4. SISTEMAS DE DETECÇÃO DE INTRUSÃO

Esta seção apresenta os conceitos relacionados com as soluções de Detecção de Intrusos, que funcionam como um mecanismo de defesa adicional na proteção da segurança da informação nas organizações. Serão apresentados os principais tipos de sistemas IDS e opções de ferramenta para cada tipo, com foco especial na ferramenta Suricata, que será utilizada no laboratório prático deste curso. Por fim, aborda-se em linhas gerais o gerenciamento de regras de assinaturas de ataques.

4.1 Visão geral sobre IDS

A complexidade dos protocolos de rede, a evolução na complexidade dos ataques e o aumento significativo nas atividades cibernéticas maliciosas têm preocupado as organizações em termos de defesas tecnológicas (Mahurin, 2017). Uma das proteções mais utilizadas para segurança de redes e sistemas é a segmentação da rede em zonas (ex: interna, internet e DMZ) e a instalação de soluções de proteção de perímetro como Firewalls. Considerando que nenhum sistema é 100% seguro, existe um risco de que alguns ataques não sejam bloqueados pela solução de perímetro, portanto fazem-se necessárias outras barreiras de proteção.

Desta forma, tecnologias como Sistemas de Detecção de Intrusos (IDS) tem emergido como mecanismo adicional de segurança contra diferentes ataques: proteção contra *malwares* modernos, intrusão de redes, ameaças avançadas persistentes e controle de aplicações (Stuart & Beaver, 2013). Além de características clássicas como análise de assinaturas e análise de comportamento da rede e das aplicações, os sistemas de IDS modernos são caracterizados por incorporarem mecanismos de Inteligência contra Ameaças (*Threat Intelligence*) para bloqueio automático de URL/DNS/IP em sites de baixa reputação, inspeção profunda de pacotes, análise de aplicações e arquivos, etc.

4.2 Tipos de IDS

A maioria dos sistemas IDS utiliza um modelo baseado na captura de pacotes, normalização do tráfego, pré-processamento de aplicações e análise pelo motor de inspeção. O motor de inspeção, por sua vez, compara o tráfego observado com um conjunto de assinaturas

pré-definidas para aquele tipo de aplicação e verifica se o comportamento está em conformidade com o desejado. Outros motores de inspeção fazem a comparação com um perfil de tráfego previamente mapeado (análise baseada em comportamento). Através desse monitoramento constante do tráfego, pode-se tomar uma ação caso alguma atividade suspeita seja detectada. Esta ação pode ser desde um alerta ao administrador até o bloqueio temporário ou permanente do atacante. Em relação às informações coletadas, monitoramento, avaliação e tomada de decisões, os IDS podem ser classificados da seguinte forma:

- **IDS de rede**: sistemas de detecção de intrusão baseados em rede (NIDS) coletam informações através do tráfego de rede da organização, entre seus diferentes segmentos (seja o tráfego norte/sul - entrada e saída da internet - seja o tráfego leste/oeste - comunicação entre VLANs internas). O IDS de rede checa os ataques ou "comportamentos maliciosos" através da inspeção dos cabeçalhos (headers) e do conteúdo (payload) dos pacotes. A checagem ocorre através da correspondência com "assinaturas de ataque" que são regras que descrevem o padrão de tráfego associado a um comportamento malicioso específico. Geralmente, a maioria das soluções de NIDS oferece a opção de definir regras customizadas com assinatura de ataques ou comportamentos particulares da organização, por exemplo: uma regra customizada que bloqueie o acesso a sites de *phishing* dirigidos aos usuários da organização, ou ainda uma regra que detecta tentativas de intrusão em uma aplicação web da organização que possui uma vulnerabilidade específica. Cabe, portanto, ao NIDS comparar as assinaturas de ataque com os pacotes capturados para identificar tráfego hostil. A inspeção de tráfego pode ocorrer de duas maneiras: i) espelhamento de tráfego para o sistema NIDS ou ii) posicionando o equipamento NIDS *inline* nos segmentos de rede que se deseja monitorar. Cada abordagem possui vantagens e desvantagens, por exemplo: em ambientes com espelhamento o desempenho da rede não é prejudicado pela sobrecarga de inspeção de pacotes, por outro lado, ataques que consistem no envio de mensagens especialmente criadas para colapsar a aplicação não são barradas até que o NIDS execute a ação de contenção. Exemplos de NIDS são: Suricata[11], Snort [12] e Bro IDS[13];

[11] https://suricata-ids.org/
[12] https://www.snort.org/
[13] https://www.bro.org/

- **IDS de host**: sistemas de detecção de intrusão baseados em host (HIDS) atuam coletando informações e atividades de segurança em um sistema específico, geralmente através de um agente instalado no sistema. O termo “host” se refere à um computador individual, de forma que faz-se necessário um sensor para cada sistema que será monitorado. Um HIDS coleta dados a partir de eventos que ocorrem no sistema operacional, processos executando, arquivos, portas de rede e, principalmente, trilhas de auditoria (logs). Algumas vantagens dos HIDS em relação a NIDS incluem: capacidade de auditar e correlacionar eventos internos do sistema; de analisar tráfego que a nível de rede poderia estar criptografado; e de mapear os eventos sendo executados com usuários do sistema, a fim de identificar intrusos internos (insiders). Exemplos de HIDS são OSSEC [14]e Tripware[15];
- **IDS baseado em anomalia**: o IDS baseado em anomalia é um tipo especial do NIDS em que, invés de analisar o tráfego de rede em comparação com “assinaturas”, o sistema mapeia o perfil de rede considerado “normal” e emprega técnicas como redes neurais, aprendizado de máquina, modelos estatísticos, dentre outros, para identificar um tráfego anômalo. Faz-se necessário, portanto, uma fase de monitoramento e aprendizagem para traçar o perfil de acesso “normal” para que o sistema passe a identificar o comportamento malicioso. Uma vantagem desse tipo de IDS é sua capacidade de identificar ataques “zero-day”, ou seja, ataques que não possuem qualquer tipo de informação antes de sua exploração. Não obstante, uma desvantagem é a geração de um grande número de alarmes falsos devido ao comportamento imprevisível de usuários e do próprio sistema. Um exemplo de solução de IDS baseado em anomalia é o Hogzilla[16], além de inúmeros outros projetos acadêmicos;
- **IDS de kernel**: nesse tipo de IDS a detecção de comportamento malicioso é feita através da análise de chamadas de sistema, bibliotecas do S.O., movimentação de registradores, enfim, de um conjunto de rotinas internas ao kernel do sistema. Embora pouco comuns, esse tipo de IDS vem ganhando atenção recentemente com os ataques de Ransomware, no qual os arquivos do usuário são criptografados e o atacante solicita resgate. Como a execução de funções criptográficas são legítimas no sistema, uma análise mais aprofundada a nível do kernel se faz necessária para identificar tal

[14] https://ossec.github.io/
[15] https://github.com/Tripwire/tripwire-open-source
[16] http://ids-hogzilla.org/

atividade maliciosa. Soluções comerciais como Checkpoint[17], Kaspersky[18], dentre outras, possuem tal funcionalidade. Algumas ferramentas *open source* como SystemTap[19] e strace[20] também se encaixam nessa categoria;

- **NGIDS**: esse é um tipo de IDS considerado de "nova geração", cujas funcionalidades englobam as outras categorias previamente relacionadas. Além disso, tem capacidade de tratar ameaças avançadas persistentes (APTs), possuem correlação de eventos (SIEM) e aplicam técnicas de ciber inteligência contra ameaças (CTI). Ademais, o esforço de configuração nessas ferramentas é reduzido se comparado às outras abordagens, posto que os fabricantes empregam recomendações de configuração que são mais comumente observadas em organizações do mesmo setor. Existem diversos exemplos de soluções comerciais que se enquadram nessa categoria. O leitor pode obter mais informações consultando o quadrante de IPS do Gartner.

Atualmente, são encontradas diferentes soluções de IDS/IPS disponíveis, sendo elas comerciais, software livre e acadêmicas. Muitos fabricantes têm investido em produtos que minimizem a necessidade de customizações e otimizem a capacidade de proteção da solução. Em termos comerciais, o Gartnet mantém uma avaliação das soluções de IDS/IPS[21], que pode ser usada pelas organizações como *baseline* de escolha. Existem também algumas soluções abertas, que são comumente utilizadas e efetivas para as organizações. Entre as principais soluções de IDP/IPS *open source* encontram-se: i) Snort - um NIDS open source, bastante conhecido, baseado em assinaturas e com estrutura modular altamente customizável, de modo que diversos plugins podem ser usados para expandir suas funcionalidades (ex: reagir a um alerta, a atualização automática das suas assinaturas e o gerenciamento de diversos sensores espalhados em uma ou mais redes); ii) Suricata - um NIDS baseado em assinatura, derivado do Snort, cujo objetivo é se tornar um padrão para ambientes de alta demanda de tráfego e desempenho; e iii) Bro - um NIDS que monitora de forma passiva o tráfego de rede em busca de atividades suspeitas, tanto com base em assinaturas quanto em termos de eventos e atividades incomuns (por exemplo, certo host conectar a determinados serviços ou falhas em

[17] https://www.checkpoint.com/
[18] https://www.kaspersky.com.br/
[19] https://sourceware.org/systemtap/
[20] https://strace.io
[21] https://www.gartner.com/reviews/market/intrusion-prevention-systems

tentativas de conexão). Neste curso será utilizada a ferramenta Suricata, que tem apresentado bom desempenho no processamento de grande volume de tráfego.

O Suricata foi desenvolvido pelo OISF (*Open Information Security Foundation*) em 2010 e, apesar de menos difundido que o Snort, por exemplo, possui uma capacidade de processamento de tráfego na ordem de 10 Gbps e um motor capaz de executar funções avançadas de detecção de intrusos. Considerado um IDS de "Nova Geração", as funcionalidades avançadas do Suricata incluem: suporte a multi-threading; suporte a detecção automática de diversos protocolos (ex: IP, TCP, UDP, ICMP, HTTP, TLS, FTP, SMB e tantos outros da camada de aplicação), independente da porta; suporte a descompressão Gzip; biblioteca HTB independente para processamento de tráfego HTTP (mesma biblioteca utilizada pelo Modsecurity - uma importante solução de Firewall Web); consulta e atualização de listas de reputação IP; dentre outros.

4.3. Gerenciamento de Regras do IDS

O IDS funciona com base em assinaturas de ataques, geralmente de forma compatível com as regras utilizadas pelo projeto Snort. Isso permite a utilização de regras desenvolvidas pela comunidade ou por empresas especializadas, além da definição de regras customizadas para a organização, a fim de identificar atividades suspeitas específicas do ambiente. A forma de definição de uma regra genérica é mostrada na Figura 4.1: cada regra é composta por uma ação (em vermelho), cabeçalhos do pacote que definem qual tráfego analisar (verde) e opções de casamento (em azul) que permitem fazer buscas também dentro do pacote. As ações que são aplicadas ao tráfego quando uma regra é acionada podem ser *block*, *alert* e *pass*.

```
alert tcp 10.0.0.31 any -> any 80
(msg:"HTTP Google Access";
content:"google.com"; http_uri;
classtype:web-access; sid:1; rev:1;)
```

Figura 4.1 - Formato de definição de regras do Suricata

É comum o agrupamento das regras do IDS em categorias, a fim de facilitar a busca e resolução de problemas, além de especializar o conjunto de métodos de monitoramento que está relacionado a cada contexto. Uma forma de organizar as regras de assinatura é

baseando-se em categorias como: Sistema Operacional (Windows, Linux, Solaris, etc); protocolos (TCP, UDP, TFTP, RPC, POP, IMAP, etc); Servidores (IIS, Apache, Tomcat, MSSQL, MySQL, Oracle, Samba, DNS, etc); entre outras. Dessa forma o administrador pode facilmente escolher quais regras serão ou não habilitadas para cada subrede de acordo com suas características individuais.

As regras de assinaturas de ataques desempenham um papel fundamental no processamento do Suricata. Através dessas regras é possível identificar ataques que são comuns às diferentes organizações. Existem diversos conjuntos de regras disponíveis para integração à ferramenta de IDS, alguns de acesso livre e outros com restrições comerciais. A seguir uma descrição sobre os conjuntos de regras mais utilizados:

- Regras Snort Community[22] - conjunto de regras com acesso aberto e mantidas pela comunidade Snort;
- Regras Snort Subscriber[23] - conjunto de regras VRT / Talos (Vulnerability Research Team) , mantida pela Sourcefire/Cisco, em que os clientes registrados conseguem acesso às regras com antecedência de até 30 dias em comparação com a versão Community;
- Regras do Emerging Threats[24] - regras comunitárias e gratuitas, desenvolvidas por voluntários. Apesar de gratuitas, as regras ET são muito eficientes e possuem um elevado índice de atualizações;
- Regras do Emerging Threats Pro[25] - versão comercial do ET mantida pela empresa Proofpoint, oferecendo maior grau de precisão e rapidez na distribuição de assinaturas de novos ataques.

Essas regras são mantidas pela comunidade e por empresas como Talos/Cisco, Proofpoint, entre outras, elas são atualizadas frequentemente e fornecem uma boa fonte de assinaturas para ataques conhecidos. Uma das formas de instalar e manter as regras supracitadas é através de ferramentas de Gerência de Regras como o software Oinkmaster[26]. Através da ferramenta Oinkmaster o administrador pode automaticamente ativar ou desativar um conjunto

[22] https://www.snort.org/downloads#rules
[23] https://www.snort.org/products#rule_subscriptions
[24] http://doc.emergingthreats.net/bin/view/Main/AllRulesets
[25] https://www.proofpoint.com/us/threat-insight/et-pro-ruleset
[26] http://oinkmaster.sourceforge.net/

de regras, para evitar falsos positivos, gerenciar assinaturas de conjuntos de regras ou ainda utilizar diferentes fontes de obtenção das assinaturas.

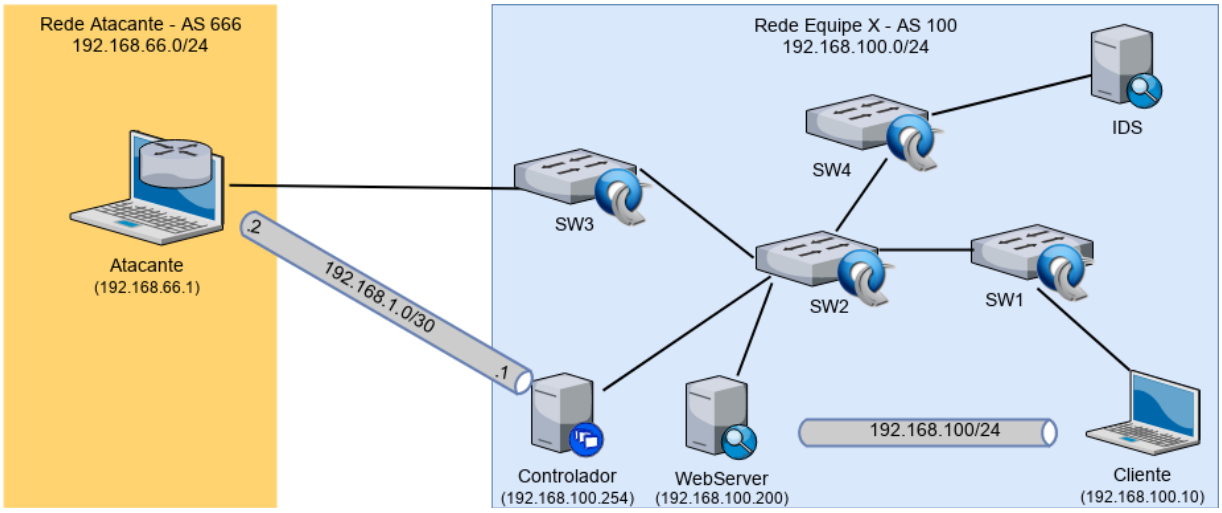
Por fim, as regras são um elemento-chave na configuração do IDS. Um conjunto de regras bem estabelecido pode ser o diferencial para uma maior capacidade de identificação de ataques. Por outro lado, muitas regras podem gerar falsos positivos ou ainda violar a privacidade dos usuários (registrando logs além do necessário), então fazer um ajuste das regras é uma tarefa cansativa, porém indispensável para que os registros de alerta sejam confiáveis e úteis. Um IDS que gera muitos alertas falsos facilmente acaba em desuso e um IDS sem regras atualizadas e relevantes ao perfil de tráfego traz baixo benefício para a organização.

4.4 Exercícios de Fixação

1. O que é um IDS e como ele funciona?
2. Quais os tipos de IDS?
3. Quais os riscos inerentes ao gerenciamento de assinaturas de IDS?
4. Considerando o cenário de rede baseada em SDN proposta neste laboratório, como viabilizar o encaminhamento do tráfego para o servidor IDS?

4.5 Roteiro de Laboratório

O objetivo deste laboratório é instalar e configurar um sistema de detecção de intrusão para monitoramento do tráfego interno e externo do AS 100 através da ferramenta Suricata IDS. Para isso, além da instalação do Suricata, deve-se habilitar o espelhamento de tráfego no switch Openflow para o Servidor IDS. A topologia utilizada neste laboratório é a mesma anterior, conforme Figura 4.2.



4.2. Topologia proposta para os experimentos da Oficina.

4.5.1 Instalação e configuração do Suricata

1) A instalação do Suricata será realizada a partir dos repositórios Debian, evitando a necessidade de compilar seu código fonte. Para isso, utilizaremos a versão "backport" do Debian Jessie, que fornece o Suricata na versão 3.2. Antes de instalar será necessário atualizar a lista de pacotes. Para isso acesse a máquina IDS e execute:

```
su -
echo "deb http://debian.pop-sc.rnp.br/debian/ jessie-backports main contrib non-free" >>
/etc/apt/sources.list
apt-get update
```

2) Em seguida deve-se proceder com a instalação do Suricata:

```
apt-get -t jessie-backports install suricata tcpdump curl python-ipcalc
```

3) Após instalar o suricata o próximo passo é obter as regras que serão utilizadas. Neste laboratório vamos trabalhar com as regras community da Emerging Threats. Para obtê-las, basta executar o seguinte comando:

```
/usr/sbin/suricata-oinkmaster-updater
```

4) Além das regras do Emerging Threats, vamos criar uma regra customizada para identificar ataques de Negação de Serviço (DoS) baseado em TCP Syn Flood para nossa organização. Observe que esse tipo de regra é baseada em limiares (threshold), portanto sua correta configuração depende de um estudo prévio do perfil de tráfego da organização para identificar o limiar mais adequado. Para fins ilustrativos, consideraremos que um limiar de 40 pacotes/seg de TCP-SYN de uma mesma origem externa é considerado ataque. Para isso, crie o arquivo **/etc/suricata/rules/local.rules** com o seguinte conteúdo:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET $HTTP_PORTS (msg:"Possible TCP Syn Flood DoS"; flags:S; flow:to_server; threshold: type both, track by_src, count 40, seconds 5; classtype:attempted-dos; sid:10001; rev:1;)
```

5) Em seguida vamos realizar a configuração do servidor Suricata, editando o arquivo **/etc/suricata/suricata.yaml**. O primeiro passo na edição desta configuração é definir o endereçamento de rede da organização (HOME_NET). Essa definição é importante para que o IDS possa diferenciar o sentido do tráfego (acessos internos *versus* acessos externos, etc). A configuração deverá ficar da seguinte forma:

```
vars:
  address-groups:
    HOME_NET: "[192.168.100.0/24]"
...
```

6) O próximo passo, ainda no arquivo **/etc/suricata/suricata.yaml**, é selecionar quais regras deseja ativar ou desativar. Vamos comentar algumas linhas (adicionando o caracter # no início), descomentar outras (retirando o caracter # do início) e adicionar a regra criada. Ao final das mudanças a configuração deve ficar da seguinte forma:

```
default-rule-path: /etc/suricata/rules
rule-files:
 - botcc.rules
 - emerging-scan.rules
 - local.rules
```

7) O passo seguinte, ainda no arquivo **/etc/suricata/suricata.yaml**, é configurar as opções de captura de pacotes do sistema IDS. Para isso edite o arquivo substituindo a interface eth0 pela eth1 (outro parâmetros não mencionados devem ser mantidos com seus valores originais):

```
af-packet:
  - interface: eth1
...
pcap:
  - interface: eth1
```

8) O próximo passo da configuração é ativar a interface de rede em que o tráfego será capturado. Para isso, edite o arquivo **/etc/network/interfaces**, e altere os seguintes parâmetros:

```
auto eth1
iface eth1 inet manual
      pre-up ifconfig $IFACE up
      post-down ifconfig $IFACE down
```

9) É preciso ativar o serviço do IDS suricata durante a inicialização do sistema. Para isso, edite o arquivo **/etc/default/suricata** e altere os seguintes parâmetros:

```
RUN=yes
IFACE=eth1
```

10) Devido a um problema nessa versão do Suricata, um passo adicional precisará ser realizado:

```
sed -i '$i /etc/init.d/suricata restart' /etc/rc.local
```

11) Finalizada essa configuração inicial, podemos iniciar a interface de captura e iniciar o daemon do Suricata para que ele monitore o tráfego de rede. Para isso, execute:

```
ifup eth1
```

```
/etc/init.d/suricata restart
```

4.5.2 Ativando o espelhamento de tráfego

Para que o sistema IDS possa monitorar o tráfego de rede que está passando na organização, é necessário que esse tráfego seja “espelhado” para porta do Suricata. Essa configuração não seria necessária se o IDS fosse implantado no modo “inline” (onde todo o tráfego da organização é forçado a passar pelo IDS), o que não o caso deste laboratório. Dessa forma vamos utilizar a aplicação SDN-IPS que vem sendo utilizada até aqui. Para que esse passo tenha sucesso você precisará identificar três informações: i) o datapath-id do switch em que o espelhamento será realizado (SW2 na Figura do laboratório); ii) a porta na qual o servidor IDS está conectado ao switch SW2 (posicione o mouse sobre o servidor IDS na interface do Ofelia OCF); e iii) o conjunto de regras que representam o fluxo que deseja-se espelhar (nesse laboratório todo o tráfego será espelhado). De posse destas informações, o espelhamento é então configurado.

1) Esta prática pressupõe que o roteiro anterior do capítulo tenha sido executado com sucesso. Portanto, caso tenha desligado o ambiente ao final da prática anterior, é necessário religar o controlador Ryu. Ele irá recarregar as configurações que foram realizadas anteriormente a partir do arquivo **`sdn-ips-config.json`** na mesma pasta da aplicação.

2) Antes de prosseguir será necessário identificar o DatapathID em que o Controlador está instalado, pois é desse switch que o tráfego será espelhado (origem do espelhamento). Para isso busque na topologia do OCF (*Physical topology*) o switch equivalente ao SW2 do desenho da prática. Um exemplo pode ser observado na Figura 4.3:

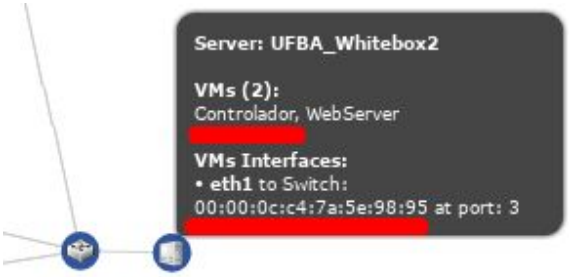


Figura 4.3 Exemplo de identificação do DatapathID do Controlador.

Na Figura 4.3 é possível observar que o servidor Controlador (no exemplo, alocado no Virt. Aggregate “UFBA_Whitebox2”) está conectado ao Switch Openflow **00:00:0c:c4:7a:5e:98:95**.

3) É necessário também identificar o “destino” do espelhamento através do DatapathID e da porta para onde o tráfego será espelhado, ou seja, onde o servidor IDS está conectado. Para isso busque na topologia do OCF (*Physical topology*) o switch equivalente ao SW4 do desenho da prática. Um exemplo pode ser observado na Figura 4.4:

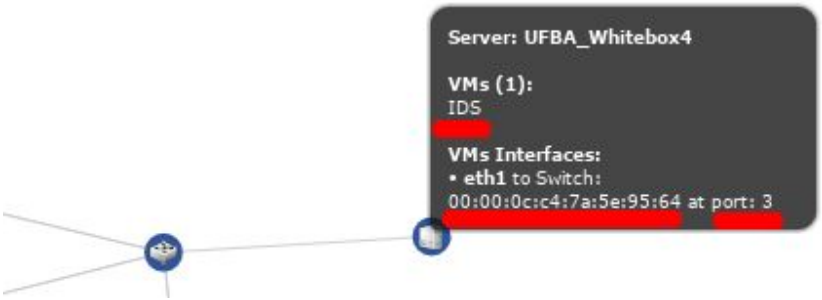


Figura 4.3 Exemplo de identificação do DatapathID do destino do espelhamento .

Na figura acima é possível observar que o servidor IDS (no exemplo, alocado no Virt. Aggregate “UFBA_Whitebox4”) está conectado ao Switch Openflow **00:00:0c:c4:7a:5e:95:64** e na **porta 3**.

4) Em seguida, na máquina Controlador, precisamos definir em qual tráfego esse espelhamento será aplicado. Em ambientes convencionais a configuração de qual tráfego será monitorado pode se dá baseado na VLAN, nos cabeçalhos TCP/IP, etc. Na aplicação SDN aqui utilizada, a definição sobre qual tráfego será monitorado se dá através dos fluxos instalados no switch, ou seja, você precisará dizer quais fluxos deseja espelhar o tráfego. Para listar os fluxos, utilize o seguinte comando substituindo o <DPID> (DatapathID) pelo valor obtido no passo anterior (2):

```
curl -s http://localhost:8080/sdnips/flows/<DPID> | python -m json.tool
```

O comando acima deve gerar como saída quatro fluxos: existem dois e-Line e para cada e-Line há dois fluxos, um para cada sentido. Como pode ser observado abaixo:

```
root@Controlador:~/fibre-2a-opencall-sdn-ips# curl -s http://localhost:8080/sdnips/flows/
00000cc47a5e9895 | python -m json.tool[
    {
        "match": {
            "dl_vlan": 3652,
            "in_port": 3
        },
        "priority": 65533
    },
    {
        "match": {
            "dl_vlan": 3652,
            "in_port": 4
        },
        "priority": 65533
    },
    {
        "match": {
            "dl_vlan": 3653,
            "in_port": 3
        },
        "priority": 65533
    },
    {
        "match": {
            "dl_vlan": 3653,
            "in_port": 1
        },
        "priority": 65533
    }
]
```

A partir do comando acima o administrador poderá selecionar qual fluxo monitorar. Neste laboratório vamos monitorar todo o tráfego, tanto da rede interna quanto da rede externa. Assim podemos usar a palavra chave “all” na escolha dos fluxos do espelhamento, conforme detalhado a seguir.

5) Agora, de fato, o espelhamento será criado a partir dos dados obtidos nos passos anteriores: DPID, porta e lista de fluxos (ou “all” para todos os fluxos). Assim execute o seguinte comando (não esqueça de alterar o <DPID-ORIG> (passo 2) e a <PORTA> e <DPID-DEST> (passo 3)):

```
curl -s -X POST -d '{"flows": "all", "to_target": "<DPID-DEST>:<PORTA>"}'
http://localhost:8080/sdnips/flows/<DPID-ORIG>/mirror | python -m json.tool
```

Exemplo:

```
root@Controlador:~# curl -s -X POST -d '{"flows": "all", "to_target": "00000cc47a5e9564:3"}'  http://localhost:8080/sdnips
/flows/00000cc47a5e9895/mirror | python -m json.tool
[
    "Success"
]
```

6) Para validar se o espelhamento está funcionando corretamente, será executado um teste de PING entre a máquina Controlador e a máquina Cliente, ao passo que a máquina IDS estará com sniffer de tráfego ligado para visualizar as mensagens. Para isso, através do console da máquina IDS, execute (o filtro mostra apenas pacotes icmp ou arp):

```
su
tcpdump -i eth1 -n icmp or arp
```

Agora, no console da máquina Controlador, execute:

```
ping -c 2 192.168.100.10
```

De volta ao console da máquina IDS, você deverá ver o seguinte:

```
root@IDS:~# tcpdump -i eth1 -n icmp or arp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 262144 bytes
00:16:20.242552 ARP, Request who-has 192.168.100.10 tell 192.168.100.254, length 42
00:16:20.242765 ARP, Reply 192.168.100.10 is-at 00:00:90:00:00:5f, length 42
00:16:20.242989 IP 192.168.100.254 > 192.168.100.10: ICMP echo request, id 1409, seq 1, length 64
00:16:20.243161 IP 192.168.100.10 > 192.168.100.254: ICMP echo reply, id 1409, seq 1, length 64
00:16:21.243555 IP 192.168.100.254 > 192.168.100.10: ICMP echo request, id 1409, seq 2, length 64
00:16:21.243701 IP 192.168.100.10 > 192.168.100.254: ICMP echo reply, id 1409, seq 2, length 64
```

4.5.3 Ferramentas de auditoria do IDS

Nesta prática vamos utilizar algumas ferramentas de auditoria de IDS para simular ataques na rede da organização e verificar se o sistema IDS consegue identificá-los. Aqui no laboratório serão utilizadas ferramentas como HPING, HYDRA e CURL. O leitor deve utilizar ferramentas como HPING e HYDRA com muito cuidado.

1) Na máquina Atacante, o primeiro passo é instalar essas ferramentas de auditoria. Para isso, acesse a máquina Atacante e execute os seguintes comandos:

```
su
apt-get update && apt-get install hping3 hydra
```

2) Agora na máquina WebServer vamos instalar o serviço de páginas web (servidor HTTP) para servir páginas externas e redirecionar os usuários infectados. Este servidor será utilizado como

alvo do teste de auditoria do HPING. Para isso, o primeiro passo é instalar o apache e depois criar o arquivo HTML que será servido. Utilize os seguintes comandos:

```
su
apt-get update && apt-get install apache2 tcpdump
cat >/var/www/html/index.html <<EOF
<h1>SDN-IPS: Sua maquina foi identificada como possivelmente infectada! Procure o departamento de TI da organizacao!</h1>
EOF
```

3) Na máquina Atacante, execute um ataque de syn-flood contra a máquina WebServer:

```
hping3 --fast -S -p 80 192.168.100.200
```

4) Na máquina Atacante, realize um ataque de brute-force SSH com hydra contra WebServer:

```
seq 1 300 > /tmp/pass.txt
hydra -l root -P /tmp/pass.txt ssh://192.168.100.200
```

5) Na máquina Cliente, vamos simular um acesso a IP malicioso de C&C (servidor de *Command and Control*). Para isso, acesse o site do Tracker do Trojan Feodo (ou qualquer outro da lista da emergint-threats botcc), mantido pelo time de segurança abuse.ch através do site https://feodotracker.abuse.ch/, escolha algum IP que esteja online de C&C para simular o acesso. A partir daí vamos rotear esse IP através da rede do AS100 e simular um acesso. Para isso execute o seguinte comando (atente-se para alterar o <IP-CnC> para o IP escolhido):

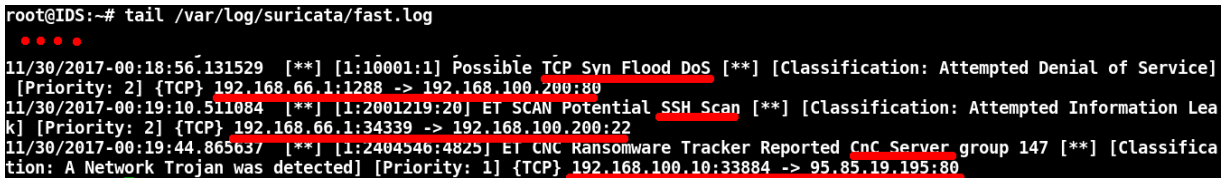
```
route add -host <IP-CnC> gw 192.168.100.254
wget http://<IP-CnC>/
```

OBS: apesar do comando acima não conseguir efetuar a conexão HTTP, afinal na rede deste laboratório não habilitamos o acesso à Internet, essa tentativa de acesso já é suficiente para identificar uma máquina possivelmente comprometida no IDS.

6) Por fim, na máquina IDS, observe os logs do Suricata (/var/log/suricata/fast.log) se as tentativas de intrusão foram corretamente identificadas:

```
tail /var/log/suricata/fast.log
```

A saída esperada é algo como:



Assim, no final deste experimento, observa-se que os ataques já estão sendo detectados pelo IDS. No entanto, nenhuma medida de prevenção está sendo tomada. Desta forma, na próxima seção serão criadas as medidas de prevenção de acordo com o tipo de ataque (interno ou externo).