MICHEL HENRIQUE AQUINO SANTOS

COMPARAÇÃO DE FERRAMENTAS PARA VERIFICAÇÃO DE INTEGRIDADE DE ARQUIVOS

Lavras – MG

2012

MICHEL HENRIQUE AQUINO SANTOS

COMPARAÇÃO DE FERRAMENTAS PARA VERIFICAÇÃO DE INTEGRIDADE DE ARQUIVOS

Monografia apresentada ao Colegiado do Curso de Ciência da Computação, para a obtenção do título de Bacharel em Ciência da Computação.

Orientador

Prof. Dr. Joaquim Quinteiro Uchôa

Lavras – MG

2012

MICHEL HENRIQUE AQUINO SANTOS

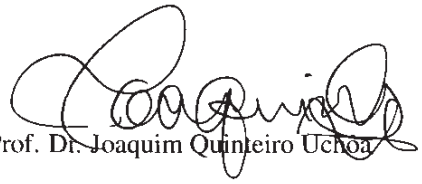
COMPARAÇÃO DE FERRAMENTAS PARA VERIFICAÇÃO DE INTEGRIDADE DE ARQUIVOS

Monografia apresentada ao Colegiado do Curso de Ciência da Computação, para a obtenção do título de Bacharel em Ciência da Computação.

APROVADA em 26 de Outubro de 2012.

Prof. Dr. José Monserrat Neto

Prof. MSc. Eric Fernandes de Mello Araújo



Prof. Dr. Joaquim Quinteiro Uchoa

(Orientador)

LAVRAS – MG

2012

Dedico este trabalho aos meus Pais, Arlete e Elio. Sem o seu apoio e confiança nenhuma palavra deste trabalho estaria escrita.

AGRADECIMENTOS

À minha família pelo apoio constante. Aos amigos da turma de Ciência da Computação 2008/2 por boas risadas e companheirismo e a todos que de alguma forma ajudaram na conclusão desse trabalho.

RESUMO

A internet vem se tornando um dos principais meios de comunicação, e com isso o número de tentativas de ataque aos sistemas computacionais cresce cada vez mais. Diante de tal fato torna-se cada vez mais necessário a proteção de sistemas e dados. Uma das primeiras ações de um invasor é a modificação ou substituição de arquivos do sistema afim de apagar os rastros de sua invasão e/ou facilitar futuras invasões. As ferramentas de verificação de integridade de arquivos são uma sub-classe das IDS (*Intrusion Detection System*) e tem por função monitorar o sistema computacional afim de detectar eventuais modificações nos arquivos checados, dando ao administrador do sistema a possibilidade de restaurar o sistema caso aconteça alguma alteração. O objetivo deste trabalho é analisar e comparar as ferramentas de verificação de integridade de arquivos Tripwire, AIDE, OSSEC e Samhain, afim de ajudar o administrador do sistema a escolher a ferramenta mais adequada para o ambiente administrado.

Palavras-chave: Segurança computacional, Integridade de arquivos, Ferramentas de verificação de integridade de arquivos, IDS

ABSTRACT

The internet is becoming the main way of communication, thus the number of attacks in computational systems grows more and more. Thereby, it is necessary to protect the systems and its data. It is common the first action of an attacker to be the modification or replacement of systems' files, with the mind in remove the invasion's traces or facilitate future invasions. The intrusion detection systems are a sub-class of IDS (Intrusion Detection System) and has the function of monitoring the computational system, detecting modifications in checked files and offering to the system administrator the possibility of restore the system if some change occurs. The objective of this study is analyse and compare intrusion detection systems like Tripwire, AIDE, OSSEC and Samhain, helping the system's admnistrator choose the most appropriate tool for his enviroment.

Keywords: Computer security, File integrity, Tools for checking integrity of files, IDS

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

1 Introdução

A internet vem se tornando um dos principais meios de comunicação da humanidade, sendo que os mais variados dispositivos usam este meio. Computadores pessoais, até celulares, *tablets* e dispositivos de bolso, todos esses dispositivos têm acesso à internet. Muitos dados gerados atualmente são sigilosos e precisam ser protegidos.

Notícias vinculadas a invasões e ataques a sistemas computacionais são cada vez mais comuns na mídia. Os invasores, chamados comumente de *hackers*, aproveitam de sistemas mal projetados e vulneráveis para invadir, tomar posse do sistema e obter dados sigilosos. Além disso, os invasores podem alterar páginas de instituições, modificando seu conteúdo e denegrindo a imagem da instituição.

Os prejuízos causados por problemas de segurança computacional são significativos, indo desde a fuga de clientes até inoperância da empresa por falhas em seus servidores (UCHÔA, 2009). Os problemas não afetam somente usuários de computador e internet. Pessoas que têm seus dados furtados em sistemas bancários ou de empresas, são atingidas indiretamente.

Novos mecanismos, técnicas e normas de segurança são constantemente desenvolvidos na tentativa de prevenir e minimizar os efeitos de uma invasão. Dentre as categorias de mecanismos criados estão os sistemas detectores de intrusos, mais conhecidos como IDS (*Intrusion Detection System*). Segundo Lima (2002), os IDS estão sendo largamente utilizados nas corporações, instituições governamentais e redes de computadores acadêmicas, como uma importante e poderosa ferramenta de auxílio aos administradores de segurança.

Mesmo com a grande quantidade de mecanismos, técnicas e normas relacionados à segurança computacional, muitas vezes os sistemas computacionais são inseguros e sujeitos a ataques. Isso se deve principalmente à falta de uma abordagem correta para escolha e implementação dos mecanismos de segurança. Muitos profissionais habilitam recursos sem considerar as reais necessidades da organização e isso acaba gerando gastos desnecessários, bem como ampliando a vulnerabilidade do sistema.

Diante de tal contexto, este trabalho tem por objetivo estudar e comparar ferramentas de verificação de integridade de arquivos, a fim de definir qual ferramenta é melhor em determinado cenário.

A metodologia para o estudo e comparação das ferramentas segue o modelo proposto inicialmente por (DOMINGUES, 2003), que compara ferramentas com as mesmas funcionalidades. Apesar disso, pretende-se adotar novos critérios de avaliação propostos por (LABS, 2006). Cabe ressaltar ainda que a comparação apresentada em (DOMINGUES, 2003) e Labs (2006) já não atendem às ferramentas atuais, que foram atualizadas e novas ferramentas foram criadas. As ferramentas a serem comparadas no presente trabalho são todas ferramentas livres e muito utilizadas, são elas, AIDE, Tripwire, OSSEC e Samhain.

O trabalho está organizado da seguinte forma: no Capítulo 2 descreve o principal tema do trabalho, integridade de arquivos. Já no Capítulo 3 é apresentada a metologia utilizada no trabalho, bem como as ferramentas estudadas. No Capítulo 4 é apresentado a comparação e os resultados obtidos. Por fim, no Capítulo 5 são apresentadas as conclusões do trabalho.

2 Integridade de Arquivos

2.1 Considerações Iniciais

O sistema de arquivos de um sistema operacional é um de seus subsistemas mais importantes. Nele são mantidos todos os arquivos pertencentes ao sistema, dentre arquivos executáveis, arquivos de configuração dos serviços e utilitários, arquivos dos usuários e o próprio sistema operacional. Com a modificação de alguns desses arquivos, é possível alterar e até mesmo comprometer todo o funcionamento do sistema operacional a ponto de não funcionar mais.

Diante de tal fato, o sistema de arquivos é um dos principais alvos de ataque e segundo Quynh e Takefuji (2007), quase todos os incidentes de segurança, resultam em arquivos críticos alterados. Um vez que um invasor consiga acesso privilegiado ao sistema, ele pode alterar, incluir e excluir arquivos do sistema, com o intuito de esconder rastros da invasão, garantir futuros acessos e alterar configurações a fim de obter mais privilégio no sistema.

Exemplos dessa prática são os *backdoors* e *rootkits*. *Backdoors* são utilizados para garantir futuros acessos ao sistema, criando outros meios de entrada a partir de uma vulnerabilidade não corrigida. *Rootkit* é um aplicativo (ou conjunto de aplicativos) com o objetivo de garantir poderes de superusuário (root) ao invasor (UCHÔA, 2009). Geralmente consistem de aplicativos alterados para funcionar de forma especial ou até mesmo versões alteradas do próprio *kernel* do sistema operacional.

Nesse contexto, o controle de integridade de arquivos passa a ser um recurso de extrema importância já que sem ele, atacantes passam despercebidos ao invadirem um sistema, inserindo códigos maliciosos ou alterando configurações do

sistema. Segundo Bace e Mell (2001) verificadores de integridade de arquivos são programas de computador que protegem sistemas operacionais, monitorando as mudanças nos arquivos críticos. Cabe ressaltar ainda que o mecanismo de verificação de integridade de arquivos é modelado e implementado seguindo as políticas de segurança do ambiente, e só emitirá alertas quando violar essa política (QUYNH; TAKEFUJI, 2007).

Este capítulo é baseado nos estudos de (SERAFIM, 2002), e tem como objetivo apresentar as principais atividades relacionadas a manutenção de integridade de arquivos.

2.2 Controle de Integridade de Arquivos

Para que o controle de integridade de arquivos possa ser feito, um processo chave é necessário: a comparação de objetos. Para isso é necessário que o objeto, cuja a integridade será futuramente verificada, seja copiado (duplicado). Quando a checagem da integridade do objeto é necessária, compara-se a versão original com a cópia armazenada. A cópia de objetos para fins de controle de integridade é chamada de instantâneo ou *snapshot*.

Esse processo é facilmente aplicado aos arquivos de um sistema de arquivos, realizando uma cópia dos arquivos para posterior comparação dos originais com as cópias feitas. O processo de comparação pode obter os seguintes resultados:

Nenhuma violação detectada: no momento da verificação, os arquivos estão íntegros;

Violações não autorizadas detectadas: no momento da verificação foram detectadas arquivos indevidamente alterados. Podem ser facilmente substituídos a partir do *snapshot*;

Violações legítimas detectadas: os arquivos foram alterados, porém com autorização. Nesse momento o *snapshot* desse arquivo precisa ser atualizado, para refletir o novo estado do arquivo.

Se todos os arquivos forem duplicados irá gerar problemas relacionados com o espaço de armazenamento no sistema, pois todo o espaço ocupado seria duplicado. Além disso a comparação dos arquivos *byte* a *byte* ocasionaria um desperdício de tempo.

A forma de se resolver esse problema consiste de utilizar funções matemáticas (funções de *Hash*), que aplicadas ao conteúdo de um arquivo, gera um resumo matemático do mesmo. Esse resumo geralmente possui um tamanho pequeno e fixo de *bits*, que depende da função utilizada, e pode variar entre 16 e 256 *bits*. Assim diversos conjuntos de *bytes* (arquivos) são identificados, diferindo em conteúdo e tamanho, através de sua representação por uma sequencia de *bits*, que recebe o nome de digital ou *fingerprint*.

Através da utilização dessas funções matemáticas, o *snapshot* não precisa ser uma cópia idêntica de cada arquivo, economizando espaço de armazenamento do *snapshot*. Outra vantagem da utilização de funções matemáticas para gerar o *fingerprint* é a velocidade do processo de geração do *snapshot* e de verificação da integridade de arquivos. Isso torna possível a criação de mecanismos eficientes de controle de integridade.

Independente de como o *snapshot* é composto, o processo de controle de integridade possui pontos sensíveis e potencialmente vulneráveis. O inadequado tratamento desses pontos, pode dar a um invasor a chance de intervir e invalidar o processo de verificação de integridade. Dessa forma, é extremamente importante que tais pontos sejam discutidos e analisados.

2.3 Funções de *Hash*

Segundo Stallings (2010), funções de *hash* são funções matemáticas que tem como entrada dados (*strings*) de comprimento variável e como saída geram uma sequência de dados de tamanho fixo. Formalmente, uma função de *hash* H aceita como entrada um bloco de dados de comprimento variável M, gerando um valor de *hash* de tamanho fixo $h = H(M)$. Desempenham um importante papel na criptografia, fazendo parte de muitos protocolos criptográficos existentes.

Ainda segundo Stallings (2010), uma boa função de *hash* tem a propriedade de que os resultados da aplicação da função para um grande conjunto de entradas, produzirá resultados uniformemente distribuídos e aleatórios. Em termos gerais, o objetivo principal da função de *hash* é a integridade dos dados. Uma mudança de qualquer *bit* ou *bits* no resultado de M implicará, com grande probabilidade, em uma mudança no resultado h.

De modo geral, uma função de *hash* H deve ter as seguintes características (STALLINGS, 2010):

- **Independência do tamanho da entrada:** H pode ter como entrada um bloco de dados de qualquer tamanho;
- **Compressão**: H produz uma saída de tamanho fixo;

- **Facilidade de cálculo:** Para qualquer x dado na entrada, $H(x)$ deve ser fácil de calcular;
- **Resistência a colisões fracas:** Para qualquer dado valor de *hash* h, é computacionalmente impraticável encontrar y tal que $H(y) = h$;
- **Resistência a colisões fortes:** Para qualquer dado bloco x, é computacionalmente impraticável encontrar $y \neq x$ com $H(y) = H(x)$. A diferença dessa em relação à anterior, é que o atacante tem liberdade para escolher x e y;
- **Resistência a colisões próximas:** É impossível computacionalmente encontrar qualquer par (x, y) tal que $H(x) = H(y)$ difiram em apenas um número pequeno de *bits*;
- **Inexistência de correlação:** Os *bits* de entrada x e da saída $H(x)$ não devem ser relacionados, afim de que cada *bit* da entrada afete todos os *bits* da saída.

Essas características tornam as funções de *hash* ferramentas extremamente úteis para o controle de integridade de arquivos, permitindo a criação de uma representação compacta e única de uma determinada massa de dados. Entre as funções de *hash* mais utilizadas podemos citar: *MD5 (Message-Digest algorithm 5)* e *SHA1 (Secure Hash Algorithm)*. Depois de encontradas falhas nas funções MD5 (WANG; YU, 2005) e SHA1 (WANG; YIN; YU, 2005), sendo possível encontrar colisões de forma eficiente, é recomendado usar funções mais fortes como SHA-224, SHA-256, SHA-384 ou SHA-512 (KACZMAREK; WROBEL, 2008).

2.4 Armazenamento do *Snapshot*

O momento em que o *snapshot* é gerado não é o único em que o processo está vulnerável. O período de armazenamento é um ponto de vulnerabilidade, e armazenar o *snapshot* de forma adequada é uma procedimento básico para que se possa fazer o controle de integridade.

Em caso de perda do arquivo do *snapshot*, é impossível realizar a verificação de integridade dos arquivos monitorados. Outro problema é a modificação do *snapshot* por parte de um invasor, a fim de mascarar sua invasão, refletindo no *snapshot* as alterações feitas no sistema por ele. As duas situações são comprometedoras, porém a perda do *snapshot* refletirá em um dano potencialmente menor, já que no segundo caso a verificação da integridade do sistema irá assegurar ao administrador que o sistema está integro, o que não é verdade. Sendo assim, o armazenamento adequado do *snapshot* é um fator crítico em qualquer método de controle de integridade de arquivos.

2.5 Atualização do *Snapshot*

Ao longo do uso do sistema, as tarefas administrativas geram modificações intencionais e não maliciosas no sistema de arquivos. Tais tarefas podem ser aplicações de correções dos aplicativos do sistema, instalação de novos programas, atualização dos programas existentes e configurações executadas pelo administrador.

Independentemente de qual tarefa está sendo executada, há uma alteração no *snapshot*, que precisa ser atualizado a fim de refletir o novo estado íntegro do sistema. A falta de conhecimento do que foi alterado no sistema de arquivos pode

concretizar uma modificação maliciosa feita por um atacante. Portanto a tarefa de atualizar o *snapshot* é de extrema importância e é tão sensível quanto à sua criação.

O incidente da modificação do *snapshot* sem o conhecimento do administrador, não pode ser evitado por nenhum mecanismo, visto que é uma falha humana. Porém é possível reduzir o risco do comprometimento do *snapshot*, através do emprego de procedimentos adequados por parte do administrador:

Verificação da integridade de arquivos do sistema: deve ser feito antes da alteração dos arquivos, para se ter certeza que o sistema encontra-se em um estado íntegro. Qualquer alteração não permitida deve ser investigada e corrigida pelo administrador.

Execução das alterações: Após ter executado o primeiro passo, pode-se realizar as alterações.

Nova verificação da integridade dos arquivos do sistema: tem por objetivo levantar as modificações realizadas no segundo passo. Todas as modificações encontradas devem ser relacionadas àquelas feitas no segundo passo e aquelas não relacionadas, devem ser tratadas como violações e devem ser investigadas e solucionadas.

Atualização do *snapshot*: Após todos os passos serem feitos, atualiza-se o *snapshot*.

Com a adoção de tais procedimentos, o administrador pode reduzir drasticamente as chances de comprometimento do *snapshot*.

2.6 Restauração dos Arquivos Violados

A detecção de uma anomalia não prevista em um sistema, não é suficiente para fornecer segurança a esse sistema. É necessário algum modo em que o sistema possa ser restaurado, corrigindo a anomalia. Os mecanismos de controle de integridade baseados em resumos matemáticos não são capazes de corrigir uma violação, pois não é possível obter o arquivo original a partir do resumo. Sendo assim, um sistema de controle de integridade deve ser implementado juntamente com uma prática relativamente simples: a realização de cópias, ou *backups*.

As cópias de segurança são indicadas para arquivos particulares, que não existem nas mídias originais de instalação. A inexistência de cópias de segurança pode levar o sistema a um estado inseguro e inválido, já que não é possível recuperalo, caso haja alguma violação. Essa situação é inadmissível do ponto de vista de segurança e administrativo, porém caso ocorra, pode recorrer a paralisação e reinstalação de partes ou de todo o sistema. Embora a realização de cópias de segurança seja de extrema importância, não é o foco desse trabalho.

2.7 Identificação de Ações Maliciosas

O mecanismo de controle de integridade de arquivos, não indica quando uma violação foi intencional ou maliciosa nos arquivos monitorados. Cabe ao administrador do sistema investigar as violações detectadas e decidir se é ou não maliciosa usando ferramentas apropriadas e seu conhecimento. As ferramentas existentes que podem auxiliar o administrador consistem de programas que varrem todo o sistema a procura de códigos maliciosos como vírus, *backdoors* e *rootkits*, o re-

gistro de operações do sistema (arquivos de *log*) e ferramentas para análise dos *logs*.

Segundo Patil *et al.* (2004) a tarefa dos administradores ao decidir se uma violação é ou não maliciosa, consiste de três passos:

1. Em primeiro lugar, os administradores devem detectar que uma intrusão ocorreu e que o sistema está em um estado inconsistente.

2. O administrador tem que investigar os danos causados pelo atacante, como a exclusão de dados e inserção de código malicioso.

3. Por último, o administrador deve corrigir a vulnerabilidade para evitar futuros ataques.

Esses passos muitas vezes são difíceis, e na maioria dos casos os sistemas são reinstalados e reconfigurados quando uma ação maliciosa acontece.

Mesmo com o emprego de tais ferramentas, o fator humano é de extrema importância para o bom funcionamento do sistema de controle de integridade. O conhecimento e experiência do administrador do sistema, muitas vezes são determinantes no sucesso de uma investigação de violações ocorridas nos arquivos do sistema.

2.8 Integridade do Mecanismo de Controle

Por ser um dos responsáveis pela segurança e integridade do sistema, o mecanismo de controle de integridade de arquivos acaba por se tornar um alvo dos ataques. Se o mecanismo for comprometido o atacante derrubará uma importante barreira para o sucesso de seus propósitos, podendo inverter o papel do mecanismo

e usá-lo ao seu favor, fazendo com que o administrador do sistema e os usuários tenham uma falsa sensação de segurança ao pensar que o mecanismo de controle de integridade funciona perfeitamente, resultando assim na não investigação de possíveis comportamentos estranhos do sistema.

Desse modo, o mecanismo de controle de integridade deve ser regularmente investigado a procura de possíveis violações no mesmo a fim de garantir sua integridade. Esse procedimento é necessário e altamente recomendável, pois o comprometimento do mecanismo pode resultar em uma grave vulnerabilidade. Caso detecte que o mecanismo foi comprometido, conclui-se que uma vulnerabilidade foi explorada, expondo o sistema ao risco de perda de arquivos e vazamento de informações.

Assim, a segurança e integridade do próprio mecanismo de verificação de integridade deve ser considerada de extrema importância, e deve ser destacada durante o processo de modelagem e implementação, sob o risco de sua aplicação ser inútil e prejudicial à segurança do sistema onde está sendo implantado.

2.9 Intervalo entre Verificações

O intervalo entre duas verificações é a janela de tempo que um atacante tem para agir em um sistema monitorado por um mecanismo de verificação de integridade de arquivos, podendo assim alterar o conteúdo dos arquivos conforme seus propósitos. Como consequência a relação entre o intervalo de tempo e segurança é inversamente proporcional, ou seja, geralmente quanto menor esse intervalo, maior segurança terá o sistema.

Dessa forma, para aumentar o nível de segurança, tende-se a reduzir o intervalo de tempo entre as verificações. Essa prática coloca em risco o desempenho

do sistema, podendo prejudicar o fornecimento dos serviços que estão sendo oferecidos. O desempenho é, assim, outro fator chave, que deve ser levado em conta na modelagem e implementação de mecanismos de controle de integridade de arquivos.

Aumentando o tempo entre as verificações, a perda de desempenho torna-se aceitável. Com intervalos grandes de tempo entre duas verificações a perda de desempenho será momentânea e não causará prejuízos às operações de rotina do sistema, por outro lado a segurança é prejudicada.

A perda de desempenho e segurança do sistema monitorado não é só determinado pelo intervalo entre as verificações, mas também pela forma como o mecanismo é modelado e implementado, os momentos em que se realiza o controle de integridade e a quantidade de arquivos monitorados.

2.10 Momentos para Verificação

O momento de verificação da integridade dos arquivos monitorados é muito importante. Os momentos escolhidos influenciam diretamente no nível de segurança do sistema e em geral existem dois momentos para a realização da verificação de integridade: durante a operação normal do sistema e durante seu isolamento.

O isolamento do sistema consiste do momento em que o sistema pode ter suas atividades temporariamente suspensas em períodos regulares do tempo. Nesse caso, a vantagem consiste de que a perda de desempenho é aceitável, pois nenhum serviço ofertado e nenhuma rotina do sistema serão prejudicados, possibilitando a monitoração de uma grande quantidade de arquivos. Com o isolamento do sistema,

evita-se ainda que um atacante, que eventualmente tenha obtido acesso privilegiado, intervenha no processo de verificação de integridade.

Como o isolamento não ocorre frequentemente, esse não é o momento ideal para a verificação da integridade pois o intervalo entre as verificações tende a ser grande, fazendo com que violações demorem a ser detectadas, comprometendo o nível de segurança fornecido.

O segundo momento usa intervalos de tempo menores nas verificações, pois suas atividades não são paralisadas. Porém, a perda de desempenho é um fator limitante, pois quanto menor o intervalo, maior o gasto de recursos do sistema. Outro fator limitante é a quantidade de arquivos monitorados, já que quanto maior for o número de arquivos, maior será o custo para a verificação.

Existem dois modos de execução da verificação quando essa é feita com o sistema em operação normal:

Verificação disparada em um dado instante: A verificação é feita um um instante do tempo escolhido pelo administrador do sistema. A verificação da integridade dos arquivos é executada uma só vez, concentrando a perda de desempenho em um curto espaço do tempo;

Verificação sob demanda: A verificação é feita cada vez que o arquivo monitorado for requisitado (leitura, escrita ou execução). Somente os arquivos utilizados são verificados, o que faz com que a perda de desempenho seja diluída ao longo do tempo. Este mecanismo evita que um arquivo violado seja utilizado sem que ocorra a detecção, podendo bloquear o acesso ao arquivo.

O segundo modo tem mais vantagens que o primeiro, visto que além de reduzir o impacto no desempenho do sistema, fornece um maior nível de segurança para o mesmo.

É importante ressaltar que um mecanismo de controle de integridade de arquivos não deve se prender a somente um dos métodos citados acima. O ideal é a possibilidade de utilização de ambos, aproveitando-se das vantagens de cada um para fornecer um alto nível de segurança. A existência de pontos críticos no sistema, como comentados anteriormente não condenam a realização de verificações durante a operação normal do sistema, porém tornam necessário uma atenção maior na modelagem e implementação do mecanismo de controle de integridade. Havendo tal preocupação é possível obter um sistema equilibrado do ponto de vista de desempenho e segurança.

2.11 Considerações Finais

O processo de verificação de integridade de arquivos contém uma série de pontos sensíveis, que se não forem tratados adequadamente podem resultar em uma série de vulnerabilidades que um invasor pode usar para inverter o papel do processo, usando-o ao seu favor.

Neste capítulo foram apresentados os principais aspectos relacionados ao controle de integridade de arquivos, bem como as principais tarefas a serem executadas para o correto tratamento dos pontos sensíveis. No próximo capítulo será apresentado as ferramentas estudadas e a metodologia usada neste trabalho.

3 Materiais e Métodos

3.1 Considerações Iniciais

Inúmeras vulnerabilidades são encontradas constantemente, o que exige um esforço maior por parte do administrador do sistema para manter o sistema atualizado, corrigindo tais vulnerabilidades. A tarefa de manter o sistema atualizado dificulta e desestimula uma possível invasão, porém, isso não basta para prevenila. Além de implementar uma política de segurança adequada baseada em serviços criptográficos, configurar serviços de firewall, entre outros, o administrador deve ficar atento às tentativas de intrusão e averiguar se o sistema está integro, utilizando ferramentas de detecção de intrusos, os IDS (*Intrusion Detection System*).

Segundo Debar (2000), a tarefa dos sistemas de detecção de intrusos é monitorar o sistema computacional a procura de qualquer anomalia que leve a um estado inseguro. Isso inclui monitorar situações onde pessoas externas tentam obter acesso não autorizado e explorar vulnerabilidades, até situações onde usuários legítimos estão envolvidos. Em Nakamura e Geus (2002), é proposta uma classificação das ferramentas IDS nas categorias a seguir:

- HIDS (*Host-Based Intrusion Detection System*): ferramentas incluídas nessa classe monitoram a integridade do sistema utilizando informações locais, tais como *logs* ou agentes de auditoria e podem ser definidas como verificadores de integridade de arquivos.

- NIDS (*Network-Based Intrusion Detection System*): São ferramentas capazes de detectar uma possível intrusão em tempo real, monitorando as conexões de rede.

- Hybrid IDS: São ferramentas que possuem as mesmas funcionalidades de HIDS e NIDS.

As ferramentas de verificação de integridade de arquivos utilizadas neste trabalho pertencem à classe HIDS e são listadas a seguir:

- Tripwire versão 2.4.2.1 (http://www.tripwire.org/ e http://www.tripwire.com/);
- OSSEC versão 2.6 (http://www.ossec.net/);
- AIDE versão 0.15.1 (http://aide.sourceforge.net/);
- Samhain versão 2.6.2 (http://www.la-samhna.de/samhain/);

As quatro ferramentas citadas, foram selecionadas por serem muito difundidas e utilizadas entre administradores de rede para verificar a integridade de arquivos, além disso, são ferramentas que já estão em desenvolvimento a algum tempo, são ferramentas livres e de código aberto. Cabe ressaltar que todas as ferramentas serão instaladas e avaliadas no sistema operacional Debian GNU/Linux[1] versão 6.

3.2 A Ferramenta Tripwire

Tripwire é uma ferramenta de controle de integridade de arquivos que foi criada em 1992 por Dr. Eugene Spafford e Gene Kim na Perdue University[2]. Seu projeto e sua implementação inicial é descrita em (KIM; SPAFFORD, 1994). Existem três versões da ferramenta Tripwire, "Tripwire Open Source",indicado para um pequeno número de servidores, "Tripwire for Servers", versão comercial recomendada para organizações que requerem monitoramento de servidor único e

[1] http://www.debian.org/
[2] http://www.purdue.edu/

"Tripwire Enterprise", versão comercial mais completa que a primeira, recomendada para organizações que necessitam de auditoria de segurança. Este trabalho focará na versão *Open Source* versão 2.4.2.1.

O processo de instalação é bem simples, bastando seguir alguns passos para conclui-la. Serão pedidas duas senhas, a primeira é a senha do "*site*", utilizada para assinar digitalmente diversos arquivos do Tripwire, como as bases de dados, arquivo de configuração e arquivo de regras. Essa senha é também utilizada pelo aplicativo "twadmin", responsável por tarefas administrativas relacionadas aos arquivos de configuração e das chaves geradas. A segunda senha é a senha "local" exigida pelo Tripwire durante a criação da base de dados e da verificação de integridade de arquivos. Ao final da instalação será exibido os locais onde foram criados os arquivos de configuração e as chaves.

A Figura 1 é um exemplo de configuração das regras de quais arquivos e diretórios o Tripwire irá monitorar. Dentro dos parêntesis "(" e ")" são definidas algumas diretivas, tais como: a severidade da regra, que neste caso não foi colocada, e o nome da regra, composto pela palavra reservada "*rulename*", do carácter "=" e do nome da regra entre aspas. Após definir tais diretivas, vem a regra propriamente dita, que deve ser colocada entre chaves ("{" e "}"). No exemplo da Figura 1 a regra chamada "Regra tripwire" define que o Tripwire irá monitorar o diretório "/testeTripwire", checando as propriedades: "M", "a", "u", "g", "m", "p", "s" e ignorando a propriedade "l".

```
# Configuração da pasta \testeTripwire
(
    rulename = "Regra tripwire"
)
{
    /testeTripwire  -> +Maugmps-l;
}
```

Figura 1: Parte do arquivo de configuração do Tripwire

3.3 A Ferramenta OSSEC

A função do OSSEC é auxiliar o administrador a determinar se um invasor modificou ou não os arquivos de um sistema. É uma solução escalável, multi-plataforma e *Open Source*. Tem um forte motor de análise, integrando análise de *log*, checagem de integridade de arquivos, detecção de *rootkits*, alertas em tempo real e resposta ativa (MICRO, 2010).

É uma ferramenta livre, sendo que qualquer pessoa pode modificar e/ou distribuir sob os termos da versão 3 do GNU(*General Public License*), licença pública geral. É um *software* amplamente utilizado por universidades, governos e até grandes centros de dados, com média de 5000 *downloads* por mês. Possui suporte comercial feito pela empresa Trend Micro, empresa que está por trás do projeto.

Existem três tipos de instalação para a ferramenta OSSEC: instalação local, em um único *host*, instalação do agente, instalada em um *host* que esteja conectado em rede, e instalação do servidor, que coleta os relatórios enviados pelos *hosts* agentes. Neste trabalho será avaliada a ferramenta instalada localmente. Para mais informações sobre quais requisitos e dependências cada sistema operacional exige para instalar o OSSEC, pode-se consultar (CID; HAY; BRAY, 2008).

A Figura 2 mostra um exemplo de parte do arquivo de configuração do OSSEC, mais precisamente a parte de configuração do *Syscheck*. As *tags* <frequency> e </frequency> definem o intervalo entre as verificações, que no exemplo está como sendo 79200 segundos, ou seja, 22 horas. As *tags* <directories> e </directories> são as que definem quais diretórios serão monitorados, no exemplo um diretório monitorado é o /etc. Esta opção está configurada para realizar todas as verificações de integridade possíveis nos diretórios marcados, sentando a opção *check_all="yes"*. Entre as *tags* <ignore> e </ignore> estão os arquivos e diretórios que serão ignorados pela verificação de integridade. O primeiro conjunto de arquivos é do sistema Linux, e o segundo do sistema Windows.

```
<syscheck>
  <!-- Frequency that syscheck is executed - default to every 22 hours -->
  <frequency>79200</frequency>

  <!-- Directories to check  (perform all possible verifications) -->
  <directories check_all="yes">/etc,/usr/bin,/usr/sbin</directories>
  <directories check_all="yes">/bin,/sbin</directories>

  <!-- Files/directories to ignore -->
  <ignore>/etc/mtab</ignore>
  <ignore>/etc/mnttab</ignore>
  <ignore>/etc/hosts.deny</ignore>
  <ignore>/etc/mail/statistics</ignore>
  <ignore>/etc/random-seed</ignore>
  <ignore>/etc/adjtime</ignore>
  <ignore>/etc/httpd/logs</ignore>
  <ignore>/etc/utmpx</ignore>
  <ignore>/etc/wtmpx</ignore>
  <ignore>/etc/cups/certs</ignore>
  <ignore>/etc/dumpdates</ignore>
  <ignore>/etc/svc/volatile</ignore>

  <!-- Windows files to ignore -->
  <ignore>C:\WINDOWS/System32/LogFiles</ignore>
  <ignore>C:\WINDOWS/Debug</ignore>
  <ignore>C:\WINDOWS/WindowsUpdate.log</ignore>
  <ignore>C:\WINDOWS/iis6.log</ignore>
  <ignore>C:\WINDOWS/system32/wbem/Logs</ignore>
  <ignore>C:\WINDOWS/system32/wbem/Repository</ignore>
  <ignore>C:\WINDOWS/Prefetch</ignore>
  <ignore>C:\WINDOWS/PCHEALTH/HELPCTR/DataColl</ignore>
  <ignore>C:\WINDOWS/SoftwareDistribution</ignore>
  <ignore>C:\WINDOWS/Temp</ignore>
  <ignore>C:\WINDOWS/system32/config</ignore>
  <ignore>C:\WINDOWS/system32/spool</ignore>
  <ignore>C:\WINDOWS/system32/CatRoot</ignore>
</syscheck>
```

Figura 2: Parte do arquivo de configuração do OSSEC

3.4 A Ferramenta Samhain

A ferramenta Samhain é um software *Open Source*. Possui, além da verificação de integridade de arquivos, emissão e análise de arquivos de *log*, detecção de *rootkits*, monitoramento de portas e detecção de processos ocultos, além de verificação de integridade do *kernel*. Embora possa ser usado em um único *host*, foi projetado para monitorar múltiplos *hosts* com sistemas operacionais diferentes, fornecendo registro de *logs* centralizado (LABS, 2006).

Segundo Wotring, Potter e Ranum (2005), por ter tais características, o Samhain pode ser considerado como um sistema de monitoramento de integridade e não simplesmente um verificador de integridade de arquivos.

O desenvolvimento do Samhain é baseado no conjunto de normas POSIX (*Portable Operating System Interface*). Por esse motivo, a ferramenta é limitada a ambientes baseados em UNIX, não havendo versão para a plataforma Windows. Embora possa ser executado no Windows através do aplicativo Cygwin[3], cuja função é portar aplicativos que executam em sistemas POSIX para a plataforma Windows, não consegue lidar com questões específicas do Windows (registro, usuários, grupos, serviços, etc).

A Figura 3 mostra um exemplo do arquivo de configuração do Samhain. A marcação [Misc] indica que começa uma nova seção no arquivo, onde se pode fazer as configurações. Há duas dessas marcações, a primeira é para a seção da definição das políticas e a segunda para a seção de configuração de alguns parâmetros da verificação de integridade. No exemplo o diretório "/pasta1" será monitorado de acordo com as regras da política "Attributes", o diretório "/etc/pasta2" pela po-

[3] http://www.cygwin.com/

lítica “LogFiles” e a “/pasta5” pela política “ReadOnly”. As demais políticas não serão usadas, portanto não há nenhum diretório contido ali.

O parâmetro “SetFileCheckTime” define o intervalo em segundos entre duas verificações. No exemplo está definido que este intervalo será de 7200 segundos, ou seja, 2 horas. Por fim, a marcação “[EOF]” marca o fim do arquivo de regras.

```
[Misc]

[Attributes]
file=/pasta1

[LogFiles]
file=/etc/pasta2
[GrowingLogFiles]
[IgnoreAll]
[IgnoreNone]
[ReadOnly]
file=/pasta5

[Misc]
SetFileCheckTime=7200

[EOF]
```

Figura 3: Parte do arquivo de configuração do Samhain

3.5 A Ferramenta AIDE

A ferramenta AIDE (*Advanced Intrusion Detection Environment*), ambiente avançado de detecção de intrusão, é um verificador de integridade de arquivos. É uma ferramenta *Open Source*, que possui licença GPL. Foi desenvolvida inicialmente por Rami Lehti e Pablo Virolainen em 1999, porém várias pessoas assumiram o projeto, entre elas Richard van den Berg e Hannes von Haugwitz[4].

A ferramenta AIDE, constrói uma base de dados a partir do arquivo de configuração armazenando vários atributos dos arquivos monitorados, entre eles: per-

[4]http://aide.sourceforge.net/

missões, número do *inode*, usuário, grupo, tamanho do arquivo, mtime e ctime, atime, número de links e nome dos links. Além disso cria o *checksum* dos arquivos com um dos seguintes algoritmos de *hash*: SHA1, SHA256, SHA512, MD5, rmd160 ou tiger (LEHTI *et al.*, 2011).

O desenvolvimento prevê a implementação de suporte à UTF-8, Threads, criptografia da base de dados e criptografia do arquivo de configuração.

A Figura 4 mostra um exemplo de parte do arquivo de configuração da ferramenta AIDE. Na primeira linha uma regra é criada com alguns parâmetros, atribuindo-se o nome "MinhaRegra". Na próxima linha, o diretório /etc é adicionado para ser monitorado seguindo as regras que estão à frente. Os diretórios /sbin e /var são adicionados seguindo a regra que foi definida anteriormente, chamada de "MinhaRegra". Por último a regra "!/var/log/" diz que o diretório "/var/log/" será ignorado da verificação de integridade.

```
MinhaRegra = p+i+n+u+g+s+b+m+c+md5+sha1


/etc p+i+u+g
/sbin MinhaRegra
/var MinhaRegra
!/var/log/
```

Figura 4: Parte do arquivo de configuração da ferramenta AIDE

3.6 Critérios de Comparação

A metodologia usada no presente trabalho segue a metodologia proposta por Domingues (2003) e Labs (2006). As quatro ferramentas serão avaliadas qualitativamente, levando em conta suas funcionalidades, modo de uso das ferramentas

e alguns critérios como: tipos de pacotes de instalação disponíveis, licença de uso das ferramentas.

Os seguintes critérios, propostos e utilizados por Domingues (2003), também serão usados neste trabalho:

Uso avançado do arquivo de configuração de regras: o uso avançado do arquivo de configuração de regras facilita o uso da ferramenta, e adiciona mais segurança ao sistema, como por exemplo a configuração do envio de *email*.

Licença de uso das ferramentas: dependendo da licença da ferramenta, a mesma não pode ser usada para fins comerciais.

Pacotes disponíveis para instalação: o tipo de pacote de instalação define em quais sistemas operacionais a ferramenta trabalha.

Documentação disponível: A documentação é importante para configurar a ferramenta corretamente.

Integração com base de dados: a integração com banco de dados pode facilitar a auditoria dos *logs* emitidos pela ferramenta.

Formato do relatório de verificação: dependendo do formato do relatório de verificação, o mesmo pode ser visualizado com mais facilidade com determinadas ferramentas.

Senhas para autenticação: a necessidade de senha para utilizar a ferramenta adiciona mais um ponto de segurança no sistema, uma vez que somente o administrador poderá utilizar a ferramenta para fazer a checagem do sistema e atualizar a base de dados, por exemplo.

Modo de funcionamento das ferramentas: dependendo de como a ferramenta trabalha, a administração do sistema fica mais fácil.

Informações para verificação de integridade: os parâmetros a serem checados em cada arquivo, pode influenciar no resultado final da checagem. Quanto mais parâmetros a ferramenta pode checar, mais genérica fica, uma vez que pode ser utilizada em diversos cenários.

Ferramentas auxiliares: a administração do sistema pode se tornar mais fácil se a ferramenta possuir ferramentas auxiliares, como visualizador dos relatórios de forma mais clara.

Além dos critérios citados anteriormente, a fim de melhorar a comparação das ferramentas, serão utilizados os seguintes critérios, propostos por Labs (2006).

Data de lançamento da última versão estável: a data de lançamento da última versão estável, mostra se a ferramenta está em constante atualização.

Gerenciamento centralizado (cliente/servidor): com gerenciamento centralizado, a manutenção do sistema de verificação de integridade de arquivo torna-se mais fácil, uma vez que os clientes fazem a verificação de integridade e os relatórios ficam centralizados no servidor.

Detecção de *rootkit*: como descrito na Seção 2, *rootkit* é um aplicativo com o objetivo de garantir poderes de superusuário ao invasor. A detecção de tais aplicativos adiciona segurança ao sistema monitorado.

Assinatura PGP: a assinatura PGP é um ponto de segurança importante para os aplicativos que têm seu código-fonte distribuído, caso das ferramentas estudadas. Um atacante pode embutir código malicioso no código original do

aplicativo. Com a assinatura PGP distribuída, pode-se verificar a integridade do código-fonte distribuído.

Base de dados (*snapshot*) criptografada: com o *snapshot* do sistema criptografado, mais um ponto de segurança é adicionado no sistema, uma vez que um atacante pode tomar posse de tal arquivo a fim de descobrir quais arquivos e diretórios são monitorados.

O arquivo de regras criptografado: idem ao item anterior, um atacante pode ter acesso ao arquivo de regras a fim de descobrir quais arquivos e diretórios são monitorados.

Checar arquivos de usuários inexistentes (sem entrada UID em "/etc/passwd"): se a ferramenta não conseguir checar arquivos de usuários inexistentes, um invasor pode aproveitar tais arquivos para inserir código malicioso a fim de danificar o sistema.

Checar arquivos de grupos inexistentes (sem entrada GID em "/etc/group"): idem ao item anterior.

Checar o diretório de arquivos do *kernel* "/proc": o diretório "/proc" é o diretório é responsável por armazenar informações sobre o *hardware* e configurações do sistema. Um invasor poderia alterar as configurações do sistema, danificando assim o mesmo.

Checar o diretório "/dev" para verificar os arquivos dos dispositivos: o diretório "/dev" contém ponteiros para os dispositivos de *hardware*. Com uma brecha de segurança, um atacante poderia criar uma nova entrada para o disco com poderes de escrita, adquirindo assim poderes de *root*.

Checar o diretório raiz "/": leva em conta se o sistema pode checar somente o diretório raiz do sistema, sem entrar em recursão, ou seja, verificar tal diretório significa que o sistema não detectaria um arquivo modificado em "/etc". Este recurso é importante afim de não escanear todo o sistema, otimizando assim o processo.

Identificação de arquivos criados e/ou deletados: um sistema de verificação de integridade de arquivos que não atender a esse requisito, está invalidando todo o processo, pois se trata de uma falha de segurança gravíssima, vindo de uma ferramenta que se propôs a acusar modificação em arquivos e diretórios.

3.7 Estudo de Caso

Para testar a eficiência das ferramentas foi simulado uma situação de invasão do sistema. As alterações foram feitas manualmente, inserindo e alterando os arquivos pelo próprio sistema operacional. As seguintes modificações foram feitas:

- Inserção de um arquivo "ssftp" (desenvolvido pelo autor) no diretório "/usr/-bin";
- Inserção de um arquivo "ssftp.conf" (desenvolvido pelo autor) no diretório "/etc";
- Substituição do arquivo "useradd" localizado no diretório "/usr/sbin" por um arquivo "useradd" desenvolvido pelo autor;
- Remoção do arquivo "ps" localizado no diretório "/bin";
- Inserção de um usuário comum;
- Inserção de um usuário com permissões de *root*;

- Modificação de um arquivo que não pertence a nenhum usuário e a nenhum grupo. Para isso será criado um arquivo com o nome “teste.txt” em “/etc” pertencente a um usuário e grupo inexistente.
- Modificação dos seguintes atributos do arquivo “/etc/sudoers”: permissão, dono, grupo e tamanho.

Foram monitorados os seguintes diretórios:

- /bin, onde são armazenados os executáveis de alguns comandos básicos do sistema, como o su, tar, cat, rm e pwd;
- /sbin, onde são armazenados os arquivos binários de aplicativos que podem ser usados apenas pelo *root*;
- /etc, onde são armazenados os arquivos de configuração do sistema;
- /usr/bin, onde estão os arquivos executáveis de aplicativos instalados.
- /usr/sbin, semelhante ao diretório /sbin, este também reservado para executáveis que podem ser usados apenas pelo *root*.

Para cada arquivo contido nos diretórios, serão geradas as seguintes informações: *checksum* MD5, uid, gid, permissões e tamanho do arquivo. Os testes para comparação foram feitos na distribuição Debian GNU/Linux versão 6, em um computador AMD Athlon(tm) X2 Dual-Core QL-65 2.0GHz com memória DDR2 de 2GB.

4 Análise e Avaliação das Ferramentas de Verificação de Integridade de Arquivos

4.1 Considerações Iniciais

Existem muitas ferramentas de controle e verificação de integridade de arquivos disponíveis. Muitas tem versões *Open Source* e são totalmente gratuitas. Além disso cada ferramenta possui características diferentes.

Essa grande quantidade de ferramentas disponíveis acaba levando dificuldades aos administradores de sistemas na escolha da ferramenta mais adequada para o ambiente administrado. Essa dificuldade se deve ao fato de que não existem muitas informações comparativas entre as ferramentas disponíveis, e as que existem já não atendem às ferramentas atuais.

Diante de tal contexto, esse capítulo tem o objetivo de apresentar um estudo de caso e uma análise comparativa das quatro ferramentas apresentas no Capítulo 3.

4.2 Análise das características das ferramentas AIDE, OSSEC, Tripwire e Samhain

4.2.1 Uso avançado do arquivo de configuração de regras

Das quatro ferramentas comparadas, a ferramenta AIDE é a que possui o menor número de configurações avançadas possíveis, sendo permitidos o uso de expressão regular para definir as regras de monitoramento.

A ferramenta Tripwire permite configurações de envio de *e-mail*, definição de variáveis, especificação de diretivas e configuração do nível de recursão da análise dos arquivos e diretórios.

A ferramenta OSSEC possui permite configurações de envio de *e-mail*, nível de detalhes do relatório, direcionar o relatório para banco de dados, conexão remota, monitoramento de arquivos de *log* de ferramentas externas, como o Apache, configurações relativas à detecção de *rootkits* e opções de resposta ativa[5].

Já a ferramenta Samhain possui as seguintes configurações avançadas: configuração de monitoramento de arquivos de *log* de ferramentas externas, como Apache, nível de detalhe do relatório, customização de mensagens de erro, configurações relativas à envio de *email*, possibilidade de executar programas externos, configuração de direcionamento do relatório para banco de dados, configurações de *syslog*[6], além de configuração do nível de recursão da análise.

4.2.2 Licença de uso das ferramentas

As quatro ferramentas são licenciadas pela GNU *General Public License* (GPL)[7]. A ferramenta AIDE é licenciada pela GPL versão 1, OSSEC pela GPL versão 2 e a ferramenta Samhain pela GPL versão 3. A ferramenta Tripwire além da licença GPL versão 1, possui uma licença própria para as versões comerciais, disponível em http://www.tripwire.com/legal/eula/.

[5] http://www.symantec.com/pt/br/security_response/glossary/define.jsp?letter=a&word=active-response

[6] http://br.norton.com/security_response/glossary/define.jsp?letter=s&word=syslog

[7] Para mais informações sobre a licença acesse http://www.fsf.org/

4.2.3 Pacotes disponíveis para instalação

Todas as ferramentas possuem pacotes disponíveis nos repositórios oficiais de cada distribuição Linux. Segue abaixo os pacotes disponíveis na página oficial de cada ferramenta:

- Tripwire: pacotes ".tar.bz2" e ".rpm" disponível em: http://sourceforge.net/projects/tripwire/files/;
- AIDE: pacote ".tar.gz" disponível em: http://sourceforge.net/projects/aide/files/;
- OSSEC: pacotes ".tar.gz, ".rpm" e uma versão para Windows com extensão ".exe" disponível em: http://www.ossec.net/main/downloads;
- Samhain: pacote ".tar.gz" disponível em: http://www.la-samhna.de/samhain/s_download.html.

4.2.4 Documentação disponível

Todas as ferramentas possuem documentação na língua inglesa em suas páginas oficiais. A última documentação da ferramenta Tripwire é datada do ano 2000 e é referente à versão 2.3 da ferramenta, porém não foram feitas muitas modificações na versão 2.4.2.1 da ferramenta, logo a documentação é suficiente para a última versão, e está disponível em http://sourceforge.net/projects/tripwire/files/tripwire-src/2.3.0-docs-pdf/.

A ferramenta AIDE possui uma documentação pequena e com poucos exemplos de uso, disponível em http://aide.sourceforge.net/ stable/manual.html#about.

A ferramenta Samhain possui ótima documentação disponível em sua página oficial, abordando muitos tópicos e com muitos exemplos de uso, disponível em http://www.la-samhna.de/samhain/manual/, além de também ser encontrada em (WOTRING; POTTER; RANUM, 2005).

Já a ferramenta OSSEC possui uma documentação grande e com muitos exemplos de uso, porém possui poucos detalhes técnicos, como por exemplo a ausência de documentação sobre o procedimento necessário para realizar o *update* da base de dados gerada pela ferramenta. Além disso, não existe documentação explicando alguns módulos implementados na ferramenta. Está disponível em http://www.la-samhna.de/samhain/manual/, além de também ser encontrada em (WOTRING; POTTER; RANUM, 2005).

4.2.5 Integração com base de dados

As ferramentas AIDE e Tripwire não possuem recurso de integração com banco de dados para armazenar os relatórios. Já as ferramentas Samhain e OSSEC possuem tal recurso. Samhain pode realizar integração com as base de dados MySQL, PostgreSQL, ORACLE e Unix ODBC. A ferramenta OSSEC possui integração com as base de dados MySQL e PostgreSQL.

4.2.6 Formato do Relatório de Verificação

De todas as ferramentas, somente Tripwire possui o relatório em formato binário além de direcionar um relatório simplificado para a saída padrão (*stdout*). A ferramenta AIDE direciona o relatório para a saída padrão. As ferramentas

Samhain e OSSEC possuem o relatório em formato de arquivo de texto simples, podendo ser visualizado por qualquer editor de texto.

4.2.7 Senhas para autenticação

Somente a ferramenta Tripwire pede senha ao usuário para realizar as operações. A senha é pedida no momentos de gerar e atualizar o *snapshot* dos arquivos e diretórios monitorados.

4.2.8 Modo de Funcionamento das Ferramentas

O modo de funcionamento das ferramentas pode tornar mais fácil a tarefa de administrar o sistema. Se a ferramenta possuir a característica de fazer as verificações de integridade automaticamente, o administrador não tem a preocupação de realizar a verificação manualmente, via comando. Uma alternativa ao administrador para ferramentas que não possuem tal característica é criar um *script* e colocá-lo no *cron* do sistema, assim a cada período de tempo o *script* é acionado e a verificação de integridade é executada.

As ferramentas Tripwire e AIDE não fazem a verificação de forma automática. O administrador precisa digitar o comando manualmente no console para realizar a verificação de integriadade. As ferramentas OSSEC e Samhain possuem parâmetros de configuração para realizar a verificação de integridade de arquivo em períodos de tempo.

4.2.9 Informações para verificação de integridade

A Tabela 1 mostra os parâmetros que cada ferramenta pode checar ao fazer a análise de verificação de integridade de um arquivo ou diretório monitorado.

Tabela 1: Informações para verificação de integridade

Parâmetro	**Ferramentas**			
	Tripwire	AIDE	OSSEC	Samhain
Permissões do arquivo	Sim	Sim	Sim	Sim
Número do *inode*	Sim	Sim	Não	Sim
Número de *links*	Sim	Sim	Não	Sim
uid do arquivo	Sim	Sim	Sim	Sim
gid do arquivo	Sim	Sim	Sim	Sim
Tamanho do arquivo em bytes	Sim	Sim	Não	Sim
Tamanho do arquivo em blocos	Sim	Sim	Não	Não
Data e hora da última modificação	Sim	Sim	Não	Sim
Data e hora do último acesso	Sim	Sim	Não	Não
Data e hora da criação do arquivo	Não	Sim	Não	Não
Data e hora de criação do *inode*	Sim	Não	Não	Não
Mudança no tamanho do arquivo	Sim	Sim	Sim	Sim
Checksum MD5 do arquivo	Sim	Sim	Sim	Não
Checksum SHA-1 do arquivo	Sim	Sim	Sim	Não
Checksum SHA-256 do arquivo	Não	Sim	Não	Não
Checksum SHA-512 do arquivo	Não	Sim	Não	Não

Checksum RIPEMD-160 do arquivo	Não	Sim	Não	Não
Checksum Tiger do arquivo	Não	Sim	Não	Sim
Checksum CRC-32 do arquivo	Sim	Sim	Não	Não
Checksum HAVAL do arquivo	Sim	Sim	Não	Não
Grupo vazio	Não	Sim	Não	Não
ID do dispositivo apontado pelo *inode*	Sim	Não	Não	Não
Tipo de arquivo	Sim	Sim	Não	Sim

4.2.10 Ferramentas auxiliares

De todas as ferramentas, somente a ferramenta OSSEC possui uma ferramenta auxiliar. Trata-se de uma interface *Web* para visualização dos relatórios de forma simples e intuitiva, disponível em: http://www.ossec.net/files/ui/ossec-wui-0.3.tar.gz.

4.2.11 Características e funcionalidades presentes

A Tabela 2 mostra quais funcionalidades e características cada ferramenta possui.

Tabela 2: Características e Funcionalidade das ferramentas

	Ferramentas			
	Tripwire	AIDE	OSSEC	Samhain
Última versão estável	21-11-2011	20-05-2011	19-07-2011	23-02-2012
Cliente/Servidor	Não	Não	Sim	Sim
Detecção de *rootkit*	Não	Não	Sim	Sim
Assinatura PGP	Não	Sim	Sim	Sim
BD criptografado	Sim	Não	Não	Sim
Regras criptografada	Sim	Não	Não	Sim
Sem UID	Sim	Sim	Sim	Sim
Sem GID	Sim	Sim	Sim	Sim
Checar "/proc"	Sim	Sim	Não	Sim
Checar "/dev"	Sim	Sim	Não	Sim
Checar "/"	Sim	Não	Não	Sim
Criar/Deletar	Sim	Sim	Não	Sim

Em relação à característica de conseguir escanear o diretório "/proc", proposto por (LABS, 2006), somente a ferramenta OSSEC não consegue gerar o *snapshot* do diretório não sendo possível identificar as modificações. Já as ferramentas Tripwire, Samhain e AIDE conseguem monitorar o diretório, porém com algumas ressalvas.

A ferramenta AIDE tem problemas com permissões, uma vez que não usa toda a permissão de *root*. Ela não consegue escanear o arquivo "/proc/sys/net/ipv4/route /flush", por exemplo, uma vez que o *root* tem permissão somente de escrita. Com isso, toda a checagem de arquivos fica comprometida. Ao repetir o teste proposto por (LABS, 2006), a ferramenta não conseguiu apontar modificações em arquivos.

Ao alterar o arquivo “/proc/sys/net/ipv4/ip_forward” a ferramenta não identifica a alteração, porém identifica a criação do diretório “/proc/q008/fd/38”.

Já as ferramentas Samhain e Tripwire possuem problemas de leitura ao escanear o diretório “/proc” recursivamente, não gerando o *snapshot* do diretório. Porém as duas ferramentas conseguem escanear o *inode* do diretório.

Ao configurar as ferramentas para monitorar o diretório “/dev”, somente a ferramenta OSSEC não consegue monitorá-lo. Isso se deve ao fato de não conseguir gerar o *snapshot* do diretório. As demais ferramentas conseguem monitorar e apontar as modificações feitas.

A característica de escanear o diretório raiz (“/”) não recursivamente, está presente somente nas ferramentas Tripwire e Samhain, uma vez que têm a possibilidade de configurar o nível de recursão da checagem dos diretórios. As ferramentas AIDE e OSSEC não possuem opções em seus arquivos de configuração para monitorar o diretório raiz sem recursão, portanto se configurar tais ferramentas para escanear esse diretório, o sistema inteiro será verificado.

4.3 Estudo de caso

O objetivo do estudo de caso, utilizando a metodologia descrita na Seção 3, é avaliar as ferramentas quanto à sua eficiência perante às simulações de invasão, bem como obter o tempo que cada ferramenta leva para gerar a base de dados e o tempo gasto para analisar e reportar as modificações feitas.

Os arquivos de configuração de cada ferramenta estudada, que descrevem o cenário apresentado na Seção 3 são apresentados no Apêndice A.1 (Tripwire), Apêndice A.2 (AIDE), Apêndice A.3 (OSSEC) e Apêndice A.4 (Samhain).

A Tabela 3 mostra a média de tempo e o desvio padrão (entre parêntesis) de cinco execuções, gasto por cada ferramenta para gerar a base de dados para as futuras comparações e o tempo gasto para a ferramenta realizar a checagem do sistema e reportar as modificações feitas. As ferramentas foram executadas após a simulação de invasão descrita na Seção 3. O relatório gerado por cada ferramenta após a verificação de integridade podem ser visualizados no Apêndice B.1 (Tripwire), Apêndice B.2 (AIDE), Apêndice B.3 (OSSEC) e Apêndice B.4 (Samhain).

Tabela 3: Análise de tempo em segundos das ferramentas estudadas

Critério	**Ferramentas**			
	Tripwire	AIDE	OSSEC	Samhain
Geração da base de dados	**5.6** (0.28)	**2.5** (0.31)	**437** (2.06)	**10.6** (0.63)
Verificação dos arquivos	**2.14** (0.26)	**1.1** (0.21)	**463** (1.9)	**7.4** (1.02)

Analisando a Tabela 3 pode-se notar que a ferramenta mais rápida é a ferramenta AIDE com média de 2.5 segundos para gerar a base de dados e 1.1 segundo para realizar a checagem dos arquivos. Já a ferramenta OSSEC é a ferramenta mais lenta na execução das duas tarefas, com um tempo muito discrepante comparado às outras ferramentas, com média de 437 segundos (7 minutos e 19 segundos) para gerar a base de dados e 463 segundos (7 minutos e 43 segundos) para realizar a verificação de integridade.

A análise dos relatórios apresentados no Apêndice B mostra que todas as ferramentas foram capazes de identificar os dois usuários adicionados, identificando modificações nos seguintes arquivos: “/etc/group”, “/etc/group-”, “/etc/passwd”, “/etc/passwd-”, “/etc/shadow”, “/etc/shadow-”, “/etc/gshadow-” e “/etc/gshadow”. Além disso, todas as ferramentas conseguem identificar as modificações em todos

os atributos monitorados do arquivo “/etc/sudoers”. O arquivo que não pertence a nenhum grupo e a nenhum usuário também foi monitorado por todas as ferramentas com sucesso.

Já em relação aos demais arquivos adicionados, alterados e removidos, as ferramentas Tripwire, AIDE e Samhain reportaram todas as alterações. A ferramenta OSSEC não conseguiu identificar o arquivo removido “/bin/ps”, e mesmo com a configuração “<alert_new_files>yes</alert_new_files>” para alertar novos arquivos criados, não consegue identificar os arquivos adicionados “/usr/bin/ssftp” e “/etc/ssftp.conf”. Isso só pode ser feito adicionando a seguinte regra no arquivo de regras “/var/ossec/rules/local_rules.xml”:

```
<rule id=''554'' level=''0''>
<category>ossec</category>
<decoded_as>syscheck_new_entry</decoded_as>
<description>File added to the system.</description>
<group>syscheck,</group>
</rule>
```

Figura 5: Regra a ser adicionada para a ferramenta OSSEC conseguir identificar arquivos adicionados.

Esta é uma falha gravíssima da ferramenta OSSEC, que não reporta este problema nem a solução na documentação oficial. Essa falha abre uma brecha na segurança do sistema monitorado e ilude o administrador do sistema, pois o mesmo supõe que o sistema está protegido, pois seguiu as orientações da documentação oficial para configurar a ferramenta.

4.4 Avaliação e comparação das ferramentas

As informações apresentadas, permitem dizer que as ferramentas podem ser divididas em dois grupos. O primeiro grupo, composto pelas ferramentas OSSEC e Samhain, pode ser definido como ferramentas mais voltadas para o uso em ambientes com grandes conjuntos de computadores conectados em rede, onde não seja possível fazer a verificação de integridade em cada *host* separadamente. Tais ferramentas possuem arquitetura cliente/servidor, o que torna mais fácil administrar muitos computadores, uma vez que cada cliente faz a verificação de integridade e envia os dados para o servidor. Elas possuem a característica de fazer a checagem do sistema automaticamente de tempos em tempos, não necessitando a criação de *scripts* em cada *host* para executar a ferramenta. Além disso as ferramentas possuem recursos de auditoria, como análise de *log* de ferramentas externas, como o Apache e detecção de *rootkit*.

As ferramentas do segundo grupo, Tripwire e AIDE, são mais voltadas para a administração de um único computador, como um servidor de aplicações por exemplo. Não possuem arquitetura cliente/servidor[8] e a verificação de integridade deve ser feita manualmente. Além disso como mostrado na Figura 6 podem verificar um conjunto maior de informações do arquivo monitorado.

[8] A ferramenta Tripwire possui uma versão com arquitetura cliente/servidor, porém é uma versão comercial, diferente da analisada neste trabalho.

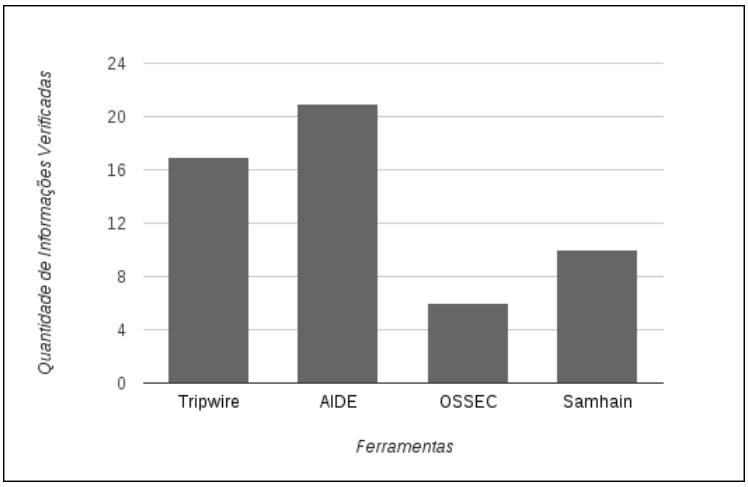


Figura 6: Comparação da quantidade de informações verificadas pelas ferramentas

Todas as ferramentas estão em constante desenvolvimento, com atualização e implementação de novas funções. Isso pode ser notado pela última versão estável lançada, descrita na Tabela 2.

Comparando as ferramentas OSSEC e Samhain, pode-se concluir que a ferramenta Samhain é a mais recomendada para o uso no cenário descrito anteriormente, pois possui todas as características consideradas importantes para uma ferramenta de verificação de integridade de arquivos, como mostrado na Tabela 2, tem a capacidade de monitorar um maior número de informações dos arquivos, como pode ser visto na Figura 6, gasta um tempo menor para gerar a base de dados e realizar a verificação de integridade em relação à ferramenta Samhain, além de ser possível definir o nível de recursão da análise, podendo assim, monitorar o diretório raiz sem recursão. Porém a ferramenta Samhain contém alguns pontos fracos, relacionados à configuração do arquivo de regras. As regras são pré definidas, limitando o sistema àquelas regras, não dando liberdade ao administrador de escolher quais atributos monitorar.

A ferramenta OSSEC possui falhas gravíssimas que podem abrir brechas no sistema para uma possível invasão e modificação do mesmo. A documentação não aborda todos os módulos implementados na ferramenta, nem as soluções dos problemas que tem, como por exemplo a necessidade de inserir uma regra no arquivo de regras para identificar aquivos inseridos, como descrito na Seção 4.3. Além disso não identifica arquivos removidos do sistema, não é capaz de monitorar os diretórios "/proc", "/dev" e "/" (não recursivamente) e não possui o arquivo de regras nem a base de dados criptografados, adicionando assim mais uma brecha no sistema. Como visto nas Figuras 6 e 7, é a ferramenta que monitora o menor número de informações nos arquivos e a que tem o menor número de características importantes em uma ferramenta de verificação de integridade de arquivos. Como mostrado na Tabela 3, leva um grande tempo para gerar a base de dados e realizar a verificação de integridade.

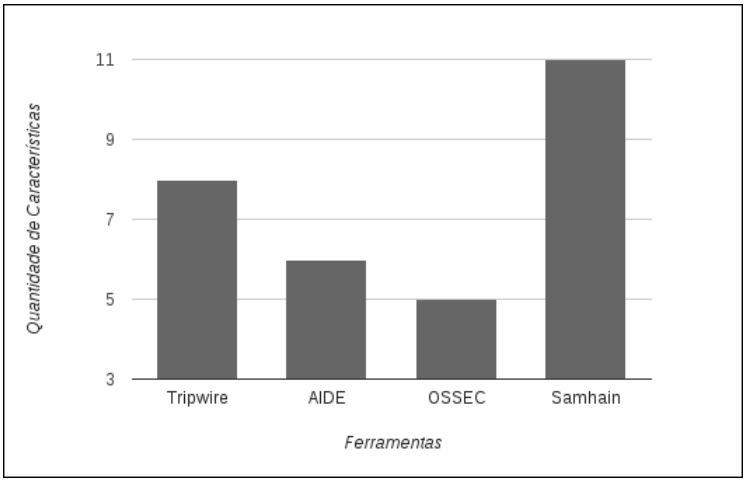


Figura 7: Comparação da quantidade de características das ferramentas

Comparando as ferramentas Tripwire e AIDE, pode-se dizer que a ferramenta Tripwire é a mais indicada para o uso no cenário descrito para essas ferramen-

tas, pois apesar de gastar um tempo maior para gerar a base de dados e realizar a checagem do sistema, como mostrado na Tabela 3, contém mais recursos de segurança do que a ferramenta AIDE. Além de possuir características importantes, como mostrado na Tabela 2, contém recurso de envio de email, arquivo de regras e base de dados criptografados e configuração do nível de recursão da análise e possui boa documentação em seu *site* oficial. Em contrapartida não possui assinatura PGP do código-fonte, falha grave que deve ser corrigida, uma vez que um atacante pode modificar o código-fonte a fim de danificar o sistema no qual será instalado a ferramenta, ou até mesmo criar *backdoors* para invasões no sistema monitorado.

A ferramenta AIDE é uma boa escolha para cenários em que o controle de integridade deve ser realizado com rapidez, pois é a ferramenta mais rápida como mostrado na Tabela 3, e em cenários que não há necessidade de um nível de segurança alto. Apesar de possuir muitos recursos de segurança, não tem alguns recursos que adicionam segurança no sistema, como arquivo de regras e base de dados criptografados. Além disso tem poucos recursos avançados, como descrito na Seção 4.2.1, a documentação é fraca e não consegue escanear o diretório raiz não recursivamente, limitando o cenário que pode ser usada. É a ferramenta que pode escanear o maior número de informações dos arquivos como mostrado na Figura 7, porém os recursos que tem a mais que a ferramenta Tripwire são funções de *hash*, como mostrado na Tabela 1.

4.5 Considerações Finais

Este capítulo teve como objetivo estudar, avaliar e comparar as ferramentas Tripwire, AIDE, OSSEC e Samhain de acordo com a metodologia descrita na Seção 3. Os critérios de comparação utilizados foram propostos por Domingues

(2003) e Labs (2006). Um estudo de caso foi feito para avaliar as ferramentas quanto a sua eficiência às modificações feitas pela simulação de invasão, além de obter o tempo gasto por cada ferramenta para gerar a base de dados do sistema e realizar a verificação de integridade. Com isso foi possível fazer uma análise das ferramentas apontando os pontos fortes e pontos fracos de cada uma, podendo assim, identificar a melhor ferramenta em cada caso.

5 Conclusão

O presente trabalho procurou estudar, analisar e comparar as ferramentas Tripwire, AIDE, OSSEC e Samhain afim de dizer qual delas é mais adequada em cada caso, ajudando assim o administrador a escolher entre as quatro ferramentas apresentadas, aquela que vai se adequar mais ao cenário que administra.

Considerando as informações apresentas neste trabalho, pode-se concluir que as ferramentas estudadas se dividem em dois grupos. Ferramentas que possuem arquitetura cliente/servidor, onde são mais indicadas para ambientes com muitos computadores, e ferramentas que não possuem tal arquitetura, e são mais adequadas para realizar o controle de integridade em um único computador, um servidor de aplicação por exemplo. Portanto as ferramentas que pertencem a um determinado grupo não são recomendas para serem usadas no cenário descrito para o outro grupo.

Das ferramentas que possuem arquitetura cliente servidor, Samhain e OSSEC, foi identificado que a ferramenta Samhain é a mais adequada para o uso no ambiente descrito. A ferramenta OSSEC possui falhas gravíssimas que abrem brechas no sistema monitorado, iludindo assim o administrador, além de gastar muito tempo para gerar a base de dados e realizar a verificação de arquivos, portanto não sendo indicada para uso. Comparando as ferramentas Tripwire e AIDE, pode-se concluir que a ferramenta Tripwire é a que mais possui recursos de segurança, sendo mais recomendada para uso em ambientes que requerem um maior nível de segurança. A ferramenta AIDE, apesar de não possuir muitos recursos de segurança é a mais adequada para ambientes em que a verificação de integridade deve ser feita de forma rápida.

Como trabalhos futuros, pretende-se avaliar e comparar configurações avançadas das ferramentas, bem como aumentar o número de ferramentas avaliadas, uma vez que tais comparações ajudam o administrador do sistema a escolher a ferramenta que melhor se adeque ao ambiente administrado. Além disso pode-se fazer um estudo separado das ferramentas que possuem as mesmas características, como por exemplo, as ferramentas que possuem arquitetura cliente/servidor, comparando como as ferramentas trabalham em rede, medindo sua eficiência e a segurança das informações transmitidas.

Referências

BACE, R.; MELL, P. *Intrusion detection systems*. [S.l.]: US Dept. of Commerce, Technology Administration, National Institute of Standards and Technology, 2001.

CID, D. B.; HAY, A.; BRAY, R. *OSSEC Host-Based intrusion Detection Guide*. [S.l.]: Elsevier, Inc., 2008.

DEBAR, H. An introduction to intrusion detection systems. *Proceedings of Connect*, Citeseer, 2000.

DOMINGUES, M. A. *Comparação de Ferramentas de Verificação de Integridade de Arquivos*. Monografia (Curso de Pós-Graduação Lato Sensu em Administração em Redes Linux) — Universidade Federal de Lavras, Lavras, 2003. Disponível em: <http://www.ginux.ufla.br/node/109>.

KACZMAREK, J.; WROBEL, M. Modern approaches to file system integrity checking. In: *Information Technology, 2008. IT 2008. 1st International Conference on*. [S.l.: s.n.], 2008. p. 1 –4.

KIM, G.; SPAFFORD, E. The design and implementation of tripwire: A file system integrity checker. In: ACM. *Proceedings of the 2nd ACM Conference on Computer and Communications Security*. [S.l.], 1994. p. 18–29.

LABS, S. *The SAMHAIN file integrity / intrusion detection system*. 2006. Disponível em: <http://www.la-samhna.de/samhain/>. Acesso em: 25 de outubro de 2011.

LEHTI, R.; VIROLAINEN, P.; BERG, R. van den; HAUGWITZ, H. von. *AIDE - Advanced Intrusion Detection Environment*. 2011. Disponível em: <http://aide.sourceforge.net/>. Acesso em: 25 de outubro de 2011.

LIMA, C. F. L. *Agentes Inteligentes para Detecção de Intrusos em Redes de Computadores*. Tese (Dissertação de Mestrado) — UFMA, São Luíz, 2002.

MICRO, I. T. *OSSEC*. 2010. Disponível em: <http://www.ossec.net/>. Acesso em: 25 de outubro de 2011.

NAKAMURA, E.; GEUS, P. de. Segurança de redes. *Berkley Brasil: São Paulo*, 2002.

PATIL, S.; KASHYAP, A.; SIVATHANU, G.; ZADOK, E. I3fs: An in-kernel integrity checker and intrusion detection file system. In: *Proceedings of the 18th Annual Large Installation System Administration Conference*. [S.l.: s.n.], 2004.

QUYNH, N.; TAKEFUJI, Y. A novel approach for a file-system integrity monitor tool of xen virtual machine. In: ACM. *Proceedings of the 2nd ACM symposium on Information, computer and communications security*. [S.l.], 2007. p. 194–202.

SERAFIM, V. da S. *Um verificador seguro de integridade de arquivos*. Tese (Dissertação de Mestrado) — UFRGS, Porto Alegre, 2002.

STALLINGS, W. *Cryptography and network security: principles and practice*. [S.l.]: Prentice Hall Press, 2010.

UCHÔA, J. Q. *Algoritmos Imunoinspirados Aplicados em Segurança Computacional: Utilização de Algoritmos Inspirados no Sistema Imune para Detecção de Intrusos em Redes de Computadores*. Tese (Tese de Doutorado) — UFMG, Belo Horizonte, 4 2009.

WANG, X.; YIN, Y.; YU, H. Finding collisions in the full sha-1. In: SPRINGER. *Advances in Cryptology–CRYPTO 2005*. [S.l.], 2005. p. 17–36.

WANG, X.; YU, H. How to break md5 and other hash functions. *Advances in Cryptology–EUROCRYPT 2005*, Springer, p. 19–35, 2005.

WOTRING, B.; POTTER, B.; RANUM, M. *Host integrity monitoring using osiris and samhain*. [S.l.]: Syngress Media Inc, 2005.

A Arquivos de configuração das ferramentas de verificação de integridade de arquivos

A.1 Arquivo de configuração da ferramenta Tripwire

```
TWBIN = /usr/sbin;
TWETC = /etc/tripwire;
TWVAR = /var/lib/tripwire;
# Monitora os arquivos contidos nos diretórios:
# /bin, /sbin, /etc, /usr/bin e /usr/sbin
(
    rulename = "Monitorar /bin, /sbin, /etc, /usr/bin e /usr/sbin"
)
{
  /bin -> +Mugps;
  /sbin -> +Mugps;
  /etc -> +Mugps;
  /usr/bin -> +Mugps;
  /usr/sbin -> +Mugps;
}
```

A.2 Arquivo de configuração da ferramenta AIDE

```
# Configurações da ferramenta relativa a base de dados
database=file:/var/lib/aide/aide.db
database_out=file:/var/lib/aide/aide.db.new
database_new=file:/var/lib/aide/aide.db.new

# Regra usada para verificacao dos arquivos
# Monitorar os seguintes atributos:
# checksum md5, usuario, grupo, permissoes e tamanho

RegraVerificacao = md5+u+g+p+s

# Monitora, utilizando a regra definida acima, os arquivos
# contidos nos diretórios: /bin, /sbin, /etc, /usr/bin e /usr/sbin

/bin RegraVerificacao
/sbin RegraVerificacao
/etc RegraVerificacao
/usr/bin RegraVerificacao
/usr/sbin RegraVerificacao
```

A.3 Arquivo de configuração da ferramenta OSSEC

```
<ossec_config>
  <global>
    <email_notification>no</email_notification>
  </global>
  <rules>
    <include>rules_config.xml</include>
    <include>pam_rules.xml</include>
    <include>sshd_rules.xml</include>
    <include>telnetd_rules.xml</include>
    <include>syslog_rules.xml</include>
    <include>arpwatch_rules.xml</include>
    <include>symantec-av_rules.xml</include>
    <include>symantec-ws_rules.xml</include>
    <include>pix_rules.xml</include>
    <include>named_rules.xml</include>
    <include>smbd_rules.xml</include>
    <include>vsftpd_rules.xml</include>
    <include>pure-ftpd_rules.xml</include>
    <include>proftpd_rules.xml</include>
    <include>ms_ftpd_rules.xml</include>
    <include>ftpd_rules.xml</include>
    <include>hordeimp_rules.xml</include>
    <include>roundcube_rules.xml</include>
    <include>wordpress_rules.xml</include>
    <include>cimserver_rules.xml</include>
    <include>vpopmail_rules.xml</include>
    <include>vmpop3d_rules.xml</include>
    <include>courier_rules.xml</include>
    <include>web_rules.xml</include>
    <include>apache_rules.xml</include>
    <include>nginx_rules.xml</include>
    <include>php_rules.xml</include>
    <include>mysql_rules.xml</include>
    <include>postgresql_rules.xml</include>
    <include>ids_rules.xml</include>
    <include>squid_rules.xml</include>
    <include>firewall_rules.xml</include>
    <include>cisco-ios_rules.xml</include>
    <include>netscreenfw_rules.xml</include>
    <include>sonicwall_rules.xml</include>
    <include>postfix_rules.xml</include>
    <include>sendmail_rules.xml</include>
    <include>imapd_rules.xml</include>
    <include>mailscanner_rules.xml</include>
    <include>dovecot_rules.xml</include>
    <include>ms-exchange_rules.xml</include>
    <include>racoon_rules.xml</include>
    <include>vpn_concentrator_rules.xml</include>
```

```
    <include>spamd_rules.xml</include>
    <include>msauth_rules.xml</include>
    <include>mcafee_av_rules.xml</include>
    <include>trend-osce_rules.xml</include>
    <include>ms-se_rules.xml</include>
    <include>zeus_rules.xml</include>
    <include>solaris_bsm_rules.xml</include>
    <include>vmware_rules.xml</include>
    <include>ms_dhcp_rules.xml</include>
    <include>asterisk_rules.xml</include>
    <include>ossec_rules.xml</include>
    <include>attack_rules.xml</include>
    <include>openbsd_rules.xml</include>
    <include>clam_av_rules.xml</include>
    <include>bro-ids_rules.xml</include>
    <include>dropbear_rules.xml</include>
    <include>local_rules.xml</include>
  </rules>
  <syscheck>
    <!-- Frequencia da verificacao - 10 min -->
    <frequency>600</frequency>
    <!-- Alertar sobre novos arquivos criados -->
    <alert_new_files>yes</alert_new_files>
    <!-- Diretorios checados -->
    <directories check_all="yes">/bin, /sbin, /etc,/usr/bin, /usr/sbin </directories>
  </syscheck>
  <alerts>
    <log_alert_level>1</log_alert_level>
  </alerts>
</ossec_config>
```

A.4 Arquivo de configuração da ferramenta Samhain

```
[Misc]

[Attributes]

[LogFiles]

[GrowingLogFiles]

[IgnoreAll]

# Checar todos os parametros possiveis
[IgnoreNone]

dir=/bin
dir=/sbin
```

```
dir=/etc
dir=/usr/bin
dir=/usr/sbin

[Prelink]

[ReadOnly]

[User0]

[User1]

[EventSeverity]

# Nivel de alerta das regras
SeverityReadOnly=crit
SeverityLogFiles=crit
SeverityGrowingLogs=warn
SeverityIgnoreNone=crit
SeverityAttributes=crit
SeverityIgnoreAll=info

# Nivel de alerta
SeverityFiles=crit
SeverityDirs=crit
SeverityNames=warn

[Log]

[Misc]
# Intervalo entre duas verificacoes
SetFileCheckTime = 600

[EOF]
```

B Relatórios de resposta das ferramentas de verificação de integridade de arquivos

B.1 Relatório da ferramenta Tripwire

```
===============================================================================
Rule Summary:
===============================================================================
-------------------------------------------------------------------------------
  Section: Unix File System
-------------------------------------------------------------------------------
```

```
  Rule Name                      Severity Level    Added    Removed  Modified
  ---------                      --------------    -----    -------  --------
* Monitorar /bin, /sbin, /etc, /usr/bin e /usr/sbin
                                 0                 2        1        11
Total objects scanned:  3089
Total violations found:  14
================================================================================
Object Detail:
================================================================================
--------------------------------------------------------------------------------
  Section: Unix File System
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Rule Name: Monitorar /bin, /sbin, /etc, /usr/bin e /usr/sbin (/bin)
Severity Level: 0
--------------------------------------------------------------------------------
  ----------------------------------------
  Removed Objects: 1
  ----------------------------------------
Removed object name:  /bin/ps

--------------------------------------------------------------------------------
Rule Name: Monitorar /bin, /sbin, /etc, /usr/bin e /usr/sbin (/etc)
Severity Level: 0
--------------------------------------------------------------------------------
  ----------------------------------------
  Added Objects: 1
  ----------------------------------------
Added object name:  /etc/ssftp.conf
  ----------------------------------------
  Modified Objects: 10
  ----------------------------------------
Modified object name:  /etc/group
  Property:            Expected                   Observed
  -------------        -----------                -----------
* Size                 676                        712
* MD5                  A26fBayx15d6p7cESoBJU6     CLh8P6MCPqeJ/tx1mVmwSV

Modified object name:  /etc/group-
  Property:            Expected                   Observed
  -------------        -----------                -----------
* Size                 693                        695
* MD5                  Cob/1+MoidNklETTFnKgZo     BCxWpVMTVYLi8MF0RlNrDy
Modified object name:  /etc/gshadow
  Property:            Expected                   Observed
  -------------        -----------                -----------
* Size                 567                        595
* MD5                  CtDAinL/dFrwnXDt1f3Fqk     CeelTuMuEIOnm/VphI0Occ
Modified object name:  /etc/gshadow-
  Property:            Expected                   Observed
  -------------        -----------                -----------
```

```
* Size                 580                       582
* MD5                  B10c2izcqVUszwGFHzkduX    BSY+tzohDmvZTzX1t4V7ef
Modified object name:  /etc/passwd
  Property:            Expected                  Observed
  -------------        -----------               -----------
* Size                 1129                      1233
* MD5                  C+bTbGZv9DV5mZSUK11qDm    C3w5NY4IROcWZtoCxzFn4f
Modified object name:  /etc/passwd-
  Property:            Expected                  Observed
  -------------        -----------               -----------
* Size                 1179                      1230
* MD5                  AYLZxVnwNAEOfDik5sng4A    Aa4iNE64tohwRCzwtmwSXB
Modified object name:  /etc/shadow
  Property:            Expected                  Observed
  -------------        -----------               -----------
* Size                 904                       1160
* MD5                  DsibUlY+OfH9rZmIlzjJHd    AHw87ebBZxyv5vYK/colyw
Modified object name:  /etc/shadow-
  Property:            Expected                  Observed
  -------------        -----------               -----------
* Size                 1031                      1033
* MD5                  DvK2Tr2WR+ah7oTdFFPsFe    BsCGiCob0exXkHvfYrjABr
Modified object name:  /etc/sudoers
  Property:            Expected                  Observed
  -------------        -----------               -----------
* Mode                 -r--r-----                -r--r--r--
* UID                  root (0)                  michel (1000)
* GID                  root (0)                  michel (1000)
* Size                 491                       514
* MD5                  DF2rDydxQR7X5n1tq2CjEf    CptMexMLRb7unDUstdmYcd
Modified object name:  /etc/teste.txt
  Property:            Expected                  Observed
  -------------        -----------               -----------
* Size                 56                        79
* MD5                  C3qe6t9E/1pgODyxoERkVp    Cem2J/bgX58qJmS7fakTG4
--------------------------------------------------------------------------------
Rule Name: Monitorar /bin, /sbin, /etc, /usr/bin e /usr/sbin (/usr/bin)
Severity Level: 0
--------------------------------------------------------------------------------
  ----------------------------------------
  Added Objects: 1
  ----------------------------------------
Added object name:  /usr/bin/ssftp
--------------------------------------------------------------------------------
Rule Name: Monitorar /bin, /sbin, /etc, /usr/bin e /usr/sbin (/usr/sbin)
Severity Level: 0
--------------------------------------------------------------------------------
  ----------------------------------------
  Modified Objects: 1
  ----------------------------------------
Modified object name:  /usr/sbin/useradd
```

```
  Property:            Expected                   Observed
  -------------        -----------                -----------
* Mode                 -r-xr--r--                 -rwxr-xr-x
* Size                 87792                      6385
* MD5                  CoVmwZb1PwH/Vpjs0rytny     BV7CS+KAjmDvyOYlhj1fQB
---------------------------------------------------------------------------
Error Report:
---------------------------------------------------------------------------
No Errors
---------------------------------------------------------------------------
*** End of report ***
```

B.2 Relatório da ferramenta AIDE

```
Summary:
  Total number of files:    3093
  Added files:          2
  Removed files:        1
  Changed files:        11
---------------------------------------------------
Added files:
---------------------------------------------------
added: /etc/ssftp.conf
added: /usr/bin/ssftp
---------------------------------------------------
Removed files:
---------------------------------------------------
removed: /bin/ps
---------------------------------------------------
Changed files:
---------------------------------------------------
changed: /etc/group
changed: /etc/group-
changed: /etc/gshadow
changed: /etc/gshadow-
changed: /etc/passwd
changed: /etc/passwd-
changed: /etc/shadow
changed: /etc/shadow-
changed: /etc/sudoers
changed: /etc/teste.txt
changed: /usr/sbin/useradd
---------------------------------------------------
Detailed information about changes:
---------------------------------------------------
File: /etc/group
 Size     : 676                           , 712
```

```
 MD5      : NunwWssdeXeqe3BEqASVOg==         , i4fD+jAj6nif7cdZlZsElQ==
File: /etc/group-
 Size     : 693                              , 695
 MD5      : qG/5fjKInTZJRE0xZyoGaA==         , QsVqVTE1WC4vDBdEZTaw8g==
File: /etc/gshadow
 Size     : 611                              , 639
 MD5      : +DPOfoec0xloDZitUQsNfA==         , NJwz3+Vb8XTkHEBUS326Rw==
File: /etc/gshadow-
 Size     : 624                              , 626
 MD5      : XnzbXrxEtNmbxtzIZmWOmQ==         , kjPj9roGj6/TL0pOZHOABQ==
File: /etc/passwd
 Size     : 1129                             , 1233
 MD5      : 39U+e6Xm+Ycd331YAOVBjg==         , b7FZ1kOkXUDmkRn3EUh5aA==
File: /etc/passwd-
 Size     : 1179                             , 1230
 MD5      : 9buIcnTszJ2Pl4vyrtuz2g==         , R5a3gCGFMwqUOhFzSypz5g==
File: /etc/shadow
 Size     : 904                              , 1160
 MD5      : yZJwf6hOhX4tC5MzGp6STg==         , LseU2PLXMCYmpJFxCyCD9Q==
File: /etc/shadow-
 Size     : 1031                             , 1033
 MD5      : J4rCXrwwTAXW8Thb/iOigA==         , SzneDJ6k1XEIg2MTiM9NLg==
File: /etc/sudoers
 Size     : 491                              , 514
 Perm     : -r--r-----                       , -r--r--r--
 Uid      : 0                                , 1000
 Gid      : 0                                , 1000
 MD5      : xdqw8ncUEe1+Z9batgoxHw==         , qbTHsTC0W+7pw1LLXZmHHQ==
File: /etc/teste.txt
 Size     : 40                               , 71
 MD5      : UeOHK/PWwiNIzMNKaOUr4w==         , Y7OuCsNpKqMhP7UaMJViGw==
File: /usr/sbin/useradd
 Size     : 87792                            , 6385
 MD5      : qFZsGW9T8B/1aY7NK8rZ8g==         , VewkvigI5g78jmJYY9X0AQ==
```

B.3 Relatório da ferramenta OSSEC

```
** Alert 1334939227.444: mail  - ossec,syscheck,
2012 Apr 20 13:27:07 OSSEC->syscheck
Rule: 550 (level 7) -> 'Integrity checksum changed.'
Integrity checksum changed for: '/etc/shadow-'
Size changed from '1114' to '1116'
Old md5sum was: 'f73aa91359a3ada1428aad8d28bf0288'
New md5sum is : 'b8131957edbcb337448592982e5926a1'
Old sha1sum was: '357900d08e8b81c85c77086ea55d00c402ca25f8'
New sha1sum is : '5225022ad6c5f79f5793b0bed90e3d744367c526'
--------------------------------------------------------
** Alert 1334939275.889: mail  - ossec,syscheck,
```

```
2012 Apr 20 13:27:55 OSSEC->syscheck
Rule: 550 (level 7) -> 'Integrity checksum changed.'
Integrity checksum changed for: '/etc/sudoers'
Size changed from '491' to '514'
Permissions changed from 'r--r-----' to 'r--r--r--'
Ownership was '0', now it is '1000'
Group ownership was '0', now it is '1000'
Old md5sum was: 'c5dab0f2771411ed7e67d6dab60a311f'
New md5sum is : 'a9b4c7b130b45beee9c352cb5d99871d'
Old sha1sum was: '9a8d2a6049c485d0ccf4125f55e4400cee2bc1fe'
New sha1sum is : '0677d712fbea893c67f399b1d59741640bdb994b'
--------------------------------------------------

** Alert 1334939291.1462: mail  - ossec,syscheck,
2012 Apr 20 13:28:11 OSSEC->syscheck
Rule: 550 (level 7) -> 'Integrity checksum changed.'
Integrity checksum changed for: '/etc/passwd'
Size changed from '1254' to '1358'
Old md5sum was: '029e7fdfefdeb8c3c99d1d06f5c83bcc'
New md5sum is : '9a6cd2525e5ed96a604898121f11c7c5'
Old sha1sum was: '3b5492f8f2c179ff9294374f5f6f3cf844407052'
New sha1sum is : '574f468359921753665cc7fae295c457c0783693'
--------------------------------------------------

** Alert 1334939291.1907: mail  - ossec,syscheck,
2012 Apr 20 13:28:11 OSSEC->syscheck
Rule: 550 (level 7) -> 'Integrity checksum changed.'
Integrity checksum changed for: '/etc/teste.txt'
Size changed from '69' to '89'
Permissions changed from 'rw-r--r--' to 'rwxrwxrwx'
Old md5sum was: '50005c32ef8a4e421c9aa04d806bd65d'
New md5sum is : 'dc21b6a1b5b801d7dc05ceb023201bdb'
Old sha1sum was: 'f3d9a224462359db866b5f62f4e69416f3011366'
New sha1sum is : '12090e95ff887638447a35c33289880a200d01dc'
--------------------------------------------------

** Alert 1334939343.2403: mail  - ossec,syscheck,
2012 Apr 20 13:29:03 OSSEC->syscheck
Rule: 550 (level 7) -> 'Integrity checksum changed.'
Integrity checksum changed for: '/etc/shadow'
Size changed from '987' to '1243'
Old md5sum was: '27bbef30f524626c67acf8a1048db241'
New md5sum is : '98514a42d3a0d1a1351f9827cddabb53'
Old sha1sum was: '95b597e74d69e72e94991de5e294c9f4873274e3'
New sha1sum is : 'e5d6de0e767bff01ff9e6ef9c6242fa6e4ca8d94'
--------------------------------------------------

** Alert 1334939345.2847: mail  - ossec,syscheck,
2012 Apr 20 13:29:05 OSSEC->syscheck
Rule: 550 (level 7) -> 'Integrity checksum changed.'
Integrity checksum changed for: '/etc/gshadow'
Size changed from '577' to '605'
Old md5sum was: 'b30a098a3d2ae49fcb9c7cbe15e1fdc5'
New md5sum is : 'a3d9b94683f2b951786e1e9d157db3a9'
Old sha1sum was: '25108b3fc63c14e59a0b067849637c98fec9265b'
```

```
New sha1sum is : '46775c4f002b44b340867101ec58581ba63a2d9e'
---------------------------------------------------
** Alert 1334939351.3291: mail  - ossec,syscheck,
2012 Apr 20 13:29:11 OSSEC->syscheck
Rule: 550 (level 7) -> 'Integrity checksum changed.'
Integrity checksum changed for: '/etc/group'
Size changed from '690' to '726'
Old md5sum was: 'b58a42657b6efb9a34faeec8a1a4b4f6'
New md5sum is : 'b288192eef44dc19ff318f2f82475245'
Old sha1sum was: 'dd1d5fbc6bc3822ca48bf07098ba0d54a8484683'
New sha1sum is : '710075c53bfe601ab39190739e965835889105ba'
---------------------------------------------------
** Alert 1334939363.3733: mail  - ossec,syscheck,
2012 Apr 20 13:29:23 OSSEC->syscheck
Rule: 550 (level 7) -> 'Integrity checksum changed.'
Integrity checksum changed for: '/etc/gshadow-'
Size changed from '590' to '592'
Old md5sum was: 'c30d1400a74013751e0a6389ed1b4d3c'
New md5sum is : '535292698cac19d69a83e9e573a21b00'
Old sha1sum was: 'ac7823c130411a54af29075b4765d720748e2a28'
New sha1sum is : '445b27f66a777ba4a330d68fd35f255c835d7ecb'
---------------------------------------------------
** Alert 1334939390.4178: mail  - ossec,syscheck,
2012 Apr 20 13:29:50 OSSEC->syscheck
Rule: 550 (level 7) -> 'Integrity checksum changed.'
Integrity checksum changed for: '/etc/passwd-'
Size changed from '1304' to '1355'
Old md5sum was: 'ffa0d2125b6432d3086480c37b69d8fd'
New md5sum is : '02e33208933319bd254840d5bbab0b36'
Old sha1sum was: 'de1d987ef9a1bc85841515235e98a51f4fdf284a'
New sha1sum is : '95860d1fd9b8e91bcd3147215d5574adb2581a47'
---------------------------------------------------
** Alert 1334939392.4624: mail  - ossec,syscheck,
2012 Apr 20 13:29:52 OSSEC->syscheck
Rule: 550 (level 7) -> 'Integrity checksum changed.'
Integrity checksum changed for: '/etc/group-'
Size changed from '707' to '709'
Old md5sum was: '44967633f110221253f6d735dca92c2a'
New md5sum is : 'fcc82b6010ddc1979de2acfbe9298dce'
Old sha1sum was: '792f686ba8b124c6914c6bf104917ff9b7a13a59'
New sha1sum is : '97f328aa5235a3abfd95ce709d6032140a224cf5'
---------------------------------------------------
** Alert 1334939537.5067: mail  - ossec,syscheck,
2012 Apr 20 13:32:17 OSSEC->syscheck
Rule: 550 (level 7) -> 'Integrity checksum changed.'
Integrity checksum changed for: '/usr/sbin/useradd'
Size changed from '87792' to '6385'
Old md5sum was: 'a8566c196f53f01ff5698ecd2bcad9f2'
New md5sum is : '55ec24be2808e60efc8e625863d5f401'
Old sha1sum was: '870ad9bdb4b67f04f051c5e06b1c80acd2118a99'
New sha1sum is : '58d387fbabcedf266c382001c173629d10c6a0cd'
```

B.4 Relatório da ferramenta Samhain

[SOF]
ALERT : [2012-04-18T20:18:57-0300] msg=<START>, program=<Samhain>, userid=<0>, path=</etc/samhainrc>,
hash=<6B28FC6B8EA7D1C5CB22D7F5778A9212404435BF7FA811AD>, path=</var/lib/samhain/samhain_file>,
hash=<67BAB9A6551021527AC00229D571476AD98E4890AC70C0AB>
706FFA512E590D8C147CC1FDEF40DAF9DAF7213676992E06

CRIT : [2012-04-18T20:18:58-0300] msg=<POLICY [IgnoreNone] --------T->, path=</bin>,
atime_old=<[2012-04-18T22:35:10]>, atime_new=<[2012-04-18T23:17:57]>, mtime_old=<[2012-04-14T18:13:23]>,
mtime_new=<[2012-04-18T23:11:26]>,
FA621B6264CDED7DDF36864FF7B0126CA7672EA80CE08852

INFO : [2012-04-18T20:18:58-0300] msg=<Checking [IgnoreNone]>, path=</bin>
55D8D00B3B92A641CA501CC63040D517F44CEC7D0BE9407D

CRIT : [2012-04-18T20:19:00-0300] msg=<POLICY ADDED>, path=</usr/bin/ssftp>, mode_new=<-rwxr-xr-x>,
attr_new=<------------>, imode_new=<33261>, iattr_new=<0>, hardlinks_new=<1>, idevice_new=<0>,
inode_new=<65577>, owner_new=<root>, iowner_new=<0>, group_new=<root>, igroup_new=<0>, size_old=<0>,
size_new=<6383>, ctime_new=<[2012-04-18T23:11:25]>, atime_new=<[2012-04-18T23:11:25]>,
mtime_new=<[2012-04-18T23:11:25]>, chksum_new=<C3BA1D7E3B89FF6C11F456DCCB4A63D9A2CD5156330152 98>
50AB84368A9312AA20F62ACD7554997E6F89583D81825C65

CRIT : [2012-04-18T20:18:58-0300] msg=<POLICY [IgnoreNone] --------T->, path=</etc>,
atime_old=<[2012-04-18T22:57:39]>, atime_new=<[2012-04-18T23:09:22]>, mtime_old=<[2012-04-18T22:57:24]>,
mtime_new=<[2012-04-18T23:11:25]>,
6F2AB9E8B344E279D8139C6980A6B13DB4C127758ACCF79C

INFO : [2012-04-18T20:18:58-0300] msg=<Checking [IgnoreNone]>, path=</etc>
830ED66C9432EDDF7E420AC3E8B52BD1A12F95226EADFA67

WARN : [2012-04-18T20:18:58-0300] msg=<No such group>, interface=<getgrgid>, group=<1005>,
path=</etc/teste.txt>
BC92FFFF21B3B392EE53615C9062D4336D7789C2CDCB52B0

WARN : [2012-04-18T20:18:58-0300] msg=<No such user>, interface=<getpwuid>, userid=<1005>,
path=</etc/teste.txt>
D1299C0EE734DB3BE06B7C0D204875396C44BDB7FACEDD86

CRIT : [2012-04-18T20:18:58-0300] msg=<POLICY [IgnoreNone] C-------TS>, path=</etc/teste.txt>,
size_old=<71>, size_new=<103>, atime_old=<[2012-04-18T22:57:24]>, atime_new=<[2012-04-18T23:09:49]>,
mtime_old=<[2012-04-18T22:57:24]>, mtime_new=<[2012-04-18T23:09:58]>,
chksum_old=<CF8D35DCBCE0D5A6CFFC00511FA2A5F6792A806F4FA329E8>,
chksum_new=<C84C6E7520DE0205B214DB4DA4A3C75D378C3458904E4DDD>,
634EC14533BF3991F525A7D27B48912E62AD217023619E63

CRIT : [2012-04-18T20:18:58-0300] msg=<POLICY ADDED>, path=</etc/ssftp.conf>, mode_new=<-rw-r--r-->,
attr_new=<------------>, imode_new=<33188>, iattr_new=<0>, hardlinks_new=<1>, idevice_new=<0>, inode_new=<460105>,
owner_new=<root>, iowner_new=<0>, group_new=<root>, igroup_new=<0>, size_old=<0>, size_new=<78>,
ctime_new=<[2012-04-18T23:11:25]>, atime_new=<[2012-04-18T23:11:25]>, mtime_new=<[2012-04-18T23:11:25]>,

chksum_new=<D97A4B4891775C73AF1B73C76C5DE372EDF38F8167B647A2>
78250BEBF94C17594A231A1FD2468B44528A5B3CAAC79C05

CRIT : [2012-04-18T20:18:58-0300] msg=<POLICY [IgnoreNone] C----MUGTS>, path=</etc/sudoers>,
mode_old=<-r--r------>, mode_new=<-r--r--r-->, attr_old=<------------>, attr_new=<------------>,
owner_old=<root>, owner_new=<michel>, iowner_old=<0>, iowner_new=<1000>, group_old=<root>, group_new=<michel>,
igroup_old=<0>, igroup_new=<1000>, size_old=<491>, size_new=<514>, atime_old=<[2012-04-18T22:44:46]>,
atime_new=<[2012-04-18T23:09:23]>, mtime_old=<[2012-04-18T22:44:52]>, mtime_new=<[2012-04-18T23:09:35]>,
chksum_old=<7C27EF6CD6CD7E6B064CA567DF786590CEC6F13FF735A4FB>,
chksum_new=<946FA5E492B573560537DFEFE795D9BE1E94E0589D70E3B1>,
D724653A31201C48130BBE34ADBA2A030B65E57738901C09

CRIT : [2012-04-18T20:18:58-0300] msg=<POLICY [IgnoreNone] C--I----TS>, path=</etc/group>, inode_old=<458884>,
inode_new=<460634>, size_old=<676>, size_new=<712>, atime_old=<[2012-04-18T22:13:09]>,
atime_new=<[2012-04-18T23:10:15]>, mtime_old=<[2012-04-14T18:29:03]>, mtime_new=<[2012-04-18T23:09:15]>,
chksum_old=<A267509A2054E58761ECE876392131A959C6201AF54788E1>,
chksum_new=<A40995010558078 35A83538754CAA850A4EC0A43D65AB2B8>,
47502BF293348F0D5FCAD8CF3D824A2ABAB2015B90E4E0B6

CRIT : [2012-04-18T20:18:58-0300] msg=<POLICY [IgnoreNone] C--I----TS>, path=</etc/passwd>, inode_old=<460639>,
inode_new=<460611>, size_old=<1129>, size_new=<1233>, atime_old=<[2012-04-18T22:13:11]>,
atime_new=<[2012-04-18T23:10:15]>, mtime_old=<[2012-04-14T18:29:03]>, mtime_new=<[2012-04-18T23:09:18]>,
chksum_old=<7C43E926CCEF10D9C779339B279CD2D12E526644D0E89016>,
chksum_new=<BEFE8C5204BA23A5AC9324AA2861958E12B605B6C0914EAE>,
24C7E3DA2E75291787CD6FD48D7ADC9A93E678A7429C011B

CRIT : [2012-04-18T20:18:59-0300] msg=<POLICY [IgnoreNone] C-------TS>, path=</etc/passwd->, size_old=<1175>,
size_new=<1230>, atime_old=<[2012-04-14T18:24:08]>, atime_new=<[2012-04-18T23:09:17]>,
mtime_old=<[2012-04-14T18:24:04]>, mtime_new=<[2012-04-18T23:09:15]>,
chksum_old=<797898512B67695598921DDD916D0C22B1FD9049BE877F6B>,
chksum_new=<C2FA2D9E35343BA32175CFC33C3503E624BCB5565CACF88F>,
AB8257CF8E06806CDF289AE3049CD6D05D9B8448C8170F16

CRIT : [2012-04-18T20:18:59-0300] msg=<POLICY [IgnoreNone] C-------TS>, path=</etc/gshadow>, size_old=<567>,
size_new=<595>, atime_old=<[2012-04-18T22:18:37]>, atime_new=<[2012-04-18T23:09:15]>,
mtime_old=<[2012-04-14T18:29:03]>, mtime_new=<[2012-04-18T23:09:15]>,
chksum_old=<A9741D89FB2B7EA855525C61D2ADFD6498BE5E5EE8B1C70D>,
chksum_new=<0F287EFDC33EFAC4FC286D7C314F1F64D85DD0D5FB0955EC>,
156382B1400D80617BBDB7E08C62F8F59E9F29BD256CF9EC

CRIT : [2012-04-18T20:18:59-0300] msg=<POLICY [IgnoreNone] C--I----TS>, path=</etc/shadow>, inode_old=<460105>,
inode_new=<458884>, size_old=<904>, size_new=<1160>, atime_old=<[2012-04-18T22:16:05]>,
atime_new=<[2012-04-18T23:17:01]>, mtime_old=<[2012-04-14T18:29:03]>, mtime_new=<[2012-04-18T23:09:17]>,
chksum_old=<E1FBD08B4632F45CC233569BEB15B258E41050D5AC8EAAE6>,
chksum_new=<8EFAD3C5C86A4264C391E8BB05B39C31F58A7AB29BB97F7A>,
A7E459F32F70B63E4968CFCD0EA2535D811C9527363455B4

CRIT : [2012-04-18T20:18:59-0300] msg=<POLICY [IgnoreNone] C-------TS>, path=</etc/gshadow->, size_old=<578>,
size_new=<582>, atime_old=<[2012-04-14T18:29:03]>, atime_new=<[2012-04-18T23:09:15]>,
mtime_old=<[2012-04-14T18:23:59]>, mtime_new=<[2012-04-18T23:09:08]>,
chksum_old=<BAE15DC5DF7BC3C232EEDF13907BD08F40E79BB2A5A17B58>,

chksum_new=<29378EF336D07B15208DD07A95ABC505AEDAD908E33D8949>,
CECDB4093675D6C960490C339C7F377FE5246FA3705C1F96

CRIT : [2012-04-18T20:18:59-0300] msg=<POLICY [IgnoreNone] C-------TS>, path=</etc/shadow->, size_old=<1029>,
size_new=<1033>, atime_old=<[2012-04-14T18:24:08]>, atime_new=<[2012-04-18T23:09:15]>,
mtime_old=<[2012-04-14T18:24:01]>, mtime_new=<[2012-04-18T23:09:10]>,
chksum_old=<4E7D247B5657809CD96F9D6AB1B055D5E48BDFE345B71CF9>,
chksum_new=<933A6DF6FFD3E86872F9E53AB859A65AEC2225D90AD1D461>,
6438C597F84AA3A9F7795257579C668389E2D873D428BEFB

CRIT : [2012-04-18T20:18:59-0300] msg=<POLICY [IgnoreNone] C-------TS>, path=</etc/group->, size_old=<691>,
size_new=<695>, atime_old=<[2012-04-14T18:24:08]>, atime_new=<[2012-04-18T23:09:15]>,
mtime_old=<[2012-04-14T18:23:59]>, mtime_new=<[2012-04-18T23:09:08]>,
chksum_old=<69D22AB2B19F9DA4D52C0053821823F01097EDD56C39A245>,
chksum_new=<5286B7855F1685A9C1EBC9617D99B3EA3AFF57552899D378>,
5A14B672F1B314561B571C6AA7190934D1FBB450D837432F

INFO : [2012-04-18T20:18:59-0300] msg=<Checking [IgnoreNone]>, path=</sbin>
8711569CFD18E91E6D899A8E75FDA72F6E7E1D6B8B6A3EF1

CRIT : [2012-04-18T20:18:59-0300] msg=<POLICY [IgnoreNone] --------T->, path=</usr/bin>,
atime_old=<[2012-04-18T22:35:10]>, atime_new=<[2012-04-18T23:17:57]>, mtime_old=<[2012-04-14T18:13:23]>,
mtime_new=<[2012-04-18T23:11:25]>,
5BF1D726E93389BC6D34D9CB4C65BC6C489E682B211306D9

INFO : [2012-04-18T20:18:59-0300] msg=<Checking [IgnoreNone]>, path=</usr/bin>
4278A839FCFA5EE7E8FFD1A7DB3637383AA3BE33580F25BF

CRIT : [2012-04-18T20:19:00-0300] msg=<POLICY ADDED>, path=</usr/bin/ssftp>, mode_new=<-rwxr-xr-x>,
attr_new=<---------->, imode_new=<33261>, iattr_new=<0>, hardlinks_new=<1>, idevice_new=<0>,
inode_new=<65577>, owner_new=<root>, iowner_new=<0>, group_new=<root>, igroup_new=<0>, size_old=<0>,
size_new=<6383>, ctime_new=<[2012-04-18T23:11:25]>, atime_new=<[2012-04-18T23:11:25]>,
mtime_new=<[2012-04-18T23:11:25]>, chksum_new=<C3BA1D7E3B89FF6C11F456DCCB4A63D9A2CD515633015298>
50AB84368A9312AA20F62ACD7554997E6F89583D81825C65

CRIT : [2012-04-18T20:19:04-0300] msg=<POLICY [IgnoreNone] --------T->, path=</usr/sbin>,
atime_old=<[2012-04-18T22:35:10]>, atime_new=<[2012-04-18T23:17:57]>, mtime_old=<[2012-04-14T18:13:23]>,
mtime_new=<[2012-04-18T23:11:26]>,
1572C53FB431A1BB97ED87720AF8CFBC47A6895CEC88B5AC

INFO : [2012-04-18T20:19:04-0300] msg=<Checking [IgnoreNone]>, path=</usr/sbin>
E2FD4419C069F0DE659402F7147964FEB28200732195B8F9

CRIT : [2012-04-18T20:19:05-0300] msg=<POLICY [IgnoreNone] C--I----TS>, path=</usr/sbin/useradd>,
inode_old=<66582>, inode_new=<66764>, size_old=<87792>, size_new=<6385>, atime_old=<[2012-04-14T18:23:59]>,
atime_new=<[2012-04-18T23:11:26]>, mtime_old=<[2011-02-15T20:54:14]>, mtime_new=<[2012-04-18T23:11:26]>,
chksum_old=<1D1349F864C61B2A380A57FBD21433DAA433BE3691C94832>,
chksum_new=<79AE859A852874D555CF1F56EF3617BBA99E1A10C6A992E3>,
263B3E047957CB70B208FE6196E0D4AD37FE0AB261359F22

CRIT : [2012-04-18T20:19:05-0300] msg=<POLICY MISSING>, path=</bin/ps>, mode_old=<-rwxr-xr-x>,

attr_old=<------------>, imode_old=<33261>, iattr_old=<0>, hardlinks_old=<1>, idevice_old=<0>,
inode_old=<426034>, owner_old=<root>, iowner_old=<0>, group_old=<root>, igroup_old=<0>, size_old=<99072>,
size_new=<0>, ctime_old=<[2012-04-14T18:13:23]>, atime_old=<[2012-04-18T22:13:17]>,
mtime_old=<[2010-05-04T11:26:22]>, chksum_old=<C5AA07D7E7EF728FC534CBED0B1B77CBD0CEFC5417C78731>
6A4752C832989D02047B30EF6FFEC1D8F67E561EBE4E6085

MARK : [2012-04-18T20:19:05-0300] msg=<File check completed.>, time=<7>, kBps=<10733.742000>
6FCCDEC9244D12420EFF75CAF9250463DFC917CF50EE3E74