E-Sentry+: Um IDS Baseado em Rede com Suporte à Especificação em Alto Nível de Assinaturas de Ataque

Marlom Alves Konrath[1], Josué Sperb[2], Eduardo Isaia Filho[2],
Luciano Paschoal Gaspary[1], Liane Tarouco[2]

[1]Universidade do Vale do Rio dos Sinos
Centro de Ciências Exatas e Tecnológicas
Av. Unisinos, 950 – CEP 93.022-000 – São Leopoldo, Brasil

[2]Universidade Federal do Rio Grande do Sul
Instituto de Informática
Av. Bento Gonçalves, 9500 – CEP 91.591-970 – Porto Alegre, Brasil

marlom@bage.unisinos.br

Resumo

O aumento da utilização das redes de computadores e o crescimento dos serviços por elas suportados têm causado um aumento no número de incidentes de segurança. Os sistemas de detecção de intrusão (IDS) têm sido usados, nesse contexto, para identificar atividades maliciosas como sondagens e ataques. No caso dos IDSs baseados em assinatura, é preciso descrever essas atividades para que eles sejam capazes de monitorar a sua ocorrência. A maioria dos sistemas, no entanto, não oferece flexibilidade a esse processo de especificação, além de exigir a utilização de notações em baixo nível. Esse trabalho apresenta um sistema de detecção de intrusão, o E-Sentry+, que aborda esse problema ao explorar uma linguagem em alto nível para a representação de assinaturas.

Abstract

The growth of the use of computer networks and the services supported by it ended up increasing the number of security incidents. Intrusion detection systems (IDSs) have been used, in this context, to identify malicious activities such as scans and attacks. Signature-based IDSs require the description of such activities so that they can monitor their occurrence. Most of the systems, however, do not offer flexibility to this specification process and require the usage of low-level notations. This work presents an intrusion detection system, E-Sentry+, which addresses this problem by exploring a high-level language to signature representation.

Palavras-chave: segurança de redes, sistemas de detecção de intrusão, assinaturas de ataques.

1. Introdução

O aumento da utilização, por parte de empresas e usuários, das redes de computadores, aliado com o crescimento, em quantidade e complexidade, dos serviços por elas suportados, têm causado um aumento no número de incidentes de segurança. O nível de proteção contra esses incidentes depende diretamente da quantidade de tempo, da estratégia utilizada e do esforço realizado na construção de um sistema seguro. Várias são as técnicas e ferramentas disponíveis para auxiliar esse processo e entre elas encontram-se os sistemas de detecção de intrusão. Segundo Northcutt [1], o objetivo dos IDSs é identificar ameaças direcionadas a uma organização e, depois, garantir que os sistemas sejam protegidos contra elas.

A rápida proliferação de novos protocolos e aplicações torna essa proteção cada vez mais complicada. De um lado, protocolos com falhas no projeto, na implementação ou pouco testados acabam sendo utilizados com graves falhas de segurança. Por outro lado, os sistemas de detecção de intrusão, na sua grande maioria, não são suficientemente flexíveis para monitorar os novos cenários que se apresentam.

Essa pouca flexibilidade deve-se, principalmente, às limitações e à complexidade das linguagens para expressar assinaturas de ataques. A maioria limita-se à análise de apenas um conjunto pré-estabelecido de protocolos e permite monitorar somente as camadas de rede e transporte. Outra limitação é a impossibilidade de se correlacionar PDUs (*Protocol Data Units*), isto é, o conjunto de regras é aplicado individualmente para cada pacote recebido. Não há como informar que uma determinada troca de mensagens entre duas estações constitui comportamento anômalo. Somado a isso, a complexidade das notações existentes facilita a ocorrência de erros nas assinaturas e acaba por impedir que os gerentes de redes criem, rapidamente, suas próprias (para observar comportamentos específicos de suas redes).

Este artigo propõe a utilização de uma linguagem de alto nível na especificação de assinaturas de ataques e apresenta um sistema de detecção de intrusão, o E-Sentry+, que utiliza tal abordagem. A linguagem utilizada para a especificação de ataques é a PTSL (*Protocol Trace Specification Language*) [2], uma linguagem gráfica/textual que permite a definição de traços de protocolo. O trabalho está organizado da seguinte forma: a seção 2 apresenta trabalhos relacionados. Na seção 3 é apresentado o IDS E-Sentry+ e seus aspectos funcionais como arquitetura, método de especificação de assinaturas, alarmes e notificação dos ataques detectados. A seção 4 descreve os testes realizados com o sistema. Por fim, na seção 5 são apresentadas as conclusões e as observações finais.

2. Trabalhos Relacionados

A área da detecção de intrusão é uma área ainda imatura. Isso pode ser notado pela ausência de padronizações quanto às terminologias e às classificações utilizadas. A imaturidade da área também pode ser observada na afirmação de Northcutt em [1] quando menciona que, atualmente, cerca de 90 por cento das detecções realizadas pelos IDS são falsas positivas. Esse alto número de alarmes falsos demonstra a complexidade da área e a ausência de uma base de conhecimento que forneça condições aos projetistas de criar algoritmos de detecção mais eficazes. Nesta seção aborda-se os trabalhos relacionados com o IDS que será apresentado na seção 3. As classificações aqui utilizadas baseiam-se nas apresentadas por Campello em [3].

O Bro [4] é uma ferramenta desenvolvida no Laboratório Nacional de Lawrence Livermore. Seu foco de pesquisa reside na construção de um sistema de detecção de intrusão robusto, que suporte o tráfego de redes de alta velocidade e seja protegido de ataques ao próprio IDS. O Bro utiliza-se de uma linguagem própria para a especificação de ataques. Segundo Paxson [4], a linguagem do Bro é muito parecida com a linguagem C, embora não possua capacidade de realizar *loops*. Essa incapacidade é proposital, pois ajuda a evitar sobrecargas que poderiam ocasionar perda de pacotes.

O EMERALD (*Event Monitoring Enabling Responses to Anomalous Live Disturbances*) [3, 5] é a ferramenta mais recente desenvolvida pela SRI International. Essa ferramenta usa as técnicas de comportamento (anomalias) e conhecimento (assinatura) para tentar identificar ataques. Usa ainda um subsistema de assinaturas chamado P-BEST [6], baseado em métodos de inteligência artificial.

O NetSTAT [7, 8] foi produzido na Universidade da Califórnia em Santa Barbara. Ele usa o conceito de máquinas de estado para representar seqüências de eventos que, se observadas, refletem atividades não autorizadas. O NetSTAT é, de todos os IDSs abordados, o mais parecido com o E-Sentry+. A especificação de assinaturas, no entanto, limita-se a campos pré-determinados dos protocolos TCP e IP.

O Snort é, segundo Northcutt [1], o sistema de detecção de intrusão mais utilizado atualmente. Foi desenvolvido por Marty Roesch. É um IDS de rede considerado leve e seguro. No Brasil, o CAIS publica assinaturas de ataque para essa ferramenta. O Snort foi originalmente desenvolvido para Unix, mas, atualmente, possui uma versão para Windows NT. Para a especificação de ataques, o Snort possui uma linguagem própria. Essa linguagem, no entanto, não permite correlacionar mensagens na especificação de ataques, ou seja, o mesmo conjunto de regras é aplicado para cada pacote recebido.

3. E-Sentry+

O E-Sentry+ é um IDS de análise passiva, baseado em rede, que se utiliza de uma base de conhecimento para a especificação dos ataques. Possui arquitetura centralizada e realiza monitoração contínua. Foi desenvolvido utilizando a linguagem de programação Delphi e, atualmente, possui apenas uma versão para os sistemas operacionais da família Windows.

3.1 A Arquitetura

A arquitetura do E-Sentry+ é apresentada na figura 1. O Módulo de Captura de Pacotes (1) é responsável por colocar a placa de rede da estação em modo promíscuo e capturar os pacotes que passam pelo segmento. O Filtro de Pacotes (2) descarta os pacotes cujos endereços IP origem ou destino não pertençam à lista de estações que estão sendo monitoradas. A Máquina de Estados (3) recebe os pacotes que passaram pelo filtro e os analisa para verificar se correspondem a uma mensagem que indique troca de estados. Esse módulo lê as assinaturas do arquivo de assinaturas e grava o *log* dos traços que ocorreram no arquivo de *logs*. Nos Arquivos de Assinaturas (5) ficam as assinaturas dos ataques. Por último, é na Interface com o Usuário (6) que ocorre a interação entre o usuário e a ferramenta.

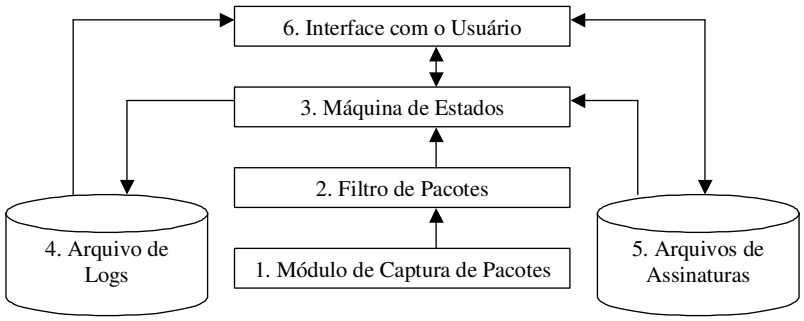

Figura 1: Arquitetura do E-Sentry+

3.2 A Linguagem PTSL

A linguagem PTSL, acrônimo de *Protocol Trace Specification Language*, foi proposta em [2] e é utilizada pelo sistema de detecção E-Sentry+ para a representação de assinaturas de ataque. Nesta seção a linguagem será apresentada e, para ajudar na compreensão, a técnica de varredura de portas TCP FIN, assim classificada por McClure em [9], é utilizada como exemplo.

Na varredura de portas TCP FIN são enviados vários pacotes com o flag FIN do cabeçalho do protocolo TCP ligado para portas diferentes em uma máquina alvo. A RFC 793 determina que um *reset* deve ser respondido para cada pacote enviado a uma porta fechada. Nesse tipo de varredura, portanto, o emissor envia vários pacotes: os que forem respondidos com *reset* correspondem a portas fechadas; os que não obtiverem resposta serão considerados portas abertas, pois o pacote terá sido ignorado pela máquina alvo.

A linguagem PTSL pode ser dividida em duas partes principais: a parte gráfica e a parte textual. A parte gráfica permite representar apenas parte de um traço, ao passo que, na textual, todo o traço pode ser representado.

A parte gráfica da linguagem PTSL utiliza-se de círculos para a representação dos estados; as transições, por sua vez, são representadas através de setas, que conectam os círculos entre si. Na figura 2 é apresentada a representação gráfica do traço Varredura TCP FIN. A máquina de estados sempre é representada do ponto de vista da estação cliente.

A seta contínua indica uma mensagem de uma estação cliente para o servidor; a pontilhada, do servidor para o cliente. Na especificação da Varredura TCP FIN, podemos notar dois estados distintos: *idle* e 2. A máquina mudará do estado *idle* para o estado 2 quando uma mensagem chamada “TCP

FIN" ocorrer; voltando ao estado *idle* com um TCP RST. Ao lado da máquina de estados é possível observar uma região onde algumas informações opcionais podem ser informadas.

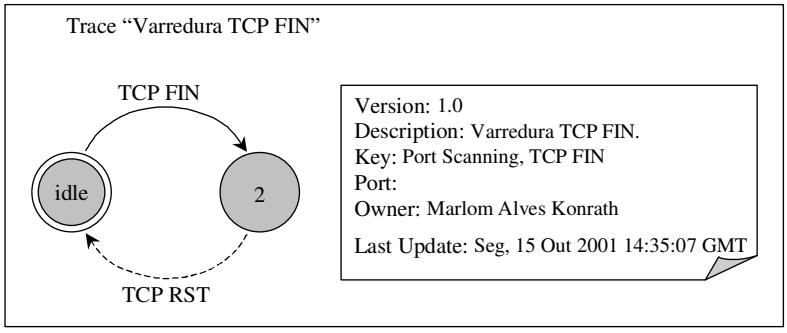

Figura 2: Representação gráfica da varredura TCP FIN em PTSL

A parte gráfica da linguagem, no entanto, não permite a descrição das causas que ocasionam as mudanças de estados. Apenas na parte textual da linguagem é possível especificar, tecnicamente, o que deve ocorrer para provocar uma transição de um estado para outro.

A parte textual da linguagem inicia com a palavra reservada TRACE e termina com a palavra ENDTRACE. A especificação pode ser dividida em quatro subgrupos, sendo eles: um cabeçalho com itens opcionais e informativos; a seção de mensagens; a seção de grupos (não utilizada no exemplo) e a seção de estados.

Primeiramente, é permitido ao gerente de rede especificar algumas informações opcionais, como versão, descrição, palavras-chave, porta e autor do traço, por exemplo. Logo após, vem a seção de mensagens, onde são nomeadas as mensagens que provocam as mudanças de estado. Ela é iniciada com a palavra reservada MESSAGESSECTION e terminada com ENDMESSAGESSECTION. Cada uma das mensagens é nomeada, iniciando com a palavra MESSAGE e terminando com ENDMESSAGE. Nessa seção são caracterizadas as mensagens que devem ocorrer para causar uma mudança na máquina de estados. Na mensagem TCP RST, por exemplo, espera-se receber um pacote com encapsulamento *Ethernet/IP* comparando-se um bit a partir do centésimo nono e que esse esteja ligado.

Em seguida à seção de mensagens, é especificada a seção de estados. A seção de estados começa na ocorrência da palavra reservada STATESSECTION e termina com ENDSTATESSECTION. Cada estado inicia com STATE e termina com ENDSTATE. A palavra reservada FINALSTATE determina o estado final do traço. Inicialmente, a máquina de estados estará no estado *idle*. Caso uma mensagem TCP FIN ocorra, a máquina irá para o estado 2. Da mesma forma, se o traço estiver no estado 2 e receber um TCP RST, retornará ao estado *idle*, quando então o traço terminará.

É importante salientar que a linguagem PTSL é bem mais abrangente. Além da seção de estados e mensagens, existe a seção de grupos. A comparação, além de binária (*BitCounter*), também pode ser por campo (*FieldCounter*) ou simplesmente a procura de uma string independente de sua posição (*NoOffSet*). O encapsulamento também inclui vários outros tipos, como *Ethernet/IP/TCP* ou *Ethernet/IP/UDP*. Uma explicação mais detalhada da linguagem pode ser obtida em [2].

```
TRACE "Varredura TCP FIN"
VERSION: 1.0
DESCRIPTION: Varredura TCP FIN
KEY: Port Scanning, TCP FIN
PORT:
OWNER: MARLOM ALVES KONRATH
LAST UPDATE: Seg, 15 Out 2001 14:35:07 GMT

MESSAGESSECTION
MESSAGE "TCP FIN"
Ethernet/IP BitCounter 111 1 1 "Flag FIN ligado"
ENDMESSAGE

MESSAGE "TCP RST"
Ethernet/IP BitCounter 109 1 1 "Flag RST (Reset) ligado"
ENDMESSAGE
ENDMESSAGESSECTION

STATESSECTION
FINALSTATE: idle
STATE idle
"TCP FIN" GOTOSTATE 2
ENDSTATE

STATE 2
"TCP RST" GOTOSTATE idle
ENDSTATE
ENDSTATESSECTION

ENDTRACE
```

Figura 3: Representação textual da varredura TCP FIN em PTSL

3.3 Interface com o Usuário

A figura 4 ilustra a interface principal do E-Sentry+. A ferramenta possui quatro seções distintas: em Ataques monitorados é que estão as assinaturas de ataque e é onde estas podem ser modificadas. Em *Hosts* Monitorados podem são informados os endereços IP das máquinas que serão monitoradas. Já em Configurações ficam localizadas as opções relacionadas a respostas que o E-Sentry+ dará ao detectar a ocorrência de um traço. Por fim, em Alertas são apresentados os alertas gerados pelo sistema.

Figura 4: Interface principal do E-Sentry+

Um traço pode ser criado utilizando a notação textual de PTSL ou através da utilização dos recursos disponíveis no E-Sentry+. Durante a criação ou edição de um traço, a janela ilustrada na figura 5 é exibida. Ao desenhar as setas que ligam os estados, o usuário é levado para outras interfaces, como a ilustrada na figura 6, para que possa especificar a mensagem que provocará a transição de estados.

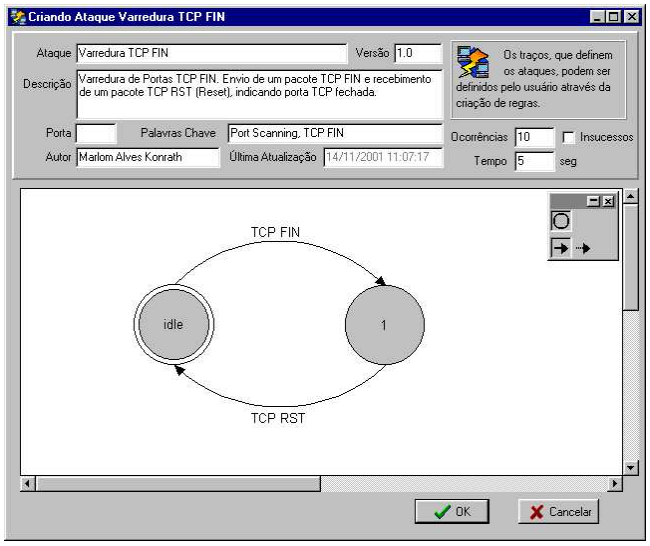

Figura 5: Especificação do ataque Varredura TCP FIN

A janela mostrada na figura 6 apresenta os campos de um cabeçalho TCP/IP, no entanto, outros cabeçalhos podem ser utilizados, bastando para isso que o usuário selecione a opção desejada. Quando os campos do encapsulamento escolhido pelo usuário aparecerem, basta selecionar um deles e informar o valor que este deverá conter.

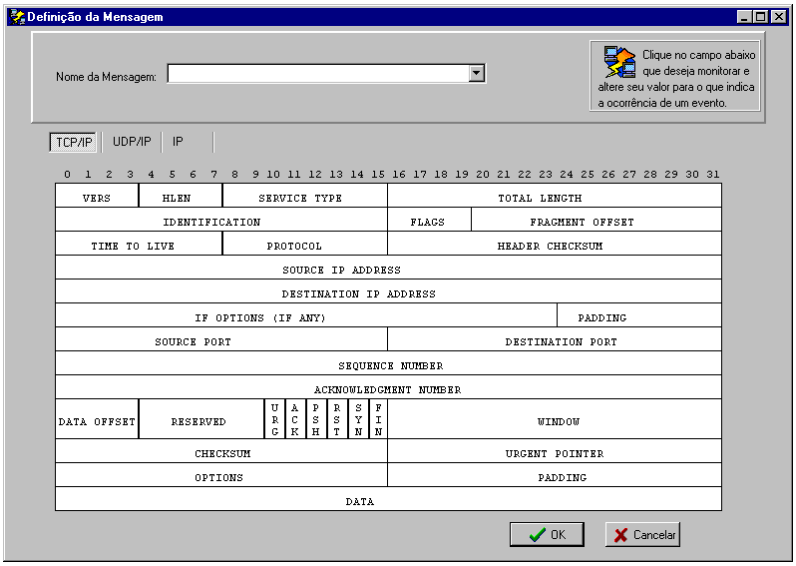

Figura 6: Definição das Mensagens no E-Sentry+

3.4 Visualizando os Resultados

Durante a fase de monitoração, os alertas gerados pela ferramenta aparecerão na janela de Alertas, conforme pode ser observado na figura 8. Esses alertas também ficarão registrados no *log* (figura 7).

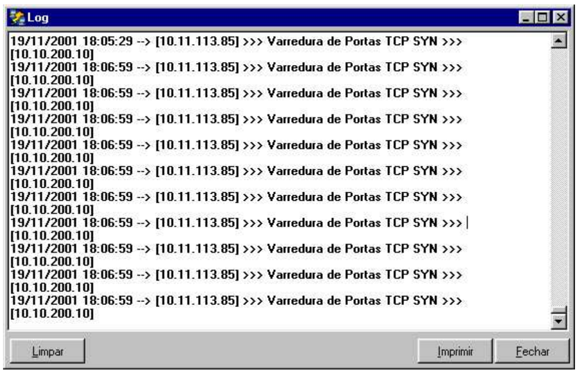

Figura 7: Visualizando *logs* no E-Sentry+

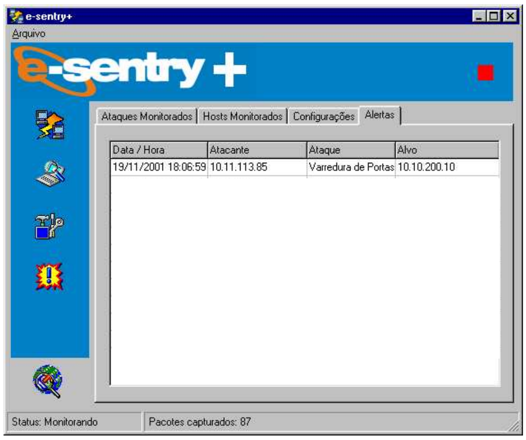

Figura 8: Janela de alertas do E-Sentry+

4. Validação do IDS

Para a validação do protótipo, dois ambientes foram utilizados. No primeiro caso, duas máquinas em uma rede com *switches* foram utilizadas. Já no segundo ambiente, três máquinas em uma rede com *hubs* participaram dos testes. A configuração das máquinas é apresentada na tabela 5.1 e a as ferramentas são listadas na tabela 5.2.

Tabela 1 – Descrição das máquinas que participaram do testes

Nome	Endereço IP	Processador	Memória	Sistema Operacional
A	10.11.113.92	Pentium 120	32 MB	Linux 6.2 (kernel 2.2.19-6.2.10)
B	10.11.113.85	Pentium III 800	128 MB	Windows 2000 Professional (SP2) Windows 95 C
C	10.11.113.243	Pentium 150	64 MB	Linux 6.2 (kernel 2.2.19-6.2.10)

Tabela 2 – Descrição das ferramentas utilizadas para os ataques

Nome	Versão	Uso
nmap	2.30 BETA 14	Ataques de varreduras de portas SYN e FIN.
synk4		ataques de inundação de SYNs.
PortScan	1.2 Basic	ataques TCP SYN partindo de máquinas Windows.

No primeiro cenário foram utilizadas duas máquinas: B e C. O *switch* possui a característica de enviar os pacotes apenas para as portas que estes foram endereçados. Por isso, a colocação de um sensor em uma porta do *switch* não teria efeito, pois ele só veria os pacotes endereçados a ele. Uma alternativa seria realizar um espelhamento de porta. No entanto, optou-se por utilizar a máquina B ao mesmo tempo como a máquina sendo atacada e máquina onde estava o sensor (E-Sentry+). As ferramentas utilizadas neste teste foram o nmap e o synk4.

Na rede utilizando *hubs* as três máquinas foram utilizadas. A máquina A gerou os ataques, utilizando-se das ferramentas nmap e synk4, na máquina B foi instalado e configurado o E-Sentry+ e a máquina C foi a que recebeu os ataques.

No primeiro teste com o nmap foram realizadas as varreduras TCP SYN e TCP FIN. Na varredura TCP SYN o E-Sentry+ detectou o ataque, mas na varredura TCP FIN isso não aconteceu. A RFC 793 determina que, quando um pacote for enviado para uma porta fechada a máquina deve responder com um *reset*. Entretanto, Mcclure afirma que a varredura TCP FIN funciona especialmente bem para máquinas utilizando o sistema operacional Unix ou seus descendentes, o que nos leva a crer que o Windows talvez não implemente a especificação conforme a RFC. Nos ataques realizados com o

synk4, em duas ocasiões, nas quais o ataque foi realizado por um período maior, a máquina que estava sendo atacada (B) ficou completamente desestabilizada e foi necessário reiniciá-la pressionando o botão de *Reset*.

Já no segundo teste, as três máquinas e as três ferramentas foram utilizadas. No ataque realizado com o PortScan, o E-Sentry+ detectou o ataque todas as vezes. No ataque realizado com o nmap, a ferramenta detectou tanto a varredura TCP SYN, como a varredura TCP FIN. Neste segundo teste, a máquina que estava sendo atacada era uma máquina Linux, o que só reforça a afirmação de Mcclure. Por último, o ataque utilizando o synk4 também foi reconhecido pelo E-Sentry+ todas as vezes.

5. Conclusões

O presente artigo propôs a utilização de uma linguagem de alto nível na especificação de assinaturas de ataques e apresentou uma ferramenta, chamada E-Sentry+, que se utiliza de tal abordagem. A utilização de uma interface gráfica, como a ilustrada na figura 5, torna a tarefa de especificação e o seu entendimento muito mais fáceis. Acredita-se ainda que ela propicie a diminuição da quantidade de erros em assinaturas.

No entanto, como a linguagem foi projetada para ser utilizada na área de gerência de redes não são permitidas algumas comparações necessárias para o registro de determinados ataques. Nesse caso, a proposição de uma extensão da linguagem seria necessária; cabe salientar, todavia, que a grande diversidade das técnicas utilizadas na realização de ataques torna difícil a especificação de uma única linguagem capaz de abranger todas as possibilidades.

Como trabalhos futuros destaca-se: (a) a realização de testes mais detalhados para que se possa verificar a escalabilidade e desempenho do IDS; (b) a ampliação da capacidade do E-Sentry+ de detectar ataques através da proposição de uma extensão da linguagem PTSL para suportar a análise de determinados pontos que a linguagem não contempla; (c) portar o E-Sentry+ para outras plataformas.

Referências

[1] NORTHCUTT, Stephen. **Segurança e prevenção em redes**. São Paulo: Berkeley Brasil, 2001.

[2] GASPARY, L. P.; BALBINOT, L. F.; STORCH, R.; WENDT, F.; TAROUCO, L. R. **Uma Arquitetura para Gerenciamento Distribuído e Flexível de Protocolos de alto Nível e Serviços de Rede** In SIMPÓSIO BRASILERIO DE REDES DE COMPUTADORES, (SBRC), Santa Catarina. Anais... Santa Catarina: SBC, 2001.

[3] CAMPELLO, Rafael Saldanha. **Sistema de Detecção de Intrusão.** In: SIMPÓSIO BRASILERIO DE REDES DE COMPUTADORES, (SBRC), Santa Catarina. Livro de Mini Cursos... Santa Catarina: SBC, 2001.

[4] PAXSON, V. **Bro: A System for Detecting Network Intruders in Real-Time**. Proceedings of the 7th USENIX Security Symposium, San Antonio, TX, Janeiro 1998.

[5] SRI International. **Event Monitoring Enabling Responses to Anomalous Live Disturbances (EMERALD).** Disponível em http://www.sdl.sri.com/projects/emerald/. (Novembro, 2001).

[6] LINDQVIST, Ulf; PORRAS, Phillip A. **Detecting Computer and Network Misuse Through the Production-Based Expert System Tollset (P-BEST)**. Proceedings of the 1999 IEEE Symposium on Security and Privacy, Oakland, California, May 9–12, 1999.

[7] ECKMANN, S.T.; VIGNA, G.; KEMMERER, R.A. **STATL: An Attack Language for State-based Intrusion Detection.** Proceedings of the ACM Workshop on Intrusion Detection, Athens, Greece, November 2000.

[8] VIGNA, G.; ECKMANN, S.T.; KEMMERER, R.A. **STATL: An Attack Language for State-based Intrusion Detection.** Journal of Computer Security, 2001.

[9] MCCLURE, Stuart et al. **Hackers Expostos**. São Paulo: Makron Books, 2000.