SecBox: Uma abordagem para segurança de *set-top boxes* em TV Digital

Diego Fiori de Carvalho[1], Mateus G. Milanez, Mário J. B. Avelino[2], Sarita M. Bruschi e Rudinei Goularte

Institute of Mathematics and Computing Sciences

University of São Paulo

São Carlos, SP, Brazil

{dfiori, mgm, mario, sarita, rudinei}@icmc.usp.br

Abstract. *In this work we present an alternative approach, named SecBox, to the security problem regarding to files and media content inside set-top boxes. Basically, the system uses a combination of cryptography and steganography algorithms, not properly explored in the area. As a result, SecBox helps to prevent storage devices hacking and digital piracy in Linux based set-top boxes.*

Resumo. *Neste trabalho é apresentada uma abordagem alternativa, denominada SecBox, para proteção de arquivos de sistema e multimídia no interior de set-top boxes. Basicamente, o sistema usa uma combinação de algoritmos de criptografia e esteganografia, não explorados na área. Como resultado direto, a SecBox pode auxiliar na prevenção de ataques aos dispositivos de armazenamento e pirataria digital. O sistema tem funcionamento em set-top boxes com Linux que utilizam quaisquer sistemas de arquivos.*

1. Introdução

Recentes avanços tecnológicos no desenvolvimento de dispositivos portáteis como PDAs, celulares, aliado à grande disseminação da *Internet*, têm feito aumentar a preocupação com a segurança de dados em sistemas computacionais embarcados [1, 2]. Esse fato relaciona-se com constantes ataques de intrusos na busca de acesso não autorizado a dados confidenciais por meio de conexões de rede e serviços de arquivos remotos. Essa preocupação está presente no desenvolvimento do Sistema Brasileiro de TV Digital (SBTVD), o qual possui como um de seus elementos centrais um sistema embarcado chamado *set-top box*.

No caso dos *set-top boxes* adiciona-se a preocupação com pirataria digital. Espera-se que uma das maiores novidades advindas da TV digital interativa seja a

[1] Apoio da FAPESP, número de processo: 06/51602-7.

[2] Apoio da FAPESP, número de processo: 05/58282-5.

possibilidade de interatividade com o conteúdo[3] sendo exibido [3], permitindo que *set-top boxes* armazenem e manipulem conteúdo multimídia.

O livre acesso aos arquivos do sistema e ao conteúdo, além do potencial risco de danos ao aparelho, pode ferir patentes e direitos autorais. Exemplos de mau uso dos dados incluem tanto acesso e troca ilegal de conteúdo via rede[4], quanto à troca de dispositivos de armazenamento (discos rígidos) entre *set-top boxes*.

Uma estratégia usual para proteção de dados é a adoção de mecanismos de criptografia. Contudo, criptografia remete à ciência da existência de uma mensagem cifrada, a qual pode ser interceptada. Uma vez interceptada a mensagem pode ser corrompida ou até mesmo decifrada. Uma estratégia adicional para proteção de dados, ainda pouco explorada em sistemas embarcados, é a esteganografia [5]. Essa técnica busca ocultar informações secretas em outras fontes de informação, como imagens, áudio, vídeo, sem causar perturbações perceptíveis na reprodução das mesmas. Assim, não se tendo ciência da existência da informação oculta, a mesma passará despercebida. A utilização conjunta das técnicas de criptografia e esteganografia podem trazer benefícios a sistemas computacionais uma vez que tem potencial para aumentar o nível de segurança nesses sistemas.

A técnica é aplicada em sistemas embarcados com o sistema operacional Linux utilizando os métodos de criptografia em diversas etapas da inicialização do sistema. Uma dessas etapas corresponde na verificação de integridade de determinados arquivos do sistema, dos quais são gerados resumos do tipo *hash* [6]. Esses resumos correspondem a chaves que identificam unicamente os arquivos originais criados na instalação do sistema. A cada nova inicialização são criados novos resumos correspondentes aos arquivos atuais do sistema. Essas novas chaves são comparadas com as chaves geradas na instalação.

Em outra etapa realiza-se a verificação dos itens do *hardware* utilizado pelo *set-top box*. Se algum componente de *hardware* for alterado (substituído ou reconfigurado), a SecBox identificará esta mudança e interromperá a inicialização. Os resumos *hash* correspondentes às informações dos itens de *hardware* são armazenados em uma imagem por meio de esteganografia. Como a imagem usada para realizar o ocultamento dessas informações não possui diferenças visuais perceptíveis quando comparada à imagem original, a informação oculta passará despercebida caso esta imagem seja interceptada. As chaves do sistema conjuntamente com a imagem esteganografada são armazenados em um sistema de arquivos criptografado [7].

Assim, a SecBox adiciona uma barreira a mais para ataques, locais ou remotos, e garante que alterações indevidas, como por exemplo a retirada de dispositivo de armazenamento (discos rígidos) do sistema para cópias indevidas, não obtenham acesso ao conteúdo.

É de conhecimento comum que sistemas embarcados freqüentemente têm limitações severas de recursos. Desse modo, a utilização de técnicas de criptografia são custosas aos mesmos. Porém, a SecBox é aplicada apenas na inicialização do sistema,

---

[3] Algumas operadoras de TV por assinatura, como a Sky [4], já oferecem serviços de gravação digital do conteúdo sendo veiculado (filmes, novelas, telejornais, etc.).

[4] Prevê-se que alguns modelos de *set-top box* terão conexão com a *Internet*.

acarretando um acréscimo de apenas alguns segundos na inicialização. Além disso, os *set-top boxes* importados mais atuais apresentam poder de processamento de aproximadamente 800 MHz [8].

A literatura, para o melhor de nosso conhecimento, não reporta trabalhos que explorem conjuntamente criptografia e esteganografia de escrita secreta para segurança em sistemas embarcados. Neste trabalho, propomos a utilização de esteganografia, de um modo não usual, aliada à criptografia com o objetivo de alcançar proteção não só do sistema como também para prevenção contra cópias ilegais de mídias digitais armazenadas no *set-top box*.

Este artigo está organizado do seguinte modo: a Seção 2 apresenta uma análise de trabalhos relacionados; a Seção 3 resume algumas tecnologias importantes empregadas na SecBox; a Seção 4 apresenta as técnicas de segurança utilizadas na SecBox; a Seção 5 discute como as técnicas apresentadas na Seção 4 são utilizadas em conjunto na SecBox; a Seção 6 descreve a arquitetura do sistema, e por fim, na Seção 7 são apresentadas algumas conclusões sobre o trabalho desenvolvido e propostas de trabalhos futuros.

2. Trabalhos Relacionados

Em sistemas embarcados, trabalhos sobre segurança são fartos. Um trabalho interessante é o de Kocher *et al* [1], que apresenta uma classificação das formas de segurança para sistemas embarcados, servindo como diretriz para a implementação de sistemas mais robustos. Koopman [9], por sua vez, discute quais técnicas criptográficas devem ser utilizadas para evitar ataques a dispositivos embarcados.

Ravi e Raghunathan [10] descrevem as diversas técnicas de segurança para sistemas embarcados, discutindo prós e contras e apontando vantagens do uso conjunto de algumas dessas técnicas para resolução de problemas de segurança. Todos esses trabalhos [1,9 e 10] preocupam-se em demonstrar como são robustas as suas técnicas em relação a criptoanálise. Já a SecBox, além de preocupar-se com a criptoanálise, utilizando algoritmos do estado da arte de criptografia, adiciona também a esteganografia digital para acrescentar mais um nível de segurança.

Os problemas de segurança descritos por Ravi e Raghunathan [10] podem ser visualizados na Figura 1. Dos sete problemas apresentados, na Figura 1, a SecBox implementa soluções para quatro delas: armazenamento seguro, funções básicas de segurança, resistência a invasões e segurança de contexto. Para proteção contra violações no dispositivo de armazenamento (“armazenamento seguro”) são verificados *hashs* de diversos arquivos de sistema e itens únicos de dispositivos da máquina[5]. Caso o dispositivo de armazenamento seja retirado para ser instalado em outro *set-top box*, o mesmo não será inicializado. Se for retirado para ser instalado em outra máquina como um PC, os dados referentes ao conteúdo digital estarão protegidos com um sistema de arquivos criptografado.

Já para implementação das “funções básicas de segurança” utilizam-se técnicas de *hashs* para verificação de integridade, para sigilo utilizam-se cifras simétricas em sistemas de arquivos criptografados e esteganografia em imagens para ocultamento de

[5] Itens: número de série de processador, identificador físico, obtidos por meio de *lshw* (*Hardware Lister*) [12].

informações. Para “resistência a invasões” utiliza-se análise de integridade dos itens de *hardware* no procedimento de inicialização determinando o desligamento automático do sistema caso alguma irregularidade seja descoberta. E finalmente para “segurança do contexto” são verificados *timestamps*[6] de arquivos importantes do sistema e codificação binária de rotinas utilizadas na inicialização do sistema por meio do aplicativo Shc[7] [11].

Figura 1. Problemas de segurança em dispositivos embarcados [10].

3. Tecnologias Utilizadas

Para realizar a implementação de soluções para armazenamento seguro, funções básicas de segurança, resistência a invasões e segurança de contexto, a SecBox utiliza um sistema de arquivos criptografados, técnicas de criptografia visando integridade dos dados e técnica de esteganografia para armazenamento de informações pertinentes. Na Seção 3.1 é apresentado o sistema de arquivos criptografados, na Seção 3.2 descreve-se a cifra de integridade, e finalmente na Seção 3.3 é discutido sobre esteganografia digital.

3.1 Sistemas de Arquivos Criptografados

Um recurso importante para proteção de dados é a utilização de sistemas de arquivos criptografados. Esses sistemas usam criptografia de cifragem simétrica de blocos para prover o sigilo sobre os arquivos armazenados em diretórios e/ou partições do disco.

Um sistema de arquivos criptografado pode ser implementado de três formas:

1. Criptografia de todo o disco rígido.
2. Criptografia de uma partição do disco rígido.

---

[6] No contexto deste trabalho, *Timestamp* é a informação de data e hora (até centésimos de segundo) adquirida no momento da criação de um arquivo de sistema.

[7] Aplicativo utilizado para criação de arquivos binários recebendo como entrada *scripts shell*.

3. Criptografia de um diretório que será montado como partição criptografada de dados.

A SecBox implementa a terceira forma pois não envolve o reparticionamento do disco, assim não é exigido que estas estejam totalmente livres. Nessa forma, são criados mecanismos para automatizar a manipulação das pastas criptografadas criadas a partir de um único diretório de arquivos com segurança dos dados.

O EncFS [7] apresentou-se como o sistema de arquivos criptografado mais viável para aplicação na SecBox, por não precisar de *patch* no *kernel*, como ocorre no TCFS [13]. Além disso, o EncFS está implementado em espaço de usuário, utilizando para tal tarefa um módulo do *kernel* denominado FUSE, enquanto que outros sistemas de arquivos criptografados são desenvolvidos a partir do NFS (*Network File System*). O CFS [14] é desenvolvido sobre o servidor NFS enquanto que o TCFS é montado sobre o cliente NFS.

Basicamente o EncFS tem seu funcionamento atrelado e baseado no módulo FUSE do *kernel*, provendo uma interface para o sistema de arquivos por meio do sistema de arquivos virtual (VFS) do *kernel*. As técnicas de criptografia utilizadas no EncFS são as de cifragem simétrica de blocos, tais como AES - padrão de criptografia avançada [15] e Blowfish [16]. O EncFS é um *software* livre protegido sob a licença GPL [17].

Na comparação entre os três sistemas (EncFS, TCFS, CFS), o EncFS foi o único pesquisado que suporta criptografia para arquivos maiores que 2 GB. Outra vantagem relaciona-se com o seu crescimento dinâmico não ocupando um espaço físico de volume fixo [18].

3.2 Cifras de Integridade

Qualquer técnica para integridade de dados utiliza a geração de cifras de única via, a qual tem a função de embaralhamento dos caracteres referente ao significado de uma mensagem de forma a não obtenção do retorno do conteúdo de origem. As cifras podem ser utilizadas de diferentes formas dependendo da aplicação. Para este projeto foi utilizado uma função de resumo sem chave do tipo *hash* (embaralhamento). Esses algoritmos têm três propriedades importantes que quando satisfeitas asseguram a correta integridade digital [19]:

- *Propriedade 1*: Dado um *hash*, é computacionalmente improvável encontrar a mensagem original, com esforço matemático de $2n^{8}$.
- *Propriedade 2*: Dado uma mensagem e o *hash* gerado por um algoritmo para esta mensagem, é computacionalmente improvável encontrar uma outra mensagem que utilizando o mesmo algoritmo de *hash*, gere o mesmo resumo, com esforço matemático de $2n$.
- *Propriedade 3*: É computacionalmente improvável encontrar duas mensagens distintas que, utilizando o mesmo algoritmo de *hash* gerem o mesmo resumo, com esforço matemático de $2n/2$.

[8] *N* representa unidades de tempo utilizadas para realizar este número de operações computacionalmente.

No desenvolvimento da SecBox, em sua primeira versão, utilizou-se o algoritmo *hash* MD5 (*Message Digest number 5*), porém a literatura reportou problemas encontrados na propriedade 3, mais especificamente este algoritmo é suscetível a colisões [20].

Tradicionalmente, os algoritmos de *hash* utilizam funções de compressão, que são modos de operação para cifra de bloco dedicada. Os algoritmos *hash* que apresentaram problemas de colisão, particularmente em sua maioria são os que utilizam compressão Davies-Meyer [21]. Segundo Ribeiro *et al.* [22], por si só, o modo Davies-Meyer não é problemático. Mas, quando combinado com encadeamento específico de dados, e utilizado uma cifra de baixa difusão, produz de fato uma função de *hash* suscetível a ataques de colisão multibloco.

Assim, foi pesquisado uma outra função de resumo sem chave para ser utilizado nesse sistema e escolheu-se o algoritmo *hash* Whirlpool [23], o qual possui características básicas diferentes do MD5. Utiliza-se compressão Miyaguchi-Preneel em vez de Davies-Meyer. Este algoritmo foi adotado no portfólio NESSIE, e incluso na norma ISO/IEC 10.118-3 [24]. Não possui até o presente momento ataques conhecidos e está disponível no *kernel* do Linux a partir da versão 2.6.11, facilitando sua utilização na SecBox e demais sistemas embarcados que utilizam este sistema operacional.

3.3 Esteganografia Digital

Uma técnica não comumente utilizada em sistemas embarcados é a esteganografia digital. A esteganografia provê meios de inserir informações ocultas no interior de mídias, sejam elas discretas ou contínuas. Essa inserção de informações ocultas tem como objetivo dificultar a obtenção de seu conteúdo por pessoas não autorizadas. A maioria dos trabalhos encontrados na literatura reportam a utilização de esteganografia em imagens. Pei e Zeng [25], apresentam uma técnica de esteganografia para inserção de marcas d'agua (*watermark*) em imagens JPEG. Já Matson e Ulieru [26] apresentam uma forma de esteganografia com utilização de marcas d'agua em sistemas de *e-Commerce* visando a preservação de direitos autorais. Os dois trabalhos remetem a introdução de marcas d'agua visando a preservação de direitos autorais, porém não exploram o uso de esteganografia para armazenamento de informações ocultas em mídias.

As técnicas de marca d'agua introduzem ruído no interior das mídias, geralmente em etapas de compressão com perda, corrompendo as informações no momento da recuperação. Estas técnicas verificam se existe ruído adicional à mídia analisada em regiões pré-determinadas para validar sua integridade. Já a técnica de esteganografia com abordagem de escrita secreta [27], explora estruturas presentes na etapa de compressão sem perda das mídias a serem esteganografadas, possibilitando a recuperação dessas informações na íntegra. Esta técnica tem como propriedade fundamental a garantia que os dados obtidos após o processo de esteganografia reversa sejam exatamente iguais aos dados adicionados à mídia na etapa de esteganografia das informações.

Para realização da etapa de esteganografia do conteúdo dos *hashs* em imagens verificou-se a possibilidade de aplicação de diversos algoritmos para este fim, tais como técnicas de filtragem e mascaramento e inserção de informações nos *bits* menos significativos (LSB).

As técnicas de filtragem e mascaramento são técnicas mais robustas que a inserção LSB (*Least Siginificant Bit*). Estas técnicas geram estego-imagens imunes à compressão e recorte. No entanto, são técnicas mais propensas a detecção (estego-análise) [28,29].

Já as técnicas baseadas em LSB podem ser aplicadas a cada *byte* de uma imagem de 24-*bits*. Essas imagens possuem seus *pixels* codificados em três *bytes*, um para cada canal de cor, vermelho (*red*), verde (*green*) e azul (*blue*). Seguramente, podemos selecionar um *bit* (o menos significativo) em cada *byte* do *pixel* para representar o *bit* a ser escondido, pois não acarretará modificações visuais na imagem.

A SecBox utiliza o algoritmo denominado JPEG-JSTEG [5], o qual realiza inserção de dados nos *bits* menos significativos (LSB) de imagens, obtendo no máximo 3% de distorção visual da imagem, o que é imperceptível ao olho humano. Esse algoritmo recebe como entrada imagens de formato GIF, TGA e BMP, além do arquivo em formato TXT, gerando como saída uma imagem no formato JPEG contendo as informações do TXT ocultas.

4. Técnicas de Segurança Empregadas na SecBox

Os mecanismos de segurança empregados na SecBox podem ser divididos, em três etapas distintas:

- *Integridade do Sistema*: Funções criptográficas de embaralhamento sem chave (*hash codes*) e verificações de *Timestamp* de geração dos arquivos do sistema operacional.
- *Proteção de arquivos do sistema e de conteúdo*: Sistema de arquivos criptografados (EncFS).
- *Proteção de informações de integridade do sistema*: Algoritmo de esteganografia para ocultamento de dados em imagens no formato JPEG.

Uma visão geral dessas etapas, com seus respectivos mecanismos de segurança, pode ser visualizada na Figura 2. Em 1, podemos verificar a etapa de validação dos *timestamps* dos arquivos do sistema. Essas variáveis são colhidas no momento da criação do sistema do *set-top box* e armazenadas para comparação e validação a cada inicialização. Esta etapa tem a função de garantir que qualquer alteração nos arquivos do sistema operacional ocasionem diferenças na comparação em seus tempos de criação.

A etapa 2, utiliza o EncFS [7] para criação de uma pasta criptografada de arquivos Ext3, a qual contém os *links* simbólicos[9] para os corretos arquivos de sistema. Essa pasta criptografada tem três funções bem definidas: garantir que os corretos arquivos de sistema sejam chamados na inicialização (2.1), proteger o conteúdo dos resumos *hashs* dos arquivos do sistema (2.2), proteger a imagem esteganografada contendo os resumos *hashs* das características físicas da máquina (2.3).

[9] Os *links* simbólicos têm a função de criar atalhos realizando o apontamento para um determinado arquivo presente no sistema. O comando utilizado para criação desses *links* no Linux é o *ln*.

Propõe-se a utilização do EncFS [17], para criação de uma pasta criptografada, uma vez que este sistema apresenta-se como evolução de outros sistemas que realizam a mesma função. Em relação a chave do EncFS, a cada inicialização, é gerado novamente a partir dos resumos *hashs* obtidos com os itens de *hardware* e arquivos sistemas atuais, assim estes valores são comparados com os originais para validação. Na SecBox, o usuário não realiza nenhuma forma de autenticação e sim a própria SecBox que gera essas chaves automaticamente.

Na etapa 2.2, são tratados diversos arquivos que representam informações únicas do sistema operacional. Para cada arquivo é gerado um resumo *hash* e estes são armazenados em um arquivo na pasta criptografada de dados. Já na etapa 2.3, são gerados resumos *hash* com as informações únicas das características físicas do *hardware* do *set-top box*, tais como barramentos, dispositivos físicos conectados, número do processador, entre outros. Estes resumos são armazenados em um arquivo temporário. Tal arquivo é então armazenado, via esteganografia, em uma imagem (um logo ou uma imagem de fundo de tela). Após a esteganografia, o arquivo temporário é apagado. Na inicialização da máquina, quando ativado o processo de esteganografia reversa, este arquivo é recuperado para possibilitar que validações sejam realizadas.

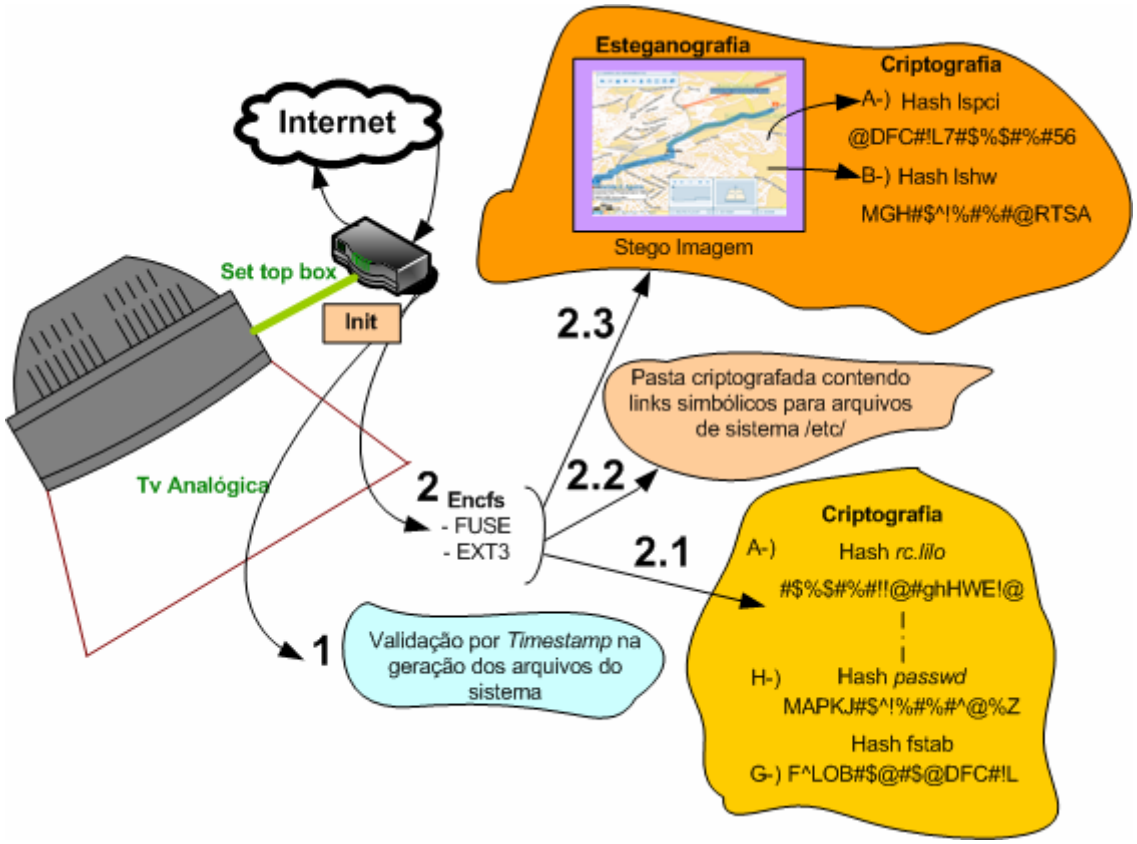

Figura 2. Diagrama geral do sistema.

5. Funcionamento da SecBox

A SecBox é ativada na inicialização do sistema operacional do *set-top box* pelo processo pai *Init.* Após ativada, a SecBox realiza validações e verificações utilizando as

técnicas descritas na seção anterior. Tais análises podem ser divididas em seis fases. Essas fases podem ser visualizadas no fluxograma presente na Figura 3.

A primeira fase consiste na validação dos *links* simbólicos concebidos no momento de criação de arquivos importantes do sistema, tais como:

- /etc/group: Arquivo que abriga as definições de grupo.
- /etc/lilo.conf: Arquivo de *boot loaders* utilizado pelo gerenciador de *boot* LILO.
- /etc/sudoers: Lista de usuários com permissão de *root.*
- /etc/passwd: Arquivo de senhas e informações de *login.*
- /etc/shadow: Senhas criptografadas do passwd.
- /etc/securetty: Terminais disponíveis ao *root.*
- /etc/fstab: Descritor dos sistemas de arquivos que podem ser montados.

Nessa fase é verificado se os arquivos de sistema do S.O. foram movidos de sua localização original no sistema de arquivos. Feito isso, na segunda fase são verificados os *timestamps* de geração dos arquivos de sistema verificando se os mesmo foram alterados.

Na fase III é montada a pasta do sistema de arquivos criptografado, via EncFS, a qual contém a imagem esteganografada. Realiza-se a esteganografia reversa da imagem, recuperando os dois resumos *hashs* gerados a partir da saída dos comandos do Linux *lspci* e *lshw* [12], os quais descrevem o *hardware* do *set-top box.*

A fase IV verifica a autenticidade dos *hashes* da fase III. Na fase V, são montadas as pastas normais do sistema operacional bem como as pastas criptografadas contendo o conteúdo digital dos usuários. E, finalmente, na fase VI, é realizada uma comparação entre os valores dos *hashes* atuais com os valores gerados anteriormente no momento de fabricação do *set-top box*.

Caso tenha havido alterações nos arquivos de sistema, tal alteração será acusada nas fases I, II e V, as quais utilizam técnicas de segurança descritas por Ravi e Raghunathan [10] para implementar a segurança do sistema operacional proposta por Kocher *et al* [1].

Na fase I serão indicadas irregularidades na verificação dos *links* simbólicos, caso haja tentativa por um usuário de mover algum arquivo de sistema. Já na fase II, não serão validados os *timestamps* caso um usuário edite algum arquivo de sistema. A fase III adiciona o ocultamento das informações criptografadas em uma imagem esteganografada, servindo de pré-requisito para as fases IV e V. Dessa forma, mesmo se um intruso conseguir a senha de superusuário terá que conseguir infiltrar-se na pasta criptografada em busca das cifras de *hardware,* as quais não serão encontradas, pois estão ocultas em uma imagem.

Para assegurar a segurança física do sistema descrita por Kocher *et al.* [1] e evitar tentativas de intrusão por agregação de novos dispositivos físicos [9], foi desenvolvida, na fase IV, um modo de verificação caso um usuário final queira intervir com a inclusão de um dispositivo de armazenamento de dados (*pen drives* e HDs) para copiar dados. Para implementação da segurança no armazenamento descrita na

classificação de Kocher *et al.* [1], foi implementada a fase V, a qual permitirá o acesso aos arquivos de conteúdo multimídia do usuário.

Se uma das seis fases não fizer a correta validação, o sistema irá desligar automaticamente sem a intervenção do usuário final. Se o sistema passar com resultados positivos por todas as etapas, ele irá prosseguir com seu funcionamento normal. Dessa forma, nestas fases, foi obtido um alto grau de segurança implementando técnicas descritas na literatura. Além disso, foram inseridas técnicas não usuais para proteção do sistema, como a fase III, no funcionamento da SecBox.

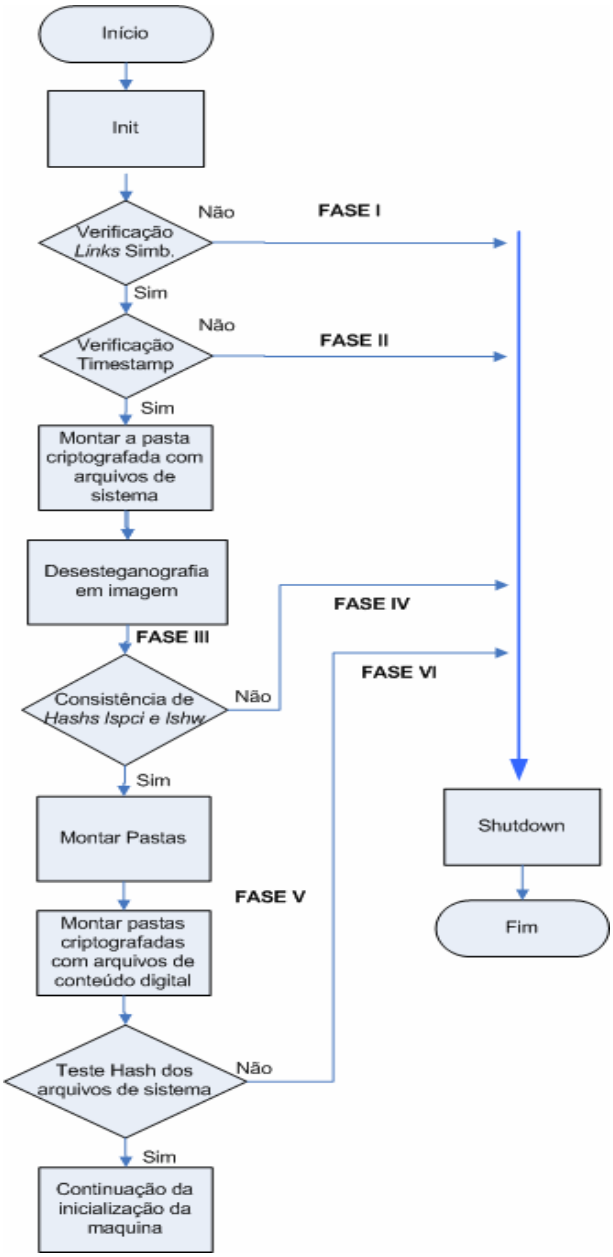

Figura 3. Fluxograma de execução do sistema.

6. Arquitetura do Sistema

A arquitetura da SecBox por utilizar sistemas de arquivos criptografados está disposta em duas camadas. A primeira camada tem funcionamento no espaço de *kernel*

do Linux com o módulo FUSE e *Virtual File System* (VFS). A segunda camada do sistema está presente em espaço de usuário e implementa a pasta criptografada de dados por meio do EncFS. Além dessas camadas, a SecBox utiliza a partição principal do sistema para implementar as outras fases, particularmente as fases I e II da Seção 5. Na Figura 4, é apresentado um diagrama para visualização da arquitetura da SecBox.

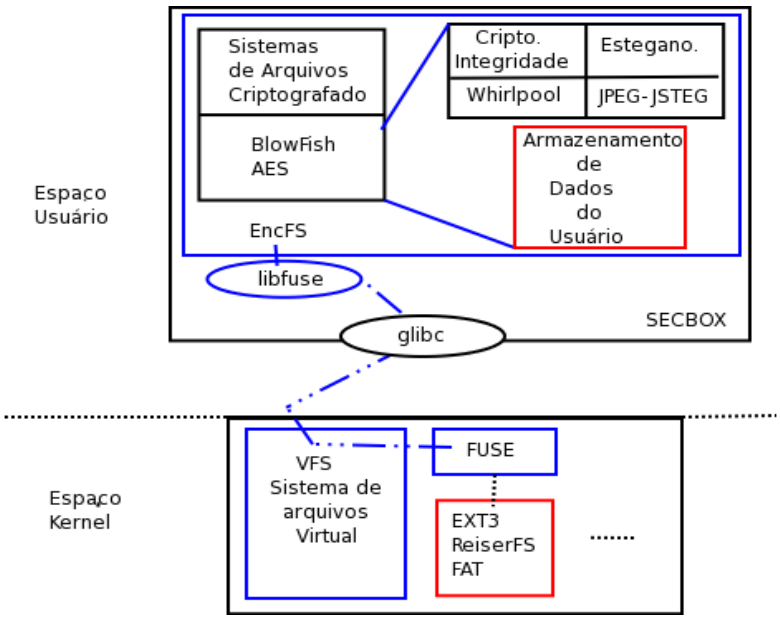

Figura 4. Arquitetura da SecBox.

O VFS é um sub-sistema do *kernel* que implementa as interfaces para que os programas de usuários acessem ao sistema de arquivos. Um fato interessante é que todos os sistemas de arquivos parecem iguais para o *kernel* quando é utilizado o VFS. Assim, a partição de armazenamento de dados do mesmo modo que a pasta criptografada pode utilizar quaisquer sistemas de arquivos, tais como Ext3, ReiserFS e FAT.

A *glibc* é uma biblioteca de funções que serve de suporte para todos os programas escritos em C dentro do sistema operacional. É por meio dessa biblioteca que é realizada a comunicação do VFS (espaço de *kernel*) com a *libfuse* (espaço de usuário). A biblioteca em espaço de usuário *libfuse* utiliza a *glibc* para comunicação com o módulo FUSE do *kernel*. O módulo FUSE do *kernel* e a *libfuse* lêem um arquivo descritor de arquivos da partição. O EncFS altera esse descritor de arquivos atualizando todas as alterações realizadas na partição criptografada. Esse descritor de arquivos é lido diversas vezes sendo o primeiro no momento da montagem da pasta criptografada na inicialização da máquina.

7. Considerações Finais

Este trabalho explorou a utilização de uma abordagem híbrida de segurança, combinando técnicas de criptografia e esteganografia, como uma alternativa para proteção do sistema e do conteúdo (filmes e músicas, por exemplo) de *set-top boxes*. Como resultado, a SecBox apresenta-se como uma alternativa viável e de baixo custo contra cópias ilegais de conteúdo e alterações no funcionamento de sistema embarcados baseados em Linux.

Vale destacar que as tentativas de cópias via *hacking* remoto são tratadas da maneira usual, por meio de técnicas de criptografia. A novidade está na utilização da esteganografia para ocultar os resumos *hashes*. Assim, mesmo que seja obtida a senha de superusuário, ainda assim não se sabe onde alguns *hashs* estão escondidos.

A SecBox protege em 3 níveis as informações referentes as características pertinentes de dispositivos físicos conectados ao *set-top box* e do sistema operacional. Para um ataque visando alteração desses *hashs* ser bem sucedido terá que corromper esses três níveis, a saber:

1. Quebra da pasta criptografada: algoritmos de cifras simétricas AES e Blowfish.
2. Quebra da imagem esteganografada: É sabido que *softwares* de estegoanálise como *Outguess* [30] alcançam sucesso em ataques sobre o JPEG-JSTEG. Porém, esses *softwares* não conseguem identificar a esteganografia quando a mensagem inserida em imagem é menor ou igual a 150 *bytes* [31]. Os dois resumos *hashs* presentes na imagem da SecBox contém aproximadamente 68 *bytes*.
3. Alteração dos resumos *hashs* (Whirlpool) armazenados na imagem esteganografada.

Para validar tentativas de violação física, no próprio *hardware* do *set-top box* foram inseridos novos dispositivos no sistema - na tentativa de copiar conteúdo – como memórias *flashs*, gravadores CD/DVD ROMs e HDs. Em todos os casos o sistema não completou o processo de inicialização, efetuando o desligamento da máquina após a verificação da alteração no sistema original.

A SecBox foi instalada na distribuição *Slackware* versão 11.0. Porém essa técnica pode ser adaptada para qualquer distribuição Linux com quaisquer sistemas de arquivos, desde que sejam atualizados os *paths* para os arquivos de sistema que serão verificados em seu funcionamento. O funcionamento da SecBox é realizado na inicialização do sistema operacional. Em relação ao seu desempenho e atraso acrescentado ao processo normal de inicialização de um sistema Linux convencional com sistema de arquivos Ext3, foi verificado que a SecBox retardou a inicialização da máquina em alguns segundos. Vale destacar que, após a inicialização do sistema, o mesmo não sofre nenhuma influência da SecBox. Podemos concluir que, dado o benefício obtido de segurança, a queda de desempenho somente na inicialização é perfeitamente aceitável.

Para o armazenamento de dados referentes ao conteúdo digital, a SecBox apresenta uma queda de performance em relação ao armazenamento em partição de dados comum, porém adiciona a segurança necessária para que os mesmos não sejam acessados quando retirado o disco rígido de um *set-top box.* Além disso, esses *set-to boxes*, nos últimos anos, estão apresentando uma evolução significativa de processamento.

Como limitações podemos citar que a SecBox funciona somente em sistemas embarcados que utilizam Linux, e há ausência de um mecanismo de atualização dos dados de segurança do sistema inviabilizando a atualização do mesmo.

Como trabalhos futuros citam-se: a implementação de soluções para os outros problemas de segurança propostas por Ravi [10], a realização de testes verificando o quanto é acrescentado de processamento ao sistema e verificação da lentidão

acrescentada na leitura/escrita aos arquivos de conteúdo digital presentes na pasta criptografada, e por fim a implementação de um mecanismo de alterações dos arquivos das diversas fases da SecBox para que se possa, por exemplo, proceder atualizações de *hardware* ou trocas de *hardware* defeituosos.

Referências

[1] Kocher P., Lee R., McGraw G., Raghunathan A. and Ravi S., "Security as a New Dimension in Embedded System Design", DAC 2004, June 7-11, San Diego, California, USA.

[2] T. Jiang et al, "Secure Communication between Set-top Box and Smart Card in DTV Broadcasting," IEEE Trans. on Consumer Electronics, Vol. 50, No. 3, August. 2004, pp.882--886.

[3] R. Goularte. "Personalização e adaptação de conteúdo baseadas em contexto para TV interativa". Tese de doutorado – ICMC – USP, São Carlos, 2003.

[4] Sky "Set Top Boxes supported by TiVo" – TIVO 2005 completer *on line http://www.garysargent.co.uk/tivo/stb.htm,* acessado em outubro de 2006.

[5] Chang *et al*. "A steganographic method based upon JPEG and quantization table modification" Information Science - An International Journal, 2000.

[6] William E. Burr, "Cryptographic Hash Standards: Where Do We Go from Here?," *IEEE Security and Privacy*, vol. 4, no. 2, pp. 88-91, Mar/Apr, 2006.

[7] Gough V. "The EncFS" *on line http://arg0.net/wiki/encfs,* acessado em novembro de 2006.

[8] Toshiba, America, "HD-XA2 Specifications", Disponível em http://www.tacp.toshiba.com/dvd/hddvd.asp, visitado em janeiro de 2007.

[9] Koopman, Philip. Morris J., Narasimhan P., "Challenges In Deeply Networked System Survivability" Workshop Embedded Security NATO August, 2005.

[10] S. Ravi and A. Raghunathan, "Security in Embedded Systems: Design Challenges", *ACM Transactions on Embedded Computing Systens*, Vol. 3, 3, August 2004, 461-491.

[11] García F. J. R.. "Shc Generic shell script compiler". Disponível em http://www.datsi.fi.upm.es/~frosal/sources/shc.html, acessado em dezembro de 2006.

[12] Lyonel Vincent, "Hardware Lister – lshw". Disponível em http://ezix.org/project/wiki/HardwareLiSter, acessado em novembro de 2006.

[13] M. Blaze. "A Cryptographic File System for Unix". In Proceedings of the first ACM Conference on Computer and Communications Security, páginas 9–16, Fairfax, VA, 1993. ACM.

[14] J. Black and H. Urtubia, "Side-channel attacks on symmetric encryption schemes: The case for authenticated encryption," in Proc. 11th USENIX Security Symposium, pp. 327–338, 2002.

[15] Joan Daemen, Steve Borg e Vincent Rijmen. "The Design of Rijndael: AES - The Advanced Encryption Standard." Springer-Verlag, 2002.

[16] B. Schneier. “Description of a New Variable-Length Key, 64-Bit Block Cipher (Blowfish)”, *Fast Software Encryption, Cambridge Security Workshop Proceedings (December 1993)*, Springer-Verlag, 1994, pp. 191-204.

[17] Pletka R., Cachin C., “Cryptographic Security for a High-Performance Distributed File System”, IBM Zurich Research Laboratory, Setembro de 2006.

[18] Zadok E., Iyer R., Joukov N., Sivathanu G. and Wright C. P., “On incremental file system development,” *ACM Transactions on Storage*, vol. 2, pp. 161–196, May 2006.

[19] Rivest R., Shamir A., Adleman L.. “A Method for Obtaining Digital Signatures and Public-Key Cryptosystems”. Communications of the ACM, Vol. 21 (2), pp.120–126. 1978. Previously released as an MIT "Technical Memo" in April 1977.

[20] Klima V., "Tunnels in Hash Functions: MD5 Collisions Within a Minute", *Cryptology ePrint Archive*, Report 2006/105, disponível em http://eprint.iacr.org/2006/105.pdf, acessado em Janeiro de 2007.

[21] HOSNER, Charlie (2005), “Security Elite hash out Encryption Alternatives”. Disponível em http://software.newsforge.com/article.pl?sid=05/11/02/2056250&from=rss, acessado em janeiro de 2007.

[22] RIBEIRO, C.H.C. ; HIRA, A. Y. ; ZUFFO, M. K. . “Aplicação da técnica de duplo hash na implementação de serviços de integridade em registros eletrônicos de saúde”. In: Congresso Brasileiro de Informática, 10., 2006, Florianópolis. CBIS'2006. Florianópolis : SBIS, 2006.

[23] P. S. L. M. Barreto, V. Rijmen, "The WHIRLPOOL Hashing Function," *First open NESSIE Workshop*, Leuven, 13-14 November 2000.

[24] Paulo S. L. M. Barreto, "A Crise das Funções de Hash" *Invited talk* (in Portuguese), VII Simpósio Segurança em Informática -- SSI'2005, São José dos Campos, Brazil, November 08--11, 2005.

[25] Pei S.C., Zeng Y.C. “Tamper Proofing and Attack Identification of Corrupted Image By using Semi-fragile Multiple-watermarking Algorithm”, ASIACCS, 2006, Taiwan.

[26] Merv Matson, Michaela Ulieru “Persistent Information Security – Beyond the e-Commerce Threat Model”, ICEC, ACM, Canada 2006.

[27] Zhao, J. “In business today and tomorrow”. ACM Communications of the ACM, p. 7, 1998.

[28] Peter Wayner. “Disappearing cryptography”. Morgan Kaufmann Publishers, San Francisco, CA, USA, second edition, 2002. ISBN 1-55860-769-2.

[29] Richard Popa. “An analysis of Steganography Techniques”. Master's thesis, The Polytechnic University of Timisoara, Timisoara, Romênia, 1998.

[30] Provos Niels, Honeyman P., “Detecting Steganographic Content on the Internet”, *ISOC NDSS'02*, San Diego, CA, February 2002.

[31] Provos Niels, Honeyman P. “Hide and Seek: An Introduction to Steganography”. IEEE Security & Privacy, Maio/Junho de 2003.