

# Agente de aplicação para detecção, prevenção e contenção de ataques em ambientes computacionais

Leonardo C. Militelli, Fernando M. B. Nogueira, Volnys B. Bernal

Laboratório de Sistemas Integráveis – Universidade de São Paulo (USP)  
São Paulo - SP – Brasil

{leonardo, fbrusi, volnys}@lsi.usp.br

***Abstract.** Secure channels, as the ones created by protocols like SSL and TLS, has been used on network services to provide partner authentication, integrity and confidentiality. However, its utilization prevents a network intrusion detection system to observe and analyze packet contents. As an alternative to circumvent this problem, the present work proposes the ADACA (Attack Detection, Analysis and Containment Agent), a hybrid agent (passive/active) that is able to detect attacks where the application payload is encrypted by secure protocols. The results shown that is practicable to use an application agent attached to an application as a complement of network intrusion detection systems.*

***Resumo.** Canais seguros, como os gerados pelos protocolos SSL e TLS, são cada vez mais utilizados nos serviços de rede para propiciar autenticação de parceiro, integridade e sigilo dos dados. Porém, sua utilização impede que sistemas de proteção possam observar o conteúdo dos pacotes, impossibilitando a análise das mensagens trafegadas. Para contornar esse problema é proposto o ADACA (Agente de Detecção, Análise e Contenção de Ataques), um agente híbrido (passivo/ativo) que, por meio do acoplamento em elementos intermediários ou mesmo à própria aplicação, possibilita a análise das mensagens direcionadas a uma aplicação. Os resultados obtidos mostraram a viabilidade da utilização de agentes de aplicação, acoplados diretamente à aplicação, como complemento aos sistemas IDS de rede.*

## 1. Introdução

Segundo [Bace; Mell, 2001], IDS (*Intrusion Detection System*) é um sistema baseado em *software* ou *hardware*, normalmente interconectado com outros elementos, capaz de monitorar um canal de transmissão de dados e arquivos de registros de servidores em busca de tráfego anômalo e ações consideradas maliciosas, por meio de diferentes técnicas de detecção. O agente IDS, também conhecido como sensor, é o elemento responsável por capturar as informações que serão remetidas para análise.

A utilização de tal sistema tem se mostrado bastante útil frente ao aumento incessante de ataques partindo da Internet e, até mesmo, internamente aos ambientes corporativos [CERT.br, 2005].

Com base no estudo realizado nos trabalhos [Abbes; Bouhoula; Rusinowitch, 2004] e [Dasgupta et al., 2005], percebe-se que grandes esforços estão sendo realizados

na área de detecção de intrusão de forma a tornar viável a identificação de atividades maliciosas com eficácia, procurando reduzir a quantidade de falsos-positivos e falsos-negativos. Porém, neste contexto, ainda é possível observar certas lacunas com grande carência de pesquisas, como é o caso da identificação de ataques quando o alvo destes é um serviço ou aplicação que utiliza mecanismos de criptografia na comunicação.

Tais mecanismos foram concebidos com o intuito de garantir a confidencialidade dos dados e autenticação de parceiros. Porém, ao mesmo tempo em que garantem quesitos de segurança, estes mecanismos inviabilizam a análise do conteúdo das mensagens pelos sistemas de proteção, tornando possível camuflar ações maliciosas.

Para contornar esta situação, algumas empresas apresentaram sistemas de detecção no qual a chave privada da aplicação a ser protegida é compartilhada e utilizada para decifrar toda a comunicação realizada entre o cliente e servidor, como é o caso do *Intrushield SSL Traffic Inspection and Prevention* [MCAFEE, 2005] e do *BreachView SSL* [BREACH, 2005]. A principal ressalva desse método é que o compartilhamento da chave privada pode não ser aceito em muitas organizações, visto que esta prática pode infringir a política de segurança adotada.

Existem outras propostas que contemplam a detecção de ações maliciosas para situações de tráfego cifrado como, por exemplo, a detecção de anomalias por perfil de tráfego [YASINSAC, 2002] [LECKIE; YASINSAC, 2004], cuja análise dos pacotes é realizada na camada de transporte, e a monitoração de mensagens de protocolos utilizando máquinas de estado [JOGLEKAR; TATE, 2004].

O presente trabalho apresenta a proposta de uma arquitetura de detecção baseada em agente de aplicação, que possa ser diretamente acoplado a uma entidade atuante no fluxo de comunicação, seja à própria aplicação, proxy ou elemento de rede (*router*, *switch*, entre outros). Para esta arquitetura, são definidas algumas interfaces (APIs) tanto para o acoplamento à entidade quanto a outros componentes do ambiente. Espera-se realizar a detecção de ataques por meio da análise das mensagens do protocolo de aplicação imediatamente antes de seu processamento pela aplicação, no caso de uma requisição, e antes do envio de sua resposta ao cliente, permitindo inserir medidas de prevenção e contenção aos ataques e descartar mensagens não adequadas.

## **2. Agentes de detecção e prevenção**

Os sistemas de detecção e prevenção necessitam de um módulo responsável por coletar os dados que serão remetidos para análise, a fim de detectar atividades maliciosas. Este módulo é comumente conhecido como agente IDS/IPS e pode ser classificado de acordo com sua atividade, tipo de informação analisada, camada e forma de atuação, entre outros.

Especificamente para os sistemas de proteção para aplicações Web, uma nova classe de arquiteturas de segurança tem sido discutida e desenvolvida. Esta nova classe ainda não possui um conceito claro sobre o escopo de atuação e objetivos secundários, porém o termo WEF – *Web Application Firewall* – visa definir uma nomenclatura comum para esta categoria de sistemas, conforme apresentado em [Ristic, 2006]. Além disso, o *Web Application Security Consortium*, órgão internacional criado para definição de padrões na área de segurança de aplicações Web, elaborou um documento de critérios

de avaliação de WEFs (WAFEC, 2006), que compreende diversos aspectos relacionados a arquitetura, técnicas de detecção e proteção, entre outros.

## 2.1 Classificação dos agentes em relação à atividade

Atualmente, existem dois principais modos de operação para os agentes acima citados, sendo classificados em passivo e ativo.

- **Agente passivo:** é um componente de ação limitada, responsável por detectar ataques e reportar possíveis alertas ao IDS Central. Este tipo de agente não possui qualquer funcionalidade de contramedida aos alertas gerados, cabendo ao IDS Central interagir com os dispositivos do ambiente, *firewalls* e servidores, para contenção de ataques;
- **Agente ativo:** introduz o conceito mais aproximado ao *Intrusion Prevention System* (IPS) [ZHANG; LI; ZHENG, 2004] no qual o agente tem as funcionalidades de prevenir o sucesso da exploração de ataques ou minimizar o impacto de um ataque bem sucedido, por meio da intervenção direta na recepção dos pacotes considerados maliciosos.

## 2.2 Classificação de agentes em relação ao tipo de informação analisada

De acordo com [NIST, 1999], os agentes podem ser classificados de três formas diferentes, conforme a origem das informações a serem analisadas.

- **Agente baseado em rede (NIDS - *Network-based Intrusion Detection System*):** possui a capacidade de detectar a presença de tráfego anômalo ou pacotes com dados maliciosos por meio da escuta digital de pacotes e classificá-los de acordo com seu conteúdo. Apesar das vantagens existentes na utilização deste tipo de agente, como a monitoração de enlaces de alta velocidade, este tipo de agente é ineficaz na análise do conteúdo de fluxos de dados cifrados;
- **Agente baseado em máquina (HIDS - *Host-based Intrusion Detection System*):** tem como objetivo monitorar os recursos do sistema operacional, como os processos ativos, trilhas de auditoria, usuários, entre outros;
- **Agente baseado em aplicação:** tem por objetivo monitorar uma única aplicação, como um antivírus em servidor de correio eletrônico, e é capaz de validar toda a interação entre o usuário, os dados e a aplicação [BACE; MELL, 2001].

[POSEY, 2005] ressalta que apesar do escopo de atuação ser diferente, o HIDS e o agente baseado em aplicação consomem uma série de recursos locais, como memória, processamento e banda, devido ao constante monitoramento de arquivos e troca de informações com o servidor central do IDS. Tais atividades podem acrescentar certa latência no funcionamento da aplicação.

## 2.3 Classificação quanto à camada de captura de pacotes

Um agente pode ser classificado de acordo com a camada que realiza a captura de pacotes e pode ser classificado em:

- Captura na camada de aplicação;

- Captura na camada de rede;
- Captura na camada de enlace.

## 2.4 Classificação quanto à forma de atuação

Quanto à forma de atuação, o agente pode ser classificado em:

- **Atuação por observação (*sniffer*):** o agente é capaz de realizar somente a observação de pacotes, não atuando no fluxo de comunicação;
- **Atuação por repasse (*inline*):** o agente recebe o pacote, o analisa e repassa. Nesta situação, o agente impõe latências ao fluxo normal e é capaz de atuar sobre o fluxo de comunicação;
- **Atuação por intermediação (*proxy*):** no caso específico de agente com atuação na camada de aplicação, o agente pode operar como uma entidade intermediária da comunicação. Neste caso, o agente deve ser capaz de entender cada mensagem e intermediar a comunicação.

## 2.5 Classificação quanto à entidade a ser protegida

Os agentes podem ser inseridos em pontos da topologia com a finalidade de proteção de entidades servidoras, entidades clientes ou ambas. Assim, os agentes podem ser classificados quanto à finalidade em:

- Proteção de entidades clientes;
- Proteção de entidades servidoras;
- Híbrido (proteção de cliente e servidor).

## 2.6 Acoplamento de um agente de aplicação à topologia

Diferentemente dos agentes NIDS e HIDS, um agente de aplicação pode ser acoplado a diferentes entidades da topologia de comunicação, conforme apresentado na Figura 1.

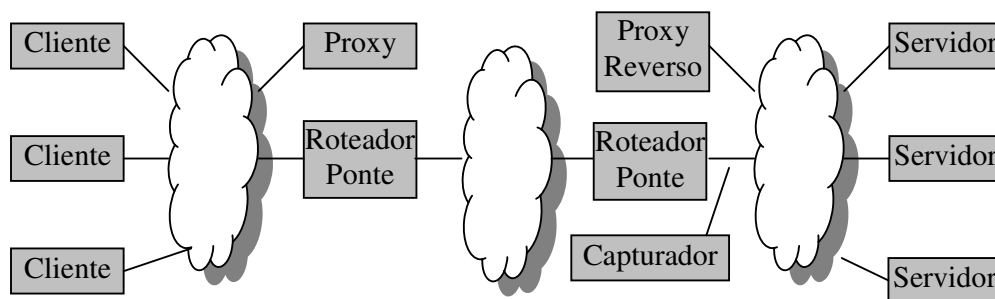


Figura 1. Elementos relevantes para acoplamento de agentes

Os principais pontos para acoplamento de agentes de aplicação são especificados a seguir e referenciados na Figura 2:

- **(a) Acoplado a capturador (*sniffer*):** agente acoplado a um capturador de pacotes (*sniffer*), tipicamente atuando na camada de enlace por observação;

- **(b) Acoplado a elemento de rede:** agente acoplado a um elemento de rede, tipicamente um elemento roteador ou ponte (*bridge*), podendo realizar captura em qualquer camada (aplicação, rede ou enlace) e operar tanto por observação quanto por repasse. Para canais SSL/TLS, quando utilizado para proteção de entidade servidora, pode possuir módulo de decodificação adicional e ter capacidade de analisar as mensagens trafegadas por meio do compartilhamento da chave privada;
- **(c) e (d) Acoplado a proxy reverso:** agente com captura na camada de aplicação, para proteção de entidades servidoras, podendo operar de forma ativa ou passiva. Operam tipicamente por intermediação, porém, para mensagens em canais SSL/TLS operam por repasse, sendo incapaz de observar o conteúdo da mensagem cifrada. Alternativamente para canais SSL/TLS, pode possuir módulo de decodificação adicional e ter capacidade de analisar as mensagens trafegadas por meio do compartilhamento da chave privada;
- **(e) Acoplado ao servidor:** acoplado diretamente ao provedor do serviço (entidade servidora), com captura na camada de aplicação e atuação por observação ou repasse. Nesta situação, o provedor do serviço interage com o agente a cada mensagem recebida ou transmitida. Quando atuando por repasse, pode operar tanto no modo ativo ou passivo. Um agente acoplado ao servidor não possui problemas em relação à decodificação de mensagens provenientes de canais seguros, pois a mensagem repassada para análise é obtida após a extremidade do canal seguro;
- **(f) Acoplado a aplicação cliente:** agente acoplado diretamente à aplicação cliente, com captura na camada de aplicação e atuação por observação ou repasse. Quando atuando por repasse, pode operar tanto no modo ativo ou passivo. Nesta situação, a entidade cliente, a cada mensagem recebida ou transmitida, interage com o agente. Não possui problemas em relação à decodificação de mensagens cifradas de canais SSL/TLS, pois a mensagem repassada para análise é obtida após a extremidade do canal seguro;
- **(g) Acoplado ao proxy direto:** agente com captura na camada de aplicação, para proteção de entidades clientes, podendo operar de maneira ativa ou passiva. Operam tipicamente por intermediação, porém, para mensagens em canais SSL/TLS operam por repasse, sendo incapaz de observar o conteúdo da mensagem cifrada. Como é utilizado para proteção de clientes, não possui a possibilidade de decodificação de mensagens SSL/TLS.

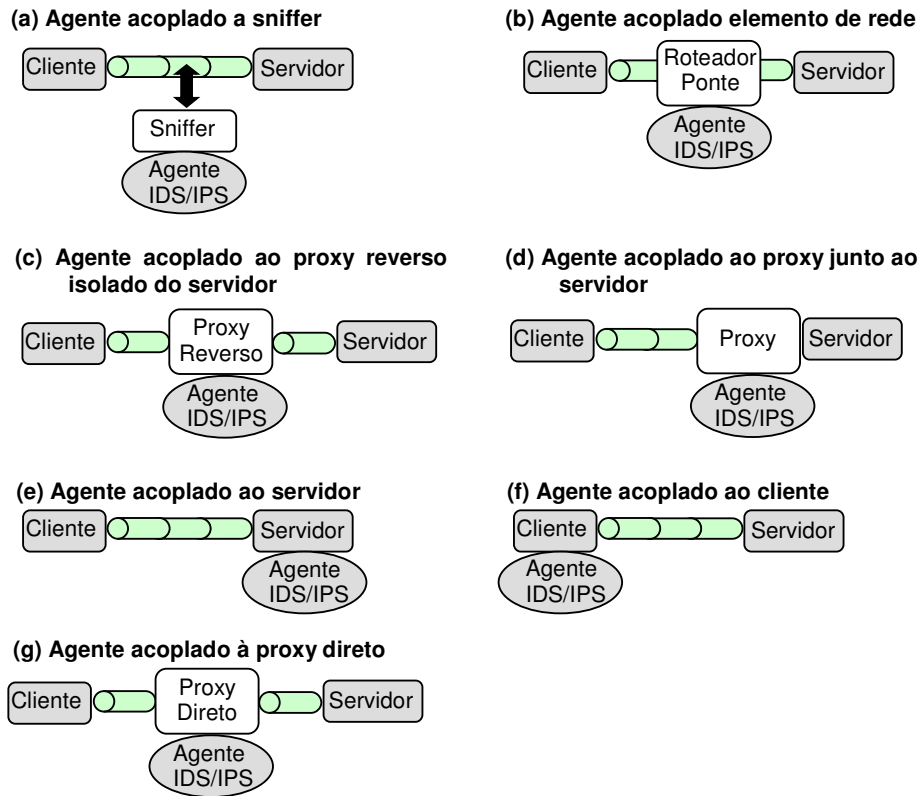


Figura 2. Arquiteturas de agentes de detecção/prevenção

### 3. Arquitetura Proposta

A proposta do **Agente de Detecção, Análise e Contenção de Ataques (ADACA)** faz parte de um arcabouço de detecção de intrusão que é composto por agentes NIDS, agentes ADACA e a central de correlação de alertas e gerenciamento (**IDS Central**), mostrado na Figura 3. O agente de aplicação, foco do trabalho, foi integrado diretamente à uma aplicação, porém, poderia ser facilmente integrado a outras entidades.

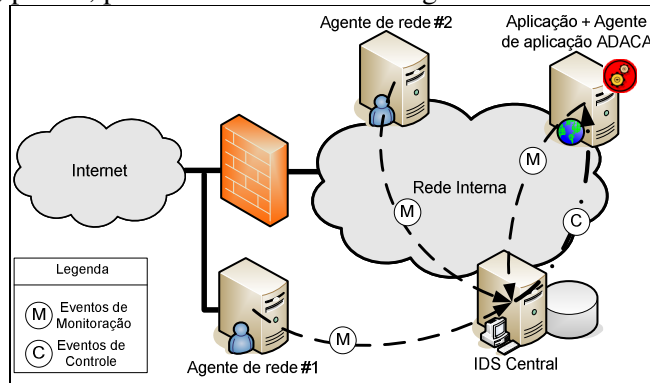
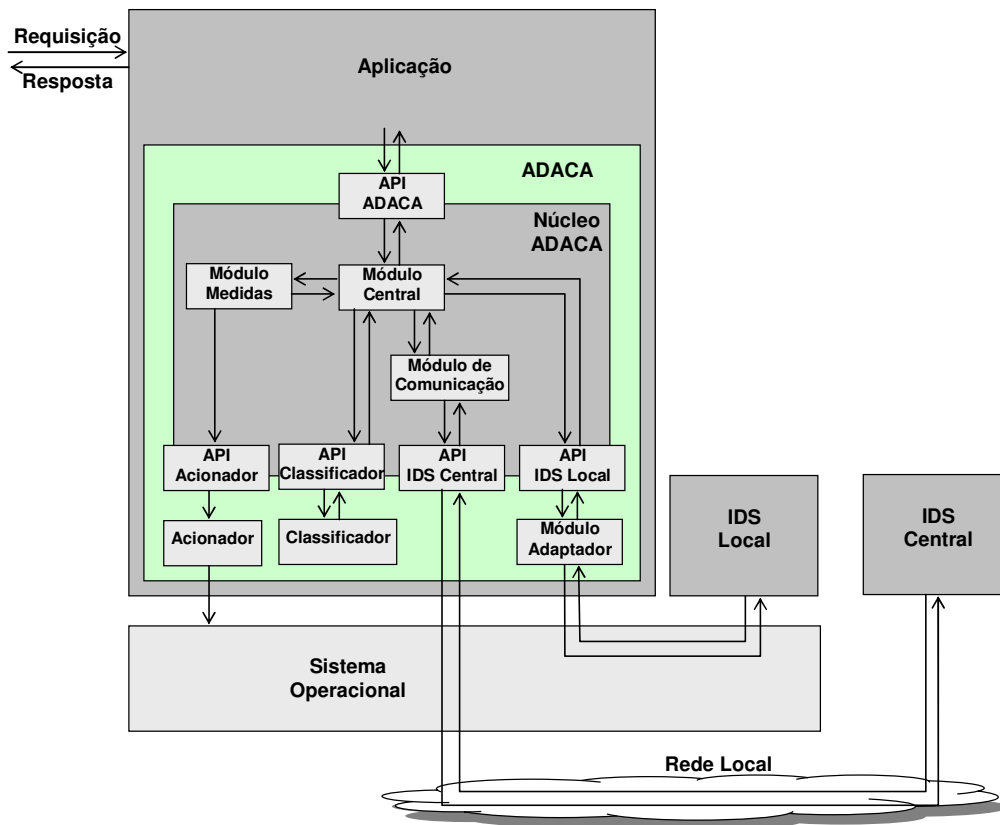


Figura 3. Exemplo de arcabouço de detecção: aplicação com ADACA, sensores de rede e central de gerência e correlação.

Para a realização da comunicação entre os elementos citados na Figura 3, foram definidos dois tipos de eventos. Os eventos de monitoração são as mensagens nas quais estão contidos os alertas detectados pelos agentes distribuídos ao longo do ambiente de rede e enviados ao IDS Central, a fim de informá-lo sobre a ocorrência de atividades anômalas. Já os eventos de controle são mensagens enviadas unidirecionalmente pelo IDS Central aos agentes, quando estes suportam tais mensagens, e neles estão contidos os comandos que deverão ser executados no agente.

O ADACA é um agente híbrido, ou seja, pode ser configurado para operar nos modos ativo e passivo, possibilitando realizar a detecção, prevenção e contenção de requisições maliciosas e respostas anômalas ou não autorizadas. A contenção e prevenção de ataques podem ser motivadas pelos alertas detectados localmente e pelo resultado do correlacionamento dos eventos no IDS Central.

A arquitetura do ADACA foi desenvolvida levando-se em consideração a portabilidade entre diversos sistemas operacionais e a possibilidade de integração a diferentes aplicações e entidades presentes em uma rede, conforme apresentado na seção 2.6. Para isto, foi definido um núcleo único no qual estão compreendidos os componentes de análise, comunicação e atuação, além das interfaces padronizadas para interação com os componentes externos.



**Figura 4. Arquitetura do ADACA e suas interações com outros componentes**

A Figura 4 apresenta os elementos que compõem o ADACA e as interações realizadas entre eles, sendo:

- **API ADACA:** viabiliza a obtenção das informações sobre a mensagem junto da aplicação utilizando a função *Analisa\_dados*(sentido,IPo, Po, IPd, Pd, Mensagem, Contexto);
- **Módulo Central:** coordena as atividades de análise, as ações realizadas pelo agente ADACA e registra as incidências detectadas em uma tabela;
- **API Classifica:** utiliza a função *Classifica\_mensagem*( ) para submeter a mensagem para análise junto ao módulo Classificador;
- **Classificador:** classifica a mensagem com base em uma série de regras definidas em arquivo de configuração;
- **API IDS Local:** com base nas informações obtidas pela API ADACA, cria um pacote no formato PCAP, suportado por diversos analisadores de pacotes e sistemas de detecção, e o submete para análise ao IDS Local por meio da função *Analisa\_IDS*( );
- **IDS Local:** responsável por analisar o pacote no formato PCAP, ou em formato definido pelo módulo adaptador, e retorna o resultado ao Módulo Central;
- **Módulo Comunica:** adequa os eventos de monitoração ao padrão IDMEF e gerencia toda comunicação realizada entre o ADACA e o IDS Central;
- **API IDS Central:** define a interface de funções entre o agente ADACA e o IDS Central remoto por meio das funções *Enviar\_alerta*( ) e *Callback\_controle*( ), para o envio de eventos de monitoração e controle, respectivamente;
- **Módulo Medidas:** gerencia as medidas existentes, registra novas medidas a serem executadas, verifica a existência de alguma medida pré-definida para cada mensagem e controla o tempo de execução das medidas;
- **API Acionador:** relaciona as ações que podem ser realizadas sobre os recursos do sistema operacional, como a criação de uma regra junto ao *firewall* local, a reiniciação de uma determinada conexão e o redirecionamento do tráfego de um endereço IP origem para outro destino;
- **Módulo Acionador:** responsável por mapear as ações definidas na API Aciona em comandos do sistema operacional, de acordo com o sistema adotado;
- **IDS Central:** gerencia os agentes ADACA, mantém um repositório unificado de alertas de todos agentes, possibilitando o correlacionamento das informações, e permite o envio de mensagens de controle por meio de interação humana aos agentes que possuam tal funcionalidade.

### 3.1 Alertas e mensagens de controle

Com o intuito de viabilizar o correlacionamento de eventos e gerenciamento de medidas de controle por um IDS Central, se faz necessário uniformizar o formato dos alertas gerados pelos agentes IDS/IPS e *firewalls* distribuídos ao longo do ambiente de rede, de forma a não haver qualquer problema de interoperabilidade [Kahn et al, 1998].

Algumas propostas foram apresentadas como é o caso do CIDF, proposto em [Kahn et al, 1998] e [Staniford-Chen; Tung; Schnackenberg, 1998], que especifica uma linguagem representativa e define um protocolo de comunicação para troca de informações de alertas e respostas aos ataques. Além desta iniciativa, foi criado o grupo *Intrusion Detection Exchange Format Working Group* – IDWG - do IETF, o qual tem por objetivo propor padrões relacionados ao formato de mensagens e protocolos de comunicação de informações de alertas.

Dentre os principais documentos elaborados por este grupo, destaca-se a especificação do padrão do formato das mensagens de alerta (IDMEF – *Intrusion Detection Message Exchange Format*) e os requisitos necessários para sua implementação, o protocolo de comunicação IDP (*IDMEF Communication Protocol*) e seus requisitos de segurança [Wood, Erlinger, 2002]. Em seguida, foi definido o modelo de dados baseado em XML para representação do IDMEF [Debar; Curry; Feinstein, 2006] e dois protocolos de comunicação: o *Intrusion Alert Protocol* (IAP) [Gupta et al, 2001] e o *Intrusion Detection Exchange Protocol* (IDXP) [Feinstein; Matthews; White, 2002]. Algumas destas especificações foram submetidas ao *Internet Engineering Steering Group* (IESG) e encontram-se em fase final de rascunho (*draft*) de padrão.

Com o intuito de estender a utilização do IDMEF, foi apresentado o *Intrusion Detection Response Exchange Format* – IDREF – [Silva; Westphall, 2005], que estende a utilização do modelo IDMEF com a funcionalidade de envio de respostas, sejam elas ativas ou informativas, aos ataques detectados.

### 3.2 Tabela de controle

Para cada mensagem submetida ao agente ADACA para análise é criado um registro na tabela de controle, conforme exemplifica a Tabela 1. Neste registro são mantidas informações relevantes associadas ao processo de análise desta mensagem, incluindo os parâmetros informados pela aplicação.

As informações relacionadas ao sentido da mensagem, endereço IP origem, porta origem, endereço IP destino, porta destino, mensagem e contexto foram obtidas da chamada da função *Analisa\_dados*(sentido,IPo, Po, IPd, Pd, Mensagem, Contexto) da API ADACA. O instante é determinado e registrado pelo núcleo do agente ADACA. Os resultados da classificação e análise da mensagem são registrados após o processamento do classificador e análise do IDS Local, respectivamente.

**Tabela 1. Exemplo de tabela de controle**

#	Alerta	Classific.	IP Orig.	P. Orig.	IP Dest.	P. Dest.	Mensagem	Contexto	Hora Data
1	78	GET	10.0.0.10	4567	10.0.0.2	80	GET /etc/passwd		14:30:4567 05/01/06
2		GET	10.0.0.6	3205	10.0.0.2	80	GET /		14:31:7892 05/01/06
3		OPTIONS	10.0.0.13	3890	10.0.0.2	443	OPTIONS /		14:32:2309 05/01/06
4	54	GET	10.0.0.11	2015	10.0.0.2	443	GET ../global.asa		14:32:3678 05/01/06

## 4. Prova de conceito

Para possibilitar a validação do arcabouço de detecção apresentado neste trabalho, foi desenvolvido o protótipo do ADACA. Foi escolhido o servidor Web Apache para ser utilizado como prova de conceito, uma vez que esta solução é de código-livre além de ser o servidor utilizado por aproximadamente 67% dos 15 milhões de sites avaliados, de acordo com a estatística [Netcraft, 2006].

A versão do Apache escolhida foi a 1.3.34, atualmente a mais recente da distribuição 1.3. As distribuições 2.0 e 2.2 não foram consideradas, pois realizam o processamento das mensagens em múltiplas linhas de execução (*multithreads*), o que dificultaria a codificação do protótipo.

Para o desenvolvimento do ADACA foi utilizada a linguagem C.

### 4.1 Integração do ADACA ao servidor Web

De forma a tornar possível integração do ADACA, se fez necessário realizar uma análise detalhada do código-fonte e das funções utilizadas pelo Apache, no tratamento de requisições e respostas. Foi identificado que a função “*ap\_process\_request()*”, do arquivo “*http\_request.c*”, realiza o processamento das requisições no servidor Apache. Em seguida, foram mapeadas as variáveis que armazenam as informações a respeito da mensagem, como endereços IPs e portas.

A função *Analisa\_Dados()* da API ADACA foi inserida antes da função “*ap\_process\_request()*”, uma vez que o Apache realiza diversas validações da mensagem da requisição entrante com o intuito de verificar a presença de requisições incompletas antes de chegar nesta função. Para realizar a análise da resposta a chamada da função *Analisa\_Dados()* da API ADACA foi inserida entre a função “*ap\_process\_request()*” e a função “*ap\_bhalfduplex()*”. A função “*ap\_bhalfduplex()*” finaliza a formatação da mensagem de resposta antes que sejam gerados os registros (logs) de auditoria e o envio da resposta ao cliente pelo Apache.

### 4.2 Componente IDS Local

O componente IDS Local, apesar de não ser o objeto de pesquisa deste trabalho, é imprescindível, pois o processo de análise da mensagem faz uso da inteligência e dos mecanismos de detecção implementados por este componente externo ao agente. Portanto, foi definido que a solução Snort deveria ser utilizada para análise das mensagens recebidas pelo ADACA, uma vez que esta ferramenta é de livre distribuição, aceita arquivos de entrada no formato PCAP e possibilita o envio dos alertas detectados por meio de um *socket* interno, possibilitando a comunicação entre processos.

### 4.3 Modulo Comunica

Para implementação do Módulo Comunica foi utilizada a biblioteca LibIDMEF (POPPI, S.; MIGUS, A.; MCALERNEY, J., 2005) que possibilita a padronização dos alertas ao formato IDMEF.

#### 4.4 Testes

A etapa de testes teve como principal objetivo realizar o estudo de viabilidade do ADACA e, conseqüentemente, medir o tempo gasto por ele para verificar, em cada uma das mensagens, a presença de dados maliciosos ou respostas não-autorizadas.

Portanto, também se fez necessário avaliar as características dos dois modos de operação do agente, sendo considerada três situações:

- Aplicação WEB sem agente;
- Aplicação WEB e ADACA operando no modo passivo;
- Aplicação WEB e ADACA operando no modo ativo.

Para cada situação, foram disparadas requisições normais e maliciosas com objetivo de diversificar o tipo de tráfego e explorar as funcionalidades implementadas pelo agente. Portanto, foi possível observar cada uma das etapas do processo de classificação, análise e tomada de decisão do agente.

#### 5. Análise de resultados

Foi realizada a medição do tempo gasto pelo Apache sem acoplamento do agente para executar a função “*ap\_process\_request( )*”, a qual realiza o processamento da requisição no Apache. Em seguida, foram colhidos os resultados computando a sobrecarga adicionada com a inserção do ADACA nos modos passivo e ativo.

A medida do tempo foi realizada por meio da função “*getsystemtime( )*”, nativa da linguagem C. Para as medições com o ADACA em operação, esta função foi posicionada antes da chamada à função “*Analisa\_dados( )*” da API ADACA e logo após a função “*ap\_process\_request( )*” do Apache. No caso das medições sobre a aplicação original, a função para obtenção do tempo foi posicionada antes e depois da função “*ap\_process\_request( )*”. Este ponto foi escolhido, pois reflete somente a sobrecarga introduzida com a utilização do ADACA.

Por meio de uma amostra de cem requisições a uma mesma URL, a média ponderada do tempo de processamento de cada requisição obtida para o Apache original foi de 101 ( $\pm 31$ ) microssegundos. Os valores obtidos com base na mesma amostragem para o Apache com o ADACA no modo ativo foi de 270 ( $\pm 94$ ) microssegundos. Não foram observados atrasos significativos no processamento de uma mensagem individual com o agente operando no modo passivo. Portanto, observa-se que a utilização do ADACA no modo ativo introduz um aumento da latência de resposta de aproximadamente duas vezes quando comparado à aplicação original.

Visto que a ordem de grandeza das aferições é de microssegundos, pode-se concluir que a inserção do ADACA no servidor WEB Apache não acarretou perdas consideráveis de tempo.

Um fator limitante desta implementação é que o protótipo desenvolvido não é reentrante, impossibilitando realizar análises de mensagens em paralelo.

## **6. Conclusão**

Este trabalho apresentou uma proposta de um arcabouço de detecção, sendo que o foco foi concentrado na definição da arquitetura de um agente de aplicação que fosse capaz de analisar mensagens de requisições e respostas.

O agente ADACA, quando acoplado diretamente à aplicação, tem como característica particular o recebimento de mensagens de aplicação fora do canal seguro, sem haver necessidade do compartilhamento da chave privada do certificado digital da aplicação. Desta forma, o tráfego passante cifrado, que só podia ser analisado pelos sistemas de detecção que compartilham a chave privada do serviço seguro, agora pode ser tratado de forma adequada, sem inserir um outro ponto passível de comprometimento desta chave.

Como característica operacional, o ADACA preveniu o processamento de requisições maliciosas e o envio de conteúdo não-autorizado na resposta. Além disso, foi possível instanciar as medidas de contenção e prevenção por meio do módulo medidas e executá-las com o módulo acionador. No caso de execução de uma medida de contenção (agente no modo passivo), notou-se que, para uma página contendo imagens, o carregamento não se deu por completo, pois a medida surtiu efeito antes mesmo que as requisições para aquisição das imagens fossem processadas.

A principal desvantagem observada com a arquitetura proposta é a necessidade de modificar a aplicação para integração do agente ADACA.

A análise dos resultados obtidos com a medição do tempo despendido para análise, classificação e tomada de decisão por parte do ADACA, nos diferentes modos de operação, permitiu concluir que a presente proposta é bastante aceitável, visto que a latência adicional inserida em cada mensagem é da ordem de microssegundos.

Finalizando, pode-se citar como contribuições importantes do trabalho a proposta de uniformização da taxonomia de classificação de agentes e a definição de um modelo de agente de aplicação e seus principais componentes, que pode ser utilizado como referência para trabalhos futuros da área de detecção de intrusos.

### **6.1 Trabalhos futuros**

No trabalho apresentado, o classificador tem uma função bastante simples, porém fundamental, a qual poderia ser incrementada por meio da implementação de uma máquina de estados que possibilitaria definir novas funcionalidades para este módulo, como a análise das mensagens trocadas com um mesmo parceiro.

As funções definidas para a API Acionador apenas abrangem um conjunto mínimo de funções. Portanto, é necessário realizar melhorias nesta API com um conjunto de funções mais completo para as atividades de prevenção e contenção.

A formatação das mensagens de controle utilizando a proposta IDREF e a implementação do protocolo de comunicação IDXP são dois pontos importantes para a padronização e interoperabilidade entre sistemas de detecção de intrusos. Para tal, deve-se criar uma interface de integração entre as linguagens Java e C ou ainda realizar a interpretação do código fonte das bibliotecas somente disponíveis em Java, com a finalidade de reproduzir o mesmo feito na linguagem C.

Um outro estudo relevante a ser realizado é o acoplamento do ADACA a outras entidades de rede, conforme citado no decorrer deste trabalho.

## 7. Referências

- Abbes, T., Bouhoula, A. and Rusinowitch, M. (2004) "Protocol Analysis in Intrusion Detection Using Decision Tree". IEEE International Conference on Information Technology: Coding and Computing.
- Bace, R. and Mell, P. (2001) "Special Publication 800-31: Intrusion Detection Systems". NIST - National Institute of Standards and Technology. <http://csrc.nist.gov/publications/nistpubs/800-31/sp800-31.pdf>
- Breach. (2005) "BreachView SSL White Paper". Breach Security. [http://www.breach.com/downloads/product/BreachView\\_SSL\\_White\\_Paper.pdf](http://www.breach.com/downloads/product/BreachView_SSL_White_Paper.pdf)
- CERT.br. (2006) "Estatísticas do CERT.br " Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil.. <http://www.cert.br/stats/incidentes/>.
- Dasgupta, D., Gonzalez, F., Yallapu, K., Gómez, J., Yarramsetti, R., Dunlap, G. and Greveas, M. (2005) "CIDS: An Agent-based Intrusion Detection System". Computers & Security Journal, April 05.
- Debar, H., Curry, D. and Feinstein, B. (2006) "The Intrusion Detection Message Exchange Format" Internet-Draft, IETF, Março. <http://www.ietf.org/internet-drafts/draft-ietf-idwg-idmef-xml-16.txt>
- Feinstein, B.; Matthews, G.; White, J. (2002) "The Intrusion Detection Exchange Protocol (IDXP)" Internet-Draft, IETF, Outubro. <http://www.ietf.org/internet-drafts/draft-ietf-idwg-beep-idxp-07.txt>
- Gupta, D., Buchheim, T., Feinstein, B., Matthews, G. and Pollock, T. (2001) "IAP: Intrusion Alert Protocol" Internet-Draft, IETF, Fevereiro <http://www.ietf.org/internet-drafts/draft-ietf-idwg-iap-05.txt>
- Joglekar, S. P. and Tate, S. R. (2004) "ProtoMon: Embedded Monitors for Cryptographic Protocol Intrusion Detection and Prevention". IEEE International Conference on Information Technology: Coding and Computing.
- Kahn, C., Porras, P. A., Staniford-Chen, S. and Tung, B. (1998) "A Common Intrusion Detection Framework" Journal of Computer Security, July.
- Leckie, T.; Yasinsac, A. (2004) "Metadata for Anomaly Based Security Protocol Attack Detection." IEEE Transactions on Knowledge and Data Engineering, Volume 16, Number 10.
- McAfee. (2005) Encrypted Threat Protection: Network IPS for SSL Encrypted Traffic. McAfee. [http://www.mcafee.com/us/local\\_content/white\\_papers/wp\\_encr\\_th\\_prot.pdf](http://www.mcafee.com/us/local_content/white_papers/wp_encr_th_prot.pdf).
- Netcraft. Web Server Survey Archives. Fevereiro, 2006. [http://news.netcraft.com/archives/web\\_server\\_survey.html](http://news.netcraft.com/archives/web_server_survey.html)

- NIST. (1999) "Acquiring and Deploying Intrusion Detection Systems". ITL Computer Security Bulletins. NIST. November.
- Posey, B. M. (2005) "Choosing an intrusion detection system: Network, host or application-based IDS". SearchWindowsSecurity.com [http://searchwindowssecurity.techtarget.com/tip/1,289483,sid45\\_gci1083969,00.html](http://searchwindowssecurity.techtarget.com/tip/1,289483,sid45_gci1083969,00.html)
- Ristic, I. (2006) "Web Application Firewalls Primer". Insecure Magazine, issue 5, pag. 6. January
- Silva, P. F., Westphall, C. B. (2005) "Aperfeiçoamentos do Modelo para Respostas de Detecção de Intrusão Compatível com o Modelo IDWG". V Simpósio Brasileiro em Segurança de Sistemas Computacionais (SBSEG 2005), Florianópolis - SC.
- Staniford-Chen, S., Tung, B. and Schnackenberg, D. (1998) "The Common Intrusion Detection Framework (CIDF)" Information Survivability Workshop, Outubro.
- Yasinsac, A. (2002) "An Environment for Security Protocol Intrusion Detection". Journal of Computer Security.
- WAFEC. (2006) "Web Application Firewall Evaluation Criteria" Web Application Security Consortium. <http://www.webappsec.org/projects/wafec/v1/wasc-wafec-v1.0.pdf>
- Wood, M. and Erlinger, M. (2002) "Intrusion Detection Message Exchange Requirements" Internet-Draft, IETF, Outubro. <http://www.ietf.org/internet-drafts/draft-ietf-idwg-requirements-10>.
- Zhang, X., Li, C. and Zheng, W. (2004) "Intrusion Prevention System Design". Computer and Information Technology, September.