatividade realizando acesso ao canal de retorno por meio de protocolos padronizados. Existem diversas categorias de aplicações interativas que podem ser construídas com uso dos protocolos implementados, tais como jogos, informações (notícias, previsão do tempo, etc), educação (*t-learning*), governo eletrônico (*t-government*), comércio eletrônico (*t-commerce*), saúde (*t-health*), bancárias (*t-banking*) e outras.

5.1 APLICAÇÕES

A aplicação utiliza a norma ABNT do Ginga-NCL (NBR 15606-2) que define uma classe “TCP” para realização de requisições com o protocolo a partir de *scripts* NCLua (*scripts* Lua embutidos em documentos NCL). O protocolo HTTP é um protocolo de camada de aplicação, que é trafegado utilizando TCP, a primeira parte dos testes basea-se na aplicação NCLua com conexão HTTP, conforme mostra a Figura 5.2.

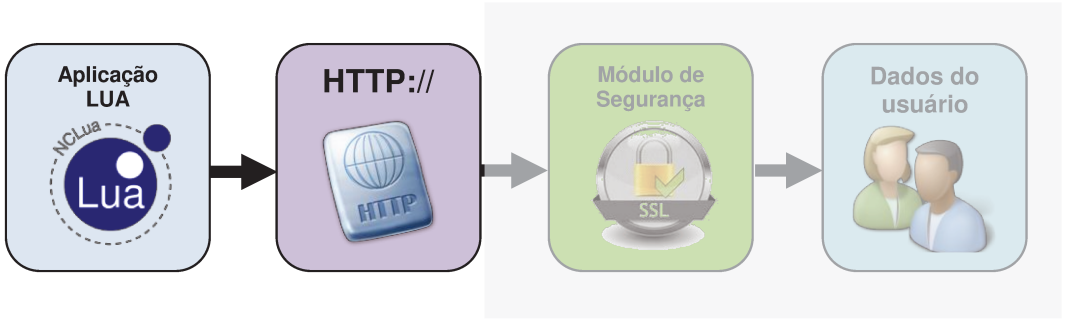

Figura 5.2 – Aplicação NCLua-HTTP.

A aplicação utilizada compreende o protocolo HTTP e entende o formato das mensagens, com cabeçalhos que estão inclusos na mensagem de requisição e também entende o formato da mensagem de resposta.

As implementações realizadas simplificam diversas chamadas de funções e co-rotinas, tais como, *tpc.send*[1], *tcp.receive*[2], necessárias à realização de uma conexão TCP no Ginga-NCL, encapsulando diversos detalhes do processo de comunicação. Algumas funções do módulo TCP são mostradas na Tabela 5.1. [26]

[1] Envia uma requisição ao servidor.
[2] Suspende a co-rotina até que algum dado seja obtido.

Tabela 5.1 – Funções do Módulo TCP NCLua. [29]

Comando	**Função**
connect (host, port)	Conecta em um servidor por meio do protocolo TCP.
disconnect ()	Fecha a conexão TCP e retorna imediatamente
execute (f, ...)	Função que deve ser chamada para iniciar uma conexão TCP.
handler (evt)	Função que trata os eventos.
receive (pattern)	Recebe resposta de uma requisição enviada previamente ao servidor.
send (value)	Envia uma requisição TCP ao servidor no qual se está conectado, e retorna imediatamente.

O módulo NCLua HTTP facilita o envio de requisições, encapsulando todo o gerenciamento das requisições assíncronas do protocolo TCP no Ginga-NCL. O módulo também permite a realização de requisições que requerem autenticação básica. Para realizar tal autenticação, os dados de login e senha devem ser codificados.

Na Figura 5.3 apresenta-se o código utilizado em NCLua para o acesso a páginas da *internet* com o protocolo HTTP.

```
package.path = package.path .. ';lib/?.lua'

require "http"

function callback (header, body)
  if body then
     print("\n\n\n", body, "\n\n\n")
  end

  event.post {class="ncl", type="presentation", action="stop"}
end

http.request (https://www.google.com.br/, callback)
```

Figura 5.3 – Programação da Aplicação para Acesso Seguro.

Os códigos da biblioteca HTTP utilizada na aplicação, são descritas no Anexo A. A função *request* utilizada, tem a função de enviar uma requisição HTTP para um determinado servidor com a página que se deseja acessar. A função *callback* ao ser executada, retorna sua assinatura com

um parâmetro *header* e um *body*, (cabeçalho e corpo da resposta), ambos campos no formato *strings*.

Para o servidor utilizou-se a programação de segurança a fim de se obter um servidor seguro, conforme a programação do módulo de segurança descrito no item a seguir.

5.2 MÓDULO DE SEGURANÇA

A base para se utilizar esse tipo de serviço é a facilidade e agilidade da implementação, sem maiores problemas na execução da aplicação ou comunicação sem esquecer o fator e característica primordial do HTTPS, a segurança. Na aplicação utilizou-se um servidor *web* capaz de fornecer por padrão criptografia baseada no protocolo SSL, o servidor *Apache.*

O HTTPS é utilizado serve para que se estabeleça um dialogo entre os navegadores e os servidores da seguinte forma: os navegadores enviam mensagens para os servidores e as mesmas são encriptografadas, de forma que somente os destinatários decifrem o seu conteúdo, com isso se agrega mais segurança entre a comunicação cliente-servidor. Na Figura 5.4 destaca-se o módulo de segurança implementado com o protocolo SSL.

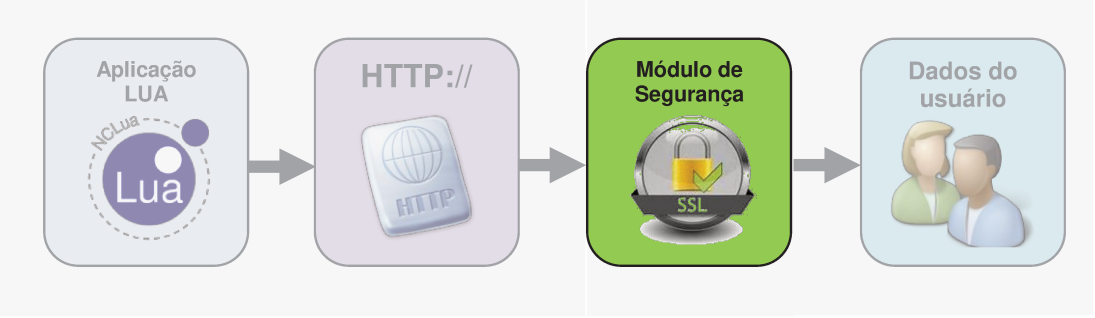

Figura 5.4 – Módulo de Segurança através do Protocolo SSL

Para a criação do módulo de segurança no *middleware* utilizou-se o protocolo de segurança SSL. As linhas de comando abaixo são necessárias para a inserção do modulo de segurança no servidor, utilizado na aplicação para televisão digital.

```
# apt-get install openssl
# apt-get install apache2 apache2.2-common
```

A primeira linha instala o SSL, que é uma tecnologia utilizada para codificar dados entre um usuário e um servidor.

A segunda linha instala o Apache que é um servidor web.

```
# a2enmod ssl
# openssl genrsa -des3 -out server.key 2048
```

A linha superior instala o módulo SSL e a segunda cria *Certificate Signing Request* – CSR, cujo tamanho será de 2048, é um arquivo de texto, gerado pelo servidor web, contendo as informações para a solicitação do seu certificado, usada para gerar um certificado assinado digitalmente.

Os dados requeridos para a instalação do CSR estão relacionados na Tabela 5.2.

Tabela 5.2 – Dados para Instalação do CSR.

Nome	**Função**	**Exemplo Utilizado**
Organization (O)	Deve ser preenchido com o Nome Empresarial (antiga Razão Social)	Unicamp
Organization Unit (OU)	Campo de preenchimento livre, normalmente contém o departamento que ficará responsável pelo certificado.	DECOM
Locality (L)	Deve ser preenchido com o nome por extenso da cidade onde a empresa está localizada.	Campinas
State (ST)	Deve ser preenchido com o nome por extenso do estado onde a empresa está localizada.	Sao Paulo
Country (C)	Deve ser preenchido com a sigla do país onde a empresa está localizada.	BR

```
# openssl req -new -key server.key -out server.csr
```

A função acima gera uma nova chave e uma nova senha.

```
# cd /etc/ssl/private
# openssl rsa -in server.key -out server.key.insecure
# mv server.key.insecure server.key
```

Os comandos acima são necessários para que o servidor inicie automaticamente sem a intervenção humana. Não será necessário, que em todos os acessos o usuário informe a chave e a contra chave.

```
# cp server.crt /etc/ssl/certs
# cp server.key /etc/ssl/private
```

As linhas de comando acima instalam os certificados criados.

```
# vim /etc/apache2/sites-available/default
```

A customização SSL é feita no servidor *Apache* através do comando acima descrito.

5.3 ACESSO SEGURO

A aplicação foi modificada para que ao acessar o servidor *web* utilize a versão com o certificado digital de segurança SSL, instalado conforme o item anterior. Dessa forma, os dados transmitidos e recebidos pelo usuário serão criptografados e protegidos quanto ao uso indevido das informações. A Figura 5.5 mostra como o cliente necessita de uma chave para autenticação de mensagens, uma chave para criptografia para de blocos.

Figura 5.5 – Autenticação Cliente-Servidor.

O servidor precisa dos mesmos elementos. O SSL requer que as chaves para que sentido sejam diferentes para outro sentido. Se ocorrer um ataque em um sentido, o outro não será afetado.

A Figura 5.6 mostra os parâmetros do certificado digital de segurança instalado para tornar o servidor seguro. O algoritmo implementado utiliza a chave pública RSA com 2048 bits (criptografia assimétrica) com SHA-1 (*hash*) para assinatura digital.

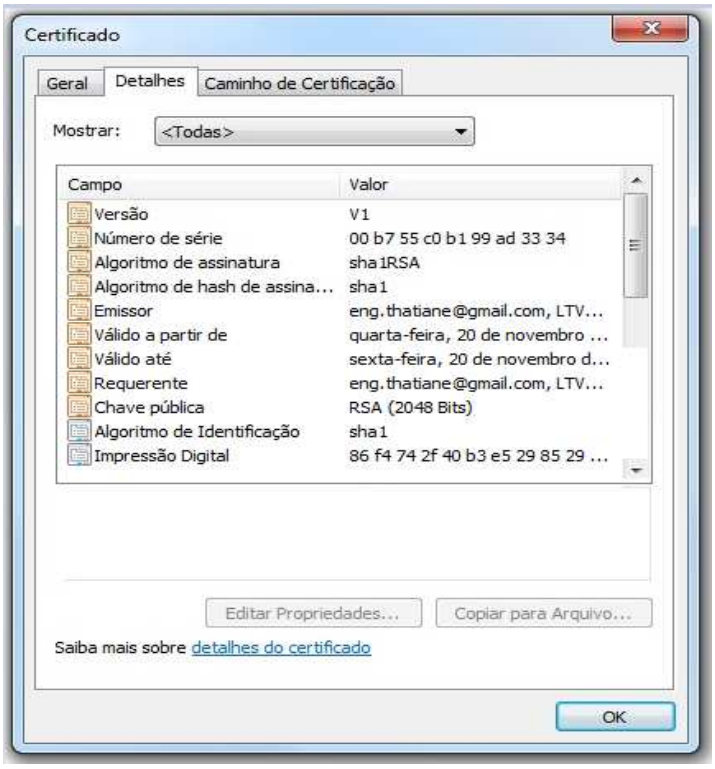

Figura 5.6 – Certificado do *Server Web*.

Na criptografia de chave pública, chaves diferentes são usadas para criptografar e descriptografar informações. Uma das principais vantagens dos certificados é que os *hosts* não têm mais que manter um conjunto de senhas para entidades individuais que precisam ser autenticadas para obter acesso. Em vez disso, o *host* simplesmente deposita confiança em um emissor de certificados.

Ao testar a aplicação no emulador do *middleware* Ginga, a resposta obtida é apresentada na Figura 5.7.

Figura 5.7 – Resposta da Aplicação com *Web Server* Seguro.

Os testes foram realizados utilizando a maquina virtual que emula o *middleware* Ginga, conforme a Figura 5.8.

Figura 5.8 – Emulador *Middleware* Ginga.

As informações obtidas a partir daí são as linhas de programação do site, conforme mostra o Anexo B, que devem ser transmitidas através de uma aplicação que traduza as linhas de programa-

ção segura de forma que o usuário veja as informações de forma usual como se estivesse navegando em um site.

A aplicação abre a página segura da *internet* (no exemplo, utilizou-se a página da Google), fazendo que a navegação de informações feitas pelo usuário esteja assegurada. A troca de informações entre o usuário, o servidor *e* aplicação é assegurada pela troca de chaves publicas e todas as senhas e chaves são criptografas e descriptografadas pelos algoritmos e chaves de segurança.

5.4 RESULTADOS

Através da metodologia desenvolvida nos testes entende-se que o cliente (usuário) faz o pedido de autenticação ao Provedor e Controlador (emissora). A partir das trocas de informações pelo cliente, provedor e controlador a autenticação do cliente é feita, liberando o acesso ao Servidor, que se mantém ativo em um Ambiente Seguro, possibilitando a troca de informações com segurança. O *Backend* após o estabelecimento da comunicação segura obtém as tarefas desempenhadas pela aplicação e envia os resultados a mesma. A Figura 6.1 apresenta a Sequencia Básica de Operação para Aplicação Segura, desenvolvida pela metodologia estudada.

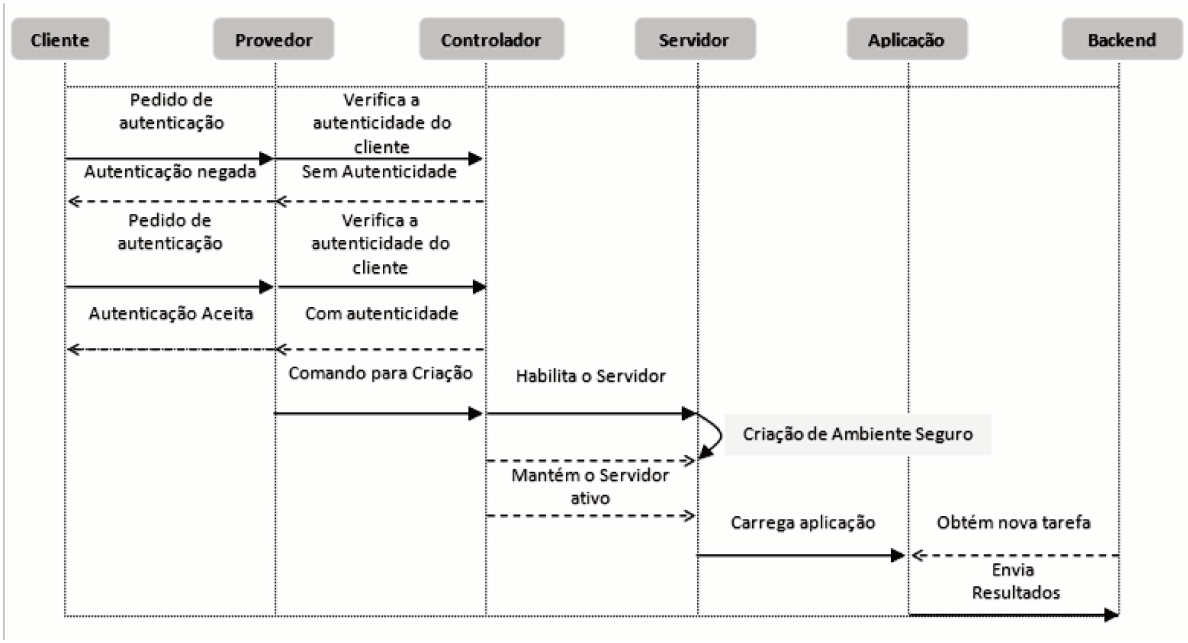

Figura 5.9 – Sequencia Básica de uma Aplicação Segura.

As vulnerabilidades estudadas para os sistemas móveis, como os *smartphones,* são vulnerabilidades presentes que podem ser mapeadas como primícias básicas para validar e aplicar técnicas já validadas em outros contextos. Dessa forma, foi possível relacionar como as vulnerabilidades afetam os sistemas de TV Digital e pode-se atender as normas de segurança a serem estabelecidas.

A nova norma ABNT NBR 15605 (Parte 2) pretende estabelecer mecanismos de autenticação de aplicativos Ginga e requisitos de segurança no uso do canal de interatividade para aplicações.

A autenticação das aplicações é baseada em assinaturas digitais conforme regras definidas pelo ICP-Brasil, tendo assim um suporte legal para as certificações digitais. A aplicação autenticada é considerada segura e pode abrir conexões HTTP. As aplicações NCLua utilizam a Infraestrutura integrada ao *Common Core* do *Middleware* Ginga. O protocolo SSL é o Mecanismo de autenticação de aplicação utilizado para prover as conexões seguras.

A aplicação se torna segura baseado no uso de uma infraestrutura de chaves públicas e de algoritmos criptográficos diversos. Permite a autenticação de aplicações e controle de acesso, impedindo o acesso não autorizado a recursos sensíveis. A aplicação estabelece um canal de comunicação seguro através de conexões seguras, utilizando o modulo de segurança SSL e o acesso a HTTPS, com autenticação do servidor e do cliente.

O objetivo de utilizar as certificações digitais estabelecidas pela ICP-Brasil é manter a padronização, e usar a hierarquia estabelecida por eles e manter a validade jurídica dos mesmos.

Foram utilizadas nos testes as recomendações da ICP-Brasil que é geração de *hash* (SHA-1) e criptografia assimétrica (RSA 2048 bits) e são compatíveis com aplicações *web*.

Figura 5.10 – Fluxo de Informações Seguras.

As aplicações seguras estabelecem um novo fluxo da informação, conforme mostra a Figura 6.2 permite a sincronização de dados com o servidor, possibilita acesso rápido às informações e mantém os dados sincronizados com o *backend*.

6 CONCLUSÕES

A preocupação com a segurança vem crescendo nos últimos anos e os sistemas operacionais dos sistemas estão sendo idealizadas com a segurança introduzida no processo de desenvolvimento, boas práticas de desenvolvimento seguro são seguidas utilizando experiências realizadas em outros sistemas. O objetivo dessas práticas de segurança é proporcionar controles de segurança para dificultar ou impossibilitar explorações ao sistema. A quantidade de aplicativos é muito maior do que a de Sistemas Operacionais, e a quantidade de desenvolvedores de SO é muito inferior a de desenvolvedores de aplicativos, mas os desenvolvedores de SO são mais experientes e os sistemas são menos vulneráveis do que os aplicativos.

Um estudo da situação atual da gestão da segurança da informação, da organização, bem como verificar dos pontos que estão de acordo com a normatização e daqueles que necessitam aprimoramentos no SBTVD, foi realizado e foram levantadas as vulnerabilidades nos aplicativos que possibilitam a execução do código remotamente e/ou vazamento de informações contidas nele para outros aplicativos.

A proposta promove a proteção da informação, via mecanismos de controle, contra possíveis ameaças – seja por ação intencional, mau uso do aplicativo, defeitos ou falhas na programação – que ocorram onde a informação estiver sendo criada, processada, armazenada ou transmitida. As informações manipuladas pelos aplicativos são propriedades dos usuários e a possível exploração pode causa danos ao usuário, aumentando os riscos e ameaças associadas a estes aplicativos.

A autenticação das aplicações é baseada em assinaturas digitais conforme regras definidas pelo ICP-Brasil, tendo assim um suporte legal para as certificações digitais. A aplicação autenticada é considerada segura e pode abrir conexões HTTP. O protocolo SSL é o mecanismo de autenticação de aplicação utilizado para prover as conexões seguras.

Uma metodologia baseada em acesso *web service* em aplicações de televisão digital interativas, fundamentada em resultados alcançados para sistema de telefonia celular, por serem sistemas análogos, em comparação às técnicas de ataque, as técnicas de segurança e proteção foi utilizada no trabalho.

Uma aplicação não assinada ou não autenticada não poderá solicitar a modificação das restrições impostas pelo receptor para ampliar seu acesso ao sistema. Uma aplicação assinada e autenticada pode solicitar permissões, desde que esteja certificado por um arquivo de requisição de permissões (certificado de atributos), emitido e assinado por entidade reconhecida. A autenticação das aplicações é baseada em assinaturas digitais conforme regras definidas pelo ICP-Brasil, tendo assim um suporte legal para as certificações digitais.

A segurança da informação para os sistemas de TV Digital, nas aplicações em Ginga-NCL e Lua deve ter parâmetros que assegurem que as rotinas não permitam a invasão indesejada e que protejam os dados dos usuários nas transações realizadas nas aplicações.

O Sistema de Televisão Digital deve ser tratado como os sistemas embarcados e para o desenvolvimento de aplicações seguras é preciso levar em consideração algumas ações, como: planejar a segurança; avaliar a vulnerabilidade de segurança do aplicativo; modelar a ameaça de segurança; avaliar o impacto de segurança; avaliar o risco de segurança; especificar as necessidades de segurança; fornecer informação de segurança; verificar e validar a segurança; monitorar o comportamento de segurança; gerenciar a segurança; garantir a segurança.

As aplicações interativas que manipulam dados confidenciais dos usuários devem ter rotinas de autenticação do usuário e sem essa autenticação não pode ser permitida o acesso e nem a modificação de nenhum dado que o receptor contenha. Desta forma, a confiabilidade, a integridade e disponibilidades estarão asseguradas. A autenticidade, a responsabilidade e privacidade são asseguradas na aplicação segura quando constantemente verificada, validada e monitorada as informações do usuário.

O protocolo SSL é um dos protocolos mais convenientes e utilizados em transações seguras. Sua implementação é relativamente simples, colocando-se o SSL no topo da pilha TCP/IP e substituindo as chamadas TCP pelas chamadas SSL. Trabalha independente das aplicações utilizadas e, após o *handshake* inicial, comporta-se como um canal seguro que permite que se execute todas as funções que normalmente estão disponíveis no TCP/IP. A maioria dos servidores e clientes (*browsers*) já têm suporte nativo para ele, fazendo do SSL um padrão de fato. O protocolo disponibiliza todas as primitivas necessárias para conexões seguras, sendo a autenticação, troca de chaves de sessão com o uso de criptografia assimétrica prévia, encriptação com métodos simétricos, MAC e certificação.

É importante observar que NCL é uma linguagem para integração e sincronização de mídias. Dessa forma, as aplicações declarativas desenvolvidas para os principais sistemas de TV digital citados não precisam, necessariamente, ser adaptadas para NCL, uma vez que a linguagem HTML/XHTML é entendida apenas como mais um tipo de objeto de mídia em NCL.

A aplicação se torna segura baseado no uso de uma infraestrutura de chaves públicas e de algoritmos criptográficos diversos. Permite a autenticação de aplicações e controle de acesso, impedindo o acesso não autorizado a recursos sensíveis. A aplicação estabelece um canal de comunicação seguro através de conexões seguras, utilizando o modulo de segurança SSL e o acesso a HTTPS, com autenticação do servidor e do cliente.

6.1 TRABALHOS FUTUROS

Como trabalhos futuros destaca-se a implementação em ambiente real, no *set-top-box*, das aplicações seguras, utilizando os mecanismos tratados no trabalho. A rotina de programação segura deve avaliar outras aplicações. A filosofia das lojas de aplicações para televisão digital deve ser mantida e implementada, mantendo as solicitações de permissões aos usuários, como forma de proteção e decisão sobre a instalação de aplicativos interativos. Os padrões de segurança, chaves publicas e privadas deve estar de acordo com os órgãos que regulamentam as certificações digitais.

BIBLIOGRAFIAS

[1]PICCOLO, Lara Schibelsky Godoy; BARANAUKAS, Maria Cecília, Desafios de Design para a TV Digital Interativa, Universidade Estadual de Campinas, Novembro de 2006.

[2]SCHIEFLER, G. H. C.. TV Digital: A nova ferramenta governamental para a inclusão social. Google Knol, 29 jul. 2008.

[3]GIOIA, Francisco. Multiplexação de Sinais, Serviços de Informação (SI) e Transmissão de Dados no Padrão Brasileiro de TV Digital. Escola de Engenharia – Universidade Federal Fluminense (UFF). 2008

[4]BECKER, V.; ZUFFO, M. K.. Desenvolvimento de Interfaces para TV Digital Interativa. In: XIV Simpósio Brasileiro de Sistemas Multimídia e Web, 2008, Vila Velha - ES. Anais: Minicursos - XIV Simpósio Brasileiro de Sistemas Multimídia e Web. SBC, 2008, 2008. p. 49-97.

[5]MANHÃES, Marcus Aurélio Ribeiro; SHIEH, Pei Jen, Canal de Interatividade: Conceitos, Potencialidades e Compromissos, 23 de agosto de 2005.

[6] OLIVEIRA, C. T., SOUZA, C. T., (2005) *A Return Channel Specification for Applications in Interactive Digital TV*. Original title: Especificação de Canal de Retorno em Aplicações para TV Digital Interativa. In: 22th Brazilian Symposium of Telecommunications (SBrT). Campinas, Brazil.

[7]PATACA, Daniel Moutinho. Tecnologias de Interação Inovadoras: Interatividade na TV Digital, CPQD, 24 de abril de 2008.

[8] MONTEZ, Carlos e PICCIONI, Carlos. Um Estudo sobre Emuladores de Aplicações para a Televisão Digital Interativa. Universidade Federal de Santa Catarina, Florianópolis. 2004.

[9]GHISI, B. C. ; LOPES, Guilherme Figueiredo; Frank Siqueira . Integração de Aplicações para TV Digital Interativa com Redes Sociais. In: Webmedia '10 (Workshop de TV Digital Interativa), 2010.

[10] SOARES, Luiz Fernando Gomes; RODRIGUES, Rogério Ferreira; MORENO, Márcio Ferreira. Ginga-NCL: *the Declarative Environment of the Brazilian Digital TV System. In: Journal of the Brazilian Computer Society.* No. 4, Vol. 13. p.37-46. ISSN: 0104-6500

[11] PAULINELLI, Fernanda; OMAIA, D.; BATISTA, Carlos Eduardo Coelho Freire; SOUZA FILHO, G. L. . Xtation: um Ambiente de Testes de Aplicações para TV Digital

Interativa Baseado no *Middleware* de Referência do Sistema Brasileiro de Televisão Digital. In: WebMedia 2006, 2006, Natal. Anais do *WebMedia* 2006 - DEMOS AND TOOLS, 2006.

[12] SANTOS, J. A. F. ; CARVALHO, R. M. ; DAMASCENO, J. R. ; SAADE, Debora Christina Muchaluat . Desenvolvendo Jogos Interativos para TV com o *Middleware* Brasileiro Ginga-NCL. In: XI Semana de Engenharia da UFF, 2009, Niterói. SEMENGE 2009, 2009. p. 1-6.

[13] NUNES, Francisco José Barreto; BELCHIOR, Arnaldo Dias; ALBUQUERQUE Adriano Bessa. *Security Engineering Approach to Support Software Security. In: 6th World Congress on Services, 2010, Miami. 6th World Congress on Services*, 2010.

[14] SOUZA FILHO, Guido Lemos de; LEITE, Luiz Eduardo Cunha; BATISTA, Carlos Eduardo Coelho Freire. Ginga-J: *The Procedural Middleware for the Brazilian Digital TV System. In: Journal of the Brazilian Computer Society*. No. 4, Vol.13. p.47- 56. ISSN: 0104-6500.

[15] SOUZA, G. L. F., L. E. C. Leite, Batista, C. E. C. F. (2007): Ginga-J: *The Procedural Middleware for the Brazilian Digital TV System. Journal of the Brazilian Computer Society*, Revista no. 4; Vol. 12; Mar. 2007 - ISSN 0104-6500.

[16] JUCÁ, Paulyne Matthews; LUCENA, Ubirajara; FERRAZ, Carlos. Desenvolvendo Aplicações da Televisão Digital. In: Congresso de Tecnologia de Rádio, Televisão e Telecomunicações, 2005, São Paulo, 2005.

[17] BRAGA, A. M. ; Restani, G.S. . *Hacking* Ginga: uma avaliação de segurança da plataforma de aplicações interativas da TV digital brasileira. In: X Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais, Fortaleza, 2010.

[18] OWASP Top 10 (2010). *The Ten Most Critical Web Application Security Risks*. 2010. www.owasp.org/index.php/Category: OWASP_Top_Ten_Project. Acesso em 01/03/2013.

[19] OBERHEIDE, J. and Jahanian, F. (2010) *When mobile is harder than fixed (and vice versa): demystifying security challenges in mobile environments. In Proceedings of the Eleventh Workshop on Mobile Computing Systems & Applications* (Annapolis, Maryland, February 22 – 23, 2010). HotMobile ‘10. ACM, New York, NY, 43-48. 2010.

[20] SOARES, L. F. G., Rodrigues, R. F., Moreno, M. F. (2007): *Ginga-NCL: the Declarative Environment of the Brazilian Digital TV System. Journal of the Brazilian Computer Society*, Revista no. 4; Vol. 12; Mar. 2007 - ISSN 0104-6500.

[21] DUNHAM, K. Abu-Nimeh, S., Becher M., Seth, F. , Hernaclti, H., J., Wright, C., Mobile Malware Attacks and Defense, Editora Elsevier Inc., 2009, ISBN 13: 978-1-59749-298-0.

[22] DAGON, C., Martin, T., & Starner, T. (2004). *Mobile phones as computing devices the viruses are coming. Pervasive Computing*, 3, 11–15.

[23] FOROUZAN, Behrouz A. Comunicação de dados e redes de computadores. 4ª Edição. Editora Mc Graw Hill, São Paulo. 2008.

[24] BECKER, V.; PICCIONI, Carlos Alexandre; MONTEZ, Carlos ; HERWEG FILHO, Gunter H . Datacasting e Desenvolvimento de Serviços e Aplicações para TV Digital Interativa. In: Cesar Augusto Camilo Teixeira; Eduardo Barrére; Iran Calixto Abrão. (Org.). Web e Multimídia: Desafios e Soluções. Poços de Caldas: PUC Minas, 2005, v. 01, p. 01-30.

[25] KOONG, S. Kai., Lai C. Liu, Shuming Bai and Bishan Lin "*Identity Theft in the USA: evidence from* 2002 to 2006," *International Journal of Mobile Communications*, Volume 6, Number 2, 199-216, 2008.

[26] YAP, T. S., & Ewe, H. T. (2005). *A mobile phone malicious software detection model with behavior checker. Lecture Notes in Computer Science*, 3597, 57–65.

[27] MAKLUF, C. A.; SANTOS, Thatiane Cristina dos; REUIZ, J. L.; Yuzo Iano; ARTHUR, R. . *A Study on the Technical Viability for Structuration of the Brazilian Digital Television s Return Channel using 3G Technologies. Proceedings - International Workshop on Telecommunicatios*, 2011.

[28] SANTOS, Thatiane Cristina dos; IANO, Y. ; Omar Branquinho. Análise de Desempenho da Transmissão de Video em Redes IEEE 802.11 visando a estruturação do canal de retorno para TV Digital. Revista Ciência e Tecnologia, v. 18, p. 29-38, 2008.

[29] MAKLUF, C. A.; SANTOS, Thatiane Cristina dos; REUIZ, J. L.; IANO, Y. ; ARTHUR, R.. Estudo de Tecnologias 3G visando à estruturação do Canal de Retorno da TV Digital. Displays Latin, 2010.

[30] SILVA FILHO, Manoel Campos da ; GONDIM . NCLua SOAP: Acesso a *Web Services* em aplicações de TVDi. In: III *Workshop* de Computação Aplicada em Governo Eletrônico, 2011, Florianópolis. 2011.

[31] KULESZA, R. ; LIMA, J. F. A. ; MIRANDA FILHO, S. ; Livio, Alan ; BRANDAO, R. R. M. ; ARAUJO, J. P. C. ; SOUZA FILHO, G. L. . Ginga-J: Implementação de Referência do Ambiente Imperativo do *Middleware* Ginga. In: *Webmedia* 2010 - XVI Simpósio Brasileiro de Sistemas Multimídia e Web, 2010, Belo Horizonte. Anais do XVI Simpósio Brasileiro de Sistemas Multimídias e Web (Webmedia 2010), 2010. p. 35-42.

[32] FEIJO, B.; BADARO, P. . Conceitos e Modelos para um Sistema Brasileiro de Produção de Conteúdo Digital. Rio de Janeiro: Departamento de Informática, 2006.

[33] PEROZZO, Reiner F.; PEREIRA, C. E. . *Framework* para Integração entre Ambientes Inteligentes e o Sistema Brasileiro de TV Digital Terrestre. In: Congresso Brasileiro de Engenharia de Televisão, 2010, São Paulo. Anais do Congresso da Sociedade Brasileira De Engenharia de Televisão. São Paulo: Revista da SET, 2010. v. 4.

[34] SQUIRRA; MOREIRA, Fernando José Garcia . A TELEVISÃO DIGITAL INTERATIVA COMO VEÍCULO QUE SALVA VIDAS. ANIMUS - Revista Interamericana de Comunicação Midiática, 2011.

[35] SALES, M. B. ; Schwaab, A. A. S. ; NASSAR, S. M. . *Application of Bayesian Networks to Assist the Expansion of the Digital Inclusion Of Elderly People*. Revista IEEE América Latina, v. 8, p. 275-279, 2010.

[36] BRAGA, A. M. ; Restani, G.S. . *Hacking* Ginga: uma avaliação de segurança da plataforma de aplicações interativas da TV digital brasileira. In: Décimo Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais, 2010, Fortaleza. Minicursos do SBSeg, 2010.

[37] SOARES, L. F. G. ; RODRIGUES, Rogério Ferreira ; MORENO, Márcio Ferreira . Ginga-NCL: the Declarative Environment of the Brazilian Digital TV System. Journal of the Brazilian Computer Society, v. 12, p. 37-46, 2007.

[38] MORENO, MARCIO F. . Ginga-NCL: *Relating Imperative, Declarative and Media Objects. In: Doctoral Consortium of European Conference on Interactive Television,* 2009, Leuven. Proceedings of of European Conference on Interactive Television, 2009.

[39] ZOLEZI, R. H. M. ; CAVENAGHI, M. A. ; SPOLON, R. ; LOBATO, R. S. ; ALBINO, J. P. ; AZEVEDO, M. L. . Identificação de Usuário em Aplicativos Interativos para a TVDi. Temática (João Pessoa. Online), v. 08, p. 1-14, 2012.

[40] GIRALDEZ, D. C.; Schweitzer, C. M. . Qualidade de Experiência (QoE) para serviços IPTV - Investigação, Análise e Proposta de Soluções. In: XXV Congresso de Iniciação Científica da UNESP, 2013, Ilha Solteira. Anais do XXV Congresso de Iniciação Científica da UNESP, 2013.

[41] FERNANDES, Jorge Henrique Cabral; SOUZA FILHO, G. L. ; SILVEIRA, Gledson Elias da . Introdução à Televisão Digital Interativa: Arquitetura, Protocolos, Padrões e Práti-

cas. In: A. M. S. Andrade; A. T. Martins; R. J. A. Macedo. (Org.). XXIII JAI - Livro Texto, Anais do XXIV Congresso da Sociedade Brasileira de Computação. 1ed.Salvador, BA: Sociedade Brasileira de Computação, 2004, v. 2, p. -.

[42] MEMÓRIA, Felipe F. P.; MONT'ALVÃO, C. R. . Pesquisas em Usabilidade no Brasil: Academia x Mercado. In: 3 USIHC - Congresso Internacional de Ergonomia e Usabilidade, Design de Interfaces e Interação Humano-Computador, 2004, Rio de Janeiro. Anais do 3 USIHC. Rio de Janeiro: LEUI/PUC-Rio, 2004.

[43] SANT ANNA, F. F. G. ; CERQUEIRA, R. F. G. ; SOARES, L. F. G. . NCLua - Objetos Imperativos Lua na Linguagem Declarativa NCL. In: XIV Simpósio Brasileiro de Sistemas Multimídia e Hipermídia, 2008, Vila Velha. Anais do XIV Simpósio Brasileiro de Sistemas Multimídia e Hipermídia, 2008. p. 83-90.

[44] EBRAHIMI, Touradj. *Quality of Multimedia Experience: Past, Present and Future. Multimedia Signal Processing Group*. Ecole Polytechnique Fédérale de Lausanne (EPFL), 2009.

[45] RODRIGUES, R. F. ; SOARES, Luiz Fernando Gomes . Produção de Conteúdo Declarativo para TV Digital. In: Seminário Integrado de *Software* e *Hardware*, 2006, Campo Grande. XXXIII SEMISH - Seminário Integrado de *Software* e *Hardware* (aceito para publicação). Porto Alegre: Sociedade Brasileira de Computação, 2006. p. 286-300.

[46] CABRAL, Eula D.T. ; CABRAL FILHO, Adilson Vaz . Tv Digital Terrestre no Brasil: inovação tecnológica para uma expansão sem novidades. In: Valério Cruz Brittos; Ruy Sardinha Lopes. (Org.). Políticas de comunicação e sociedade. 1ed. São Paulo: Intercom, 2012, v. 1, p. 149-172.

[47] BRAGA, A. M. ; RESTANI, G.S. . Capítulo 5: Introdução à segurança de aplicações para TV Digital Interativa Brasileira. In: SBC. (Org.). Caderno de Minicursos - X Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais. : , 2010, v. , p. -.

[48] REIS, Bruno; MOTA, Jimmy Costa, OLIVEIRA, Patryck Pabllo Borges de. Classificação da Informação. Universidade Católica de Brasília (UCB), Campos Universitário II, SGAN 916 – Módulo B – Brasília – DF – Brasil, 2005.

[49] SILVA, Reginaldo F. ; CUNHA, José A.. ARQUITETURA DE SEGURANÇA EM APLICAÇÕES BASEADAS EM WEB SERVICES. Holos, Ano 21, dezembro 2005.

[50] NUNES, Francisco José Barreto; BELCHIOR, Arnaldo Dias . Processo Seguro de Desenvolvimento de *Software*. In: IV Conferência Ibero Americana WWW/*INTERNET*, 2006, Murcia. Anais da IV Conferência Ibero Americana WWW/*Internet*, 2006.

[51] SOUSA, E. R. ; MELONI, L.G.P. ; Monteiro, C. C ; GONDIM, P. R. L . Plataforma Estruturada em *Software* para Desenvolvimento e Testes de Aplicações para TV Digital Interativa, Segundo o Padrão do SBTVD. Simpósio Internacional de Competências em Tecnologias Digitais Interativas na Educação, v. 1, p. 63-75, 2009.

[52] COSTA, Laisa. ZUFFO, M. K. Segurança para o Sistema Brasileiro de Televisão Digital: Contribuições à Proteção de Direitos Autorais e à Autenticação de Aplicativos, Mestrado em Engenharia Elétrica, USP. Ano de Obtenção: 2009.

[53] SOARES NETO, C. S. ; BARBOSA, S. D. J. ; SOARES, L. F. G. ; RODRIGUES, R. F. . Desenvolvimento de Aplicações Declarativas para TV Digital Interativa. In: César A.C. Teixeira; Cléver Ricardo G. de Farias; Jair C. Leite; Raquel O. Prates.. (Org.). Tópicos em Sistemas Interativos e Colaborativos. Porto Alegre: Sociedade Brasileira de Computação, 2006, v. 1, p. 7-45.

[54] BORDIGNON, A.; ROESLER, V. . Um *framework* para prover comunicação Segura em aplicativos de TV Digital. In: 4o ERRC - Escola Regional de Redes de Computadores, 2006, Passo Fundo. 4o ERRC - Escola Regional de Redes de Computadores, 2006. v. 1. p. 16-21

ANEXOS

ANEXO A – Biblioteca http.lua

A Biblioteca HTTP.lua utilizada para o desenvolvimento da aplicação. As linhas de comando que são descritas são funções da Biblioteca HTTP.lua utilizada para a implementação da aplicação testada na seção 5. [30]

```
require "tcp"
require "base64"
require "util"

local _G, tcp, print, util, base64, string, coroutine, table, type =
      _G, tcp, print, util, base64, string, coroutine, table, type

module "http"

version = "NCLuaHTTP/0.9.9"

local function getHeaderAndContent(response)
       local i = string.find(response, string.char(13,10,13,10))
       local header, body = "", ""
       if i then
          header = string.sub(response, 1, i)
          body = string.sub(response, i+4, #response)
       else
          header = response
       end
       return header, body
end



function request(url, callback, method, params, userAgent, headers, user, password,
port)
    headers = headers or ""
```

```
params = params or ""
if method == nil or method == "" then
   method = "GET"
end
userAgent = userAgent or version
   port = port or 80
method = string.upper(method)
if method ~= "GET" and method ~= "POST" then
   error("Parâmetro method deve ser GET ou POST")
end

local protocol, host, port1, path = splitUrl(url)

if port1 ~= "" then
   port = port1
end
if protocol == "" then
   protocol = "http://"
   url = protocol .. url
end

   tcp.execute(
            function ()
                tcp.connect(host, port)
                --conecta no servidor
                print("Conectado a "..host.." pela porta " .. port)
                         url = string.gsub(url, " ", "%%20")
                local request = {}
                        local fullUrl = ""
                        if port == 80 then
                           fullUrl = url
```

```
            else
               fullUrl = protocol .. host .. ":" ..port .. path
            end

         table.insert(request, method .." "..fullUrl.." HTTP/1.0")

         if userAgent and userAgent ~= "" then
            table.insert(request, "User-Agent: " .. userAgent)
         end

         if params ~= "" then
            if (method=="POST") and (type(params) == "table") then
               if headers ~= "" then
                  headers = headers .. "\n"
               end
               headers  =  headers.."Content-type:  application/x-www-form-
urlencoded"
            end
         end

         if headers ~= "" then
            table.insert(request, headers)
         end

         if user and password and user ~= "" and password ~= "" then
            table.insert(request, "Authorization: Basic " ..
            base64.enc(user..":"..password))
         end
      if params ~= "" then
         if type(params) == "table" then
            params = util.urlEncode(params)
```

```
                    end
                    --length of the URL-encoded params data
                    table.insert(request, "Content-Length: " .. #params.."\n")
                    table.insert(request, params)
                end
                    table.insert(request, "\n")
                       local requestStr = table.concat(request, "\n")
                   print("\n--------------------Request: \n\n"..requestStr)

                   local response = tcp.receive("*a") --parâmetro "*a" =

                   if response ~= nil then
                   print("\n\n----------------------------Resposta da requisição obti-
da\n\n")
                   print(response)
                     end--]]

                   tcp.disconnect()
                        --print("\n-------------------------Desconectou")
                     if response then
                        callback(getHeaderAndContent(response))
                     end
            end
        )
end


function getFile(url, callback, fileName, userAgent, user, password, port)
    local function fileDownloaded(header, body)
          if header then
             --print(response, "\n")
```

```
            print("Dados da conexao TCP recebidos")
            --Verifica se o código de retorno é OK
            if string.find(header, "200 OK") then
               if fileName then
                  util.createFile(body, fileName, true)
                  print('Arquivo criado com sucesso: '..fileName)
               end
            end
         else
            print("Erro ao receber dados da conexao TCP")
         end

      if callback then
         callback(header, body)
      end
   end

   local header, body = request(url, fileDownloaded, "GET", nil, userAgent, nil, user,
password, port)
end

function getHttpHeader(header, fieldName)
  local i = string.find(header, fieldName .. ":")
  if i then
     local fim = string.find(header, "\n", i) or string.find(header, " ", i)
     return string.sub(header, i, fim)
  else
     return nil
  end
end
```

```
function splitUrl(url)

  local protocolo = ""
  local separadorProtocolo = "://"
  local i = string.find(url, separadorProtocolo)
  if i then
     protocolo = string.sub(url, 1, i+2)
          i=i+#separadorProtocolo
  else
     i = 1
  end

  local host, porta, path = "", "", ""
  local j = string.find(url, "/", i)
  if j then
     host = string.sub(url, i, j-1)
     path  = string.sub(url, j, #url)
  else
     host = string.sub(url, i)
  end

  i = string.find(host, ":")
  if i then
    porta = string.sub(host, i+1, #host)
    host = string.sub(host, 1, i-1)
  end

  return protocolo, host, porta, path
end
```

ANEXO B – Resposta do Site Requerido

As informações obtidas a partir daí são as linhas de programação do site, conforme mostra o Anexo B, que devem ser transmitidas através de uma aplicação que traduza as linhas de programação segura de forma que o usuário veja as informações de forma usual como se estivesse navegando em um site.

A aplicação abre a página segura da internet (no exemplo, utilizou-se a página da Google), fazendo que a navegação de informações feitas pelo usuário esteja assegurada.

<!doctype html><html itemscope="" itemtype="http://schema.org/WebPage"><head><meta itemprop="image" content="/images/google_favicon_128.png"><title>Google</title><script>(function(){
window.google={kEI:"xZKTUvqaD8jbkQfM0IEQ",getEI:function(a){for(var b;a&&(!a.getAttribute||!(b=a.getAttribute("eid")));)a=a.parentNode;return b||google.kEI},
https:function(){return"https:"==window.location.protocol},kEXPI:"17259,146363,148115,4000116,4007278,4007661,4007714,4007830,4008067,4008133,4008142,4008725
,4009033,4009352,4009565,4009641,4010310,4010806,4010899,4011053,4011063,4011109,4011228,4011258,4011423,4011524,4011525,4011529,4011559,4011678,4011709,4011
847,4011863,4011911,4011971,4012001,4012137,4012141,4012145,4012147,4012190,4012271,4012275,4012302,4012316,4012318,4012342,4012344,4012361,4012365,4012414,4
012422,4012506,4012526,4012585,4012591,4012593,4012611,4012629,4012632,4012681,4012732,4012836,4012839,4012851,4012869,4012884,4012895,4012916,4013008,401306
8,4013140,4013146,4013178,4013312,4013374,4013378,4013416,4013513,8300004",kCSI:{e:"17259,146363,148115,4000116,4007278,4007661,4007714,4007830,4008067,40081
33,4008142,4008725,4009033,4009352,4009565,4009641,4010310,4010806,4010899,4011053,4011063,4011109,4011228,4011258,4011423,4011524,4011525,4011529,4011559,40
11678,4011709,4011847,4011863,4011911,4011971,4012001,4012137,4012141,4012145,4012147,4012190,4012271,4012275,4012302,4012316,4012318,4012342,4012344,4012361
,4012365,4012414,4012422,4012506,4012526,4012585,4012591,4012593,4012611,4012629,4012632,4012681,4012732,4012836,4012839,4012851,4012869,4012884,4012895,4012
916,4013008,4013068,4013140,4013146,4013178,4013312,4013374,4013378,4013416,4013513,8300004",ei:"xZKTUvqaD8jbkQfM0IEQ"},authuser:0,ml:function(){},kHL:"pt-BR
",time:function(){return(new Date).getTime()},log:function(a,b,c,l,k){var d=new Image,f=google.lc,e=google.li,g="",h="gen_204";k&&(h=
k);d.onerror=d.onload=d.onabort=function(){delete f[e];f[e]=d;c||-1!=b.search("&ei=")||(g="&ei="+google.getEI(l));c=c||"/"+h+"?atyp=i&ct="+a+"&cad="+b+g+"&z
x="+google.time();
a=/^http:/i;a.test(c)&&google.https()?(google.ml(Error("GLMM"),!1,{src:c}),delete f[e]):(d.src=c,google.li=e+1)},lc:[],li:0,Toolbelt:{},y:{},x:function(a,
b){google.y[a.id]=[a,b];return!1},load:function(a,b,c){google.x({id:a+m++},function(){google.load(a,b,c)})}};var m=0;})();
(function(){google.sn="webhp";google.timers={};google.startTick=function(a,b){google.timers[a]={t:{start:google.time()},bfr:!!b}};google.tick=function(a,b,g)
{google.timers[a]||google.startTick(a);google.timers[a].t[b]=g||google.time()};google.startTick("load",!0);
try{}catch(d){}})();
var _gjwl=location;function _gjuc(){var a=_gjwl.href.indexOf("#");if(0<=a&&(a=_gjwl.href.substring(a),0<a.indexOf("&q=")||0<=a.indexOf("#q="))&&(a=a.substrin
g(1),-1==a.indexOf("#")){for(var d=0;d<a.length;){var b=d;"&"==a.charAt(b)&&++b;var c=a.indexOf("&",b);-1==c&&(c=a.length);b=a.substring(b,c);if(0==b.indexO
f("fp="))a=a.substring(0,d)+a.substring(c,a.length),c=d;else if("cad=h"==b)return 0;d=c}_gjwl.href="/search?"+a+"&cad=h";return 1}return 0}
function _gjh(){!_gjuc()&&window.google&&google.x&&google.x({id:"GJH"},function(){google.nav&&google.nav.gjh&&google.nav.gjh()})};
window._gjh&&_gjh();</script><style>#gbar,#guser{font-size:13px;padding-top:1px !important;}#gbar{height:22px}#guser{padding-bottom:7px !important;text-align
:right}.gbh,.gbd{border-top:1px solid #c9d7f1;font-size:1px}.gbh{height:0;position:absolute;top:24px;width:100%}@media all{.gb1{height:22px;margin-right:.5em
;vertical-align:top}#gbar{float:left}}a.gb1,a.gb4{text-decoration:underline !important}a.gb1,a.gb4{color:#00c !important}.gbi .gb4{color:#dd8e27 !important}.
gbf .gb4{color:#900 !important}</style><style>body,td,a,p,.h{font-family:arial,sans-serif}body{margin:0;overflow-y:scroll}#gog{padding:3px 8px 0}td{line-heig
ht:.8em}.gac_m td{line-height:17px}form{margin-bottom:20px}.h{color:#36c}.q{color:#00c}.ts td{padding:0}.ts{border-collapse:collapse}em{font-weight:bold;font
-style:normal}.lst{height:25px;width:496px}.gsfi,.lst{font:18px arial,sans-serif}.gsfs{font:17px arial,sans-serif}.ds{display:inline-box;display:inline-block
;margin:3px 0 4px;margin-left:4px}input{font-family:inherit}a.gb1,a.gb2,a.gb3,a.gb4{color:#11c !important}body{background:#fff;color:black}a{color:#11c;text-
decoration:none}a:hover,a:active{text-decoration:underline}.fl a{color:#36c}a:visited{color:#551a8b}a.gb1,a.gb4{text-decoration:underline}a.gb3:hover{text-de
coration:none}#ghead a.gb2:hover{color:#fff !important}.sblc{padding-top:5px}.sblc a{display:block;margin:2px 0;margin-left:13px;font-size:11px}.lsbb{backgro
und:#eee;border:solid 1px;border-color:#ccc #999 #999 #ccc;height:30px}.lsbb{display:block}.ftl,#fll a{display:inline-block;margin:0 12px}.lsb{background:url
(/images/srpr/nav_logo80.png) 0 -258px repeat-x;border:none;color:#000;cursor:pointer;height:30px;margin:0;outline:0;font:15px arial,sans-serif;vertical-alig
n:top}.lsb:active{background:#ccc}.lst:focus{outline:none}#addlang a{padding:0 3px}.tiah{width:458px}</style><script></script> </head><body bgcolor="#fff"><s
cript>(function(){var src='/images/srpr/nav_logo80.png';var iesg=false;document.body.onload = function(){window.n && window.n();if (document.images){new Imag
e().src=src;}
if (!iesg){document.f&&document.f.q.focus();document.gbqf&&document.gbqf.q.focus();}
}
})();</script><textarea id="csi" style="display:none"></textarea> <nobr>Pesquisa <a class=gb1 href="http://www.go
ogle.com.br/imghp?hl=pt-BR&tab=wi">Imagens Maps <a class=gb1 href="https://play.go
ogle.com/?hl=pt-BR&tab=w8">Play YouTube <a class=gb1 href="http://news.google.com.br/nwshp?h
l=pt-BR&tab=wn">Not&#65533;cias Gmail Drive <u>Mais</u> »</nobr>
<nobr>Hi
st&#65533;rico da Web | Configura&#65533;&#65533;es | <a target=_top id=gb_70 href="https://accounts.google.com/ServiceLogin?hl
=pt-BR&continue=http://www.google.com.br/" class=gb4>Fazer login</nobr> <
center><br clear="all" id="lgpd"> <div title="Google" align="left" id="hplogo" onload="window.lol&&lol()" style=
"height:110px;width:276px;background:url(/images/srpr/logo9w.png) no-repeat"><div nowrap="" style="color:#777;font-size:16px;font-weight:bold;position:relati
ve;top:70px;left:218px">Brasil
 <form action="/search" name="f"><table cellpadding="0" cellspacing="0"><tr valign="top"><td width="
25%"> </td><td align="center" nowrap=""><input name="ie" value="ISO-8859-1" type="hidden"><input value="pt" name="hl" type="hidden"><input name="source"
type="hidden" value="hp"> <input autocomplete="off" class="lst tiah" v

A.1 – Linhas de Programação do Site Requerido