ScanNet
Localizador de *Hosts* Vizinhos e Gerador Automático de Arquivos de Configuração

ARLINDO FOLLADOR NETO[1]
JOAQUIM QUINTEIRO UCHÔA[1]

[1]Curso de Especializacao em Administração em Redes Linux
Departamento de Ciência da Computação – Universidade Federal de Lavras (UFLA)
Caixa Postal 37 – 37.200-000 – Lavras – MG – Brasil
arlindo@lopan.eti.br, joukim@ginux.ufla.br

Resumo: *Este artigo apresenta o desenvolvimento de uma aplicação que auxilia no reconhecimento de redes locais e geração automática de arquivos de configuração. A solução proposta coletará de hosts vizinhos informações como: Nome do host, MAC e IP. Programado em linguagem de fácil assimilação, a aplicação pode ser rapidamente interpretada e alterada se necessário. Sendo modular, permite ao usuário personalizar atuais ou criar novas rotinas de geração de arquivos de configuração.*

Palavras-Chave: *ScanNet, Localizador Hosts, Gerador Automático.*

1 Introdução

Segurança é assunto que constantemente está sendo discutido e a sua ausência expõe empresas e usuários a danos irreparáveis. Atualmente, configurar serviços de modo que aceitem apenas os acessos necessários e neguem os demais, envolve muito trabalho manual. Além disso, tornar uma rede local segura exige do administrador a configuração e gerenciamento de recursos como: *Firewall*[1], *IDS*[2], AntiVírus[3], dentre outros (UCHÔA, 2005). Tudo isso para proteger do perigo de ataques externos, mas quando o perigo pode estar dentro da empresa? Constantemente, é descartada a possibilidade de alguém, sorrateiramente ter acesso a um ponto de conexão local.

Em geral, quanto maior a empresa, maior o número de computadores interconectados, pontos de conexão, servidores e maior o risco de haver um acesso local não desejável. Dessa forma, acessar serviços de Banco de Dados, *E-mail*, *Web*, dentre outros, não será tarefa difícil. Esta situação agrava-se ainda mais com a chegada das Redes *Wireless*, em que normalmente o ponto de acesso é promíscuo e como trata-se de rede local, não é configurado para lidar individualmente com os acessos.

Este trabalho apresenta o ScanNet, produzido para tornar simples o trabalho de reconhecimento de *hosts* em redes locais e geração automática de arquivos de configuração,

[1]Sistema para controlar acessos indevidos à rede, *Iptables*: http://www.netfilter.org/.
[2]Sistema de Detecção de Intrusos.
[3]Aplicativos que realizam checagem em arquivos para verificar a existência de vírus de computador.

de acordo com os módulos pré-estabelecidos. O objetivo principal é descomplicar e tornar rápida a geração de arquivos de configuração de serviços como: `/etc/hosts` ou *NIS* (KUKUK, 2003), *DNS* (LANGFELDT, 2001), *Iptables* (SILVA, 2005), *DHCP* (CONSORTIUM, 2004). Existe a possibilidade de explorar a aplicação desenvolvendo novos módulos para serem executados, gerando assim novos arquivos de configuração com base nas informações adquiridas na rede local.

2 Acesso à Rede na *Intranet*

O *DHCP*, é uma forma prática e descomplicada de fornecer elementos de configuração de rede à *hosts*, incluindo endereço *IP*, *DNS*, *Gateway*[4] e domínio *SMB*[5]. O *DHCP* quando configurado para fornecer endereços sem nenhum controle, põe em risco a rede local, tornando fácil a aquisição de um endereço *IP* válido localmente, mesmo por quem não deveria ter acesso. Dessa forma, qualquer acesso proveniente de um ponto de conexão na rede local, automaticamente estará apto a utilizar os recursos desta rede. A solução para limitar as concessões de endereços fornecidos pelo *DHCP*, seria obedecer à regra: *liberar o necessário e negar o restante*. Para fazer isso, é necessário fornecer *IPs* para cada endereço *MAC*[6] e rejeitar solicitações sem cadastro prévio.

Ajustado o serviço de *DHCP* da forma descrita, restaria ainda um outro problema. Este serviço apenas concede os endereços automaticamente, mas não impede que isto seja feito manualmente pelo usuário. Neste caso, o acesso aos serviços da rede interna, será exposto mesmo com o *DHCP* fornecendo endereços individualmente por *MAC*. É necessário então, não só atribuir endereços individualmente, mas também aceitar as conexões individualmente. Para triar o acesso aos servidores de uma rede local, podem ser utilizados recursos como: *TCP Wrappers*[7] ou *Firewall*. *TCP Wrappers* não traz tanta segurança, pois apega-se unicamente ao endereço *IP*. Uma solução mais consistente seria por meio de *Firewall*, verificando para cada acesso ao servidor, o *IP* e *MAC* de origem da conexão.

Dessa maneira, é interessante que fosse implementado uma aplicação simples, maleável e modular, que poupasse tempo na procura de informações de *hosts* na rede local e o esforço manual na geração de arquivos de configuração. Além disso, deve ser possível a correção de erros e implementação de novos recursos de geração de arquivos de configuração, utilizando-se das informações básicas:

- Endereço *IP*;
- Endereço *MAC*;
- Nome de *host*, utilizando resoluções básicas *DNS*, `/etc/hosts` e *SMB*.

Estas informações serão necessárias para criar arquivos que configuram serviços de redes como: `dhcpd.conf`, `/etc/hosts`, zonas de *DNS*, configuração de *Firewall*, entre

[4] Endereço *IP* de Roteamento para *Internet*.

[5] O protocolo *SMB* (*Session Message Block*) é usado para compartilhar arquivos e impressoras em redes *Windows*®(KIRCH; DAWSON, 2000).

[6] *Media Access Control*, endereço de 48 bits, expresso em doze dígitos hexadecimais, para identificar unicamente cada dispositivo de rede.

[7] Restringe acesso aos serviços que tem suporte à biblioteca libwrap.

outros. É desejável, portanto, uma aplicação que, com base nos *hosts* da rede local, crie automaticamente esses arquivos de configuração, ao invés do trabalho repetitivo e manual.

3 ScanNet: Solução Proposta

Esta seção apresenta o ScanNet, solução proposta para ajudar na tarefa de automatizar o reconhecimento de redes locais e geração de arquivos de configuração. A aplicação inicialmente faz o reconhecimento da rede local, buscando *hosts* ativos no intervalo de rede configurado, armazenando o endereço *MAC* de cada *host*, e, com base em funções externas, resolve os nomes através de *DNS*, `/etc/hosts` e *SMB*. Após reunir estas informações, são executados os módulos de geração de arquivos de configurações, que recebem as informações coletadas como parâmetros, para a sua construção correta.

Os módulos devem ser declarados no arquivo de configuração `scannet.conf`, para serem executados. Alguns módulos previamente desenvolvidos, dão um exemplo da facilidade de criação de novos módulos. Toda a aplicação é desenvolvida em *Shell Bash*[8] (FOUNDATION, 2002), tornando fácil e viável o entendimento e as alterações no código, visando atender necessidades futuras.

Nesse arquivo de configuração pode ainda ser configurado o comportamento do programa quanto a busca de nomes para os *IPs*, a ordem de prioridade nesta busca e as decisões caso não seja possível resolver o *IP* do *host* em um nome. Opcionalmente, a aplicação pode solicitar um nome ao administrador de forma interativa ou automaticamente, utilizando um nome-máscara (configurado previamente) seguido do número de sequência do *host*. No arquivo de configuração, serão cadastradas ainda informações como: domínio da rede trabalhada, localização de módulos, diretórios para alocação dos dados e demais necessidades da aplicação.

A síntese de funcionamento do ScanNet é:

1. Catalogar cada estação de trabalho, armazenando seus dados em um arquivo temporário contendo o *IP*. Uma vez adquirido o *IP*, será necessário buscar pelo endereço *MAC* e o nome do *host*;
2. Criar o arquivo de configuração `dhcpd.conf`[9], referenciando cada *MAC* a um *IP* e nome de *host*. Para maior seguranca, é necessário que o administrador habilite a opção deny unknown-clients, que impede que o *DHCP* forneca um *IP* para uma estação se a *MAC* dela não estiver cadastrada no dhcpd.conf;
3. Criar o arquivo que lançará os comandos para o *Iptables*, de forma a restringir os acessos não mencionados no *Firewall*. A operação consiste em negar toda a entrada ao servidor e liberar acessos vindo dos *IPs* e *MACs* cadastrados. Assim como *Firewall* e *DHCP*, criar os arquivos `/etc/hosts` e as entradas de *DNS*, em seus respectivos padrões.

Fazer esse trabalho de forma manual pode implicar um esforço muito grande. Um administrador pode não realizar esta operação, por causa do grande tempo que será de-

[8] Forma popularmente usada para descrever o *GNU Bourne-Again SHell*.

[9] Arquivo de configuração do Servidor *DHCP*, normalmente localizado em `/etc/dhcpd.conf`. Detalhes dessa configuração podem ser vistos em (VUKSAN, 2000).

mandado. A proposta do ScanNet é fazer isso de forma automática, seguindo o algoritimo apresentado na Figura 1.

1. Verifica a existência do arquivo de configuração, caso não encontre encerra a execução;
2. Dispara a função de checagem de dependência, caso uma dependência grave seja encontrada, será encerrada a execução, uma dependência mais insignificante resulta em uma mensagem de alerta apenas;
3. Verifica se o usuário em execução é o *root*, se sim, continua a execução, se não, aborta a execução;
4. Enquanto o número do parâmetro atual é menor que 3. Envia o parâmetro atual para o passo 5;
5. Checa o parâmetro se é nulo, igual a nis, dns ou smb. Se em nenhum dos testes, o parâmetro corrente for verdadeiro, ele é tido como inválido e a aplicação é interrompida;
6. Verifica a quantidade de módulos declarados;
7. Executa o *Nmap* para a rede declarada, e os *hosts* com status igual a *appears to be up*, são redirecionados para /tmp/scan.tmp;
8. Verifica quantos *hosts* foram encontrados no arquivo /tmp/scan.tmp, se maior que 1, vai ao passo 9;
9. Busca o *IP* no arquivo /tmp/scan.tmp e vai ao passo 10;
10. Executa o comando arp para o *IP*, salvando sua *MAC*;
11. Verifica o conteúdo do passo 10, se for igual a --, executa o comando ifconfig para pegar a *MAC* do *localhost* e vai ao passo 12;
12. Verifica o conteúdo do passo 11, se igual a -- regista o erro em error.log e volta ao passo 9, se diferente de -- vai ao passo 13;
13. Executa funções externas para resolução de *IP* em nome;
14. Verifica o conteúdo do passo 13 é nulo, se sim vai ao passo 15, se não vai ao passo 16;
15. Verifica se a interatividade foi marcada como sim, se verdadeiro pede para que o usuário digite um nome e executa o passo 16. Se não, atribui um nome sequencial e execute o passo 16;
16. Executa o(s) módulo(s) enviando as informações adquiridas e vai ao passo 17.
17. Enquanto o número do módulo menor que o total encontrado no passo 6, executa passo 16;
18. Enquanto número de *host* atual menor que a quantidade verificada no passo 8, executa o passo 9;

Figura 1: Algoritimo do ScanNet

4 Detalhes de Implementação

Desenvolvido em *Shell Bash*, o ScanNet utiliza um conjunto de ferramentas externas, para a busca, controle e seleção de informações:

1. *Nmap*[10]: utilizado para a busca de *hosts* ativos, no intervalo de rede declarado em `scannet.conf`, na variável REDE, obedecendo o seu formato. O ScanNet executa o *Nmap* com a sintaxe `nmap -sP 10.0.0.0/24`, caso a variável REDE contenha `10.0.0.0/24`. A opção `-sP` determina o scaneamento baseado em ping. O resultado da busca é redirecionado, contendo os *hosts* achados, para um arquivo temporário `/tmp/scan.tmp`;
2. *Awk, Grep, sed, cat, cut, wc*: ferramentas para manipulação de texto, com utilidade decisiva no ScanNet, essas ferramentas em conjunto são responsáveis por selecionar e extrair dados necessários de arquivos, variáveis e comandos. A aplicação é na verdade, alimentada pela ação dessas ferramentas em conjunto. O ScanNet é divido entre as ferramentas que geram dados e as ferramentas que usam estes dados gerados. Mais informações sobre as ferramentas usadas pelo ScanNet podem ser encontradas em (CAMARGO, 2005);
3. *Arp*: busca em uma lista interna de entradas *ARP* (PLUMMER, 1982), o *IP* desejado, a fim de achar a *MAC* do *host*, é usado o comando `arp -n IP`. O parâmetro `-n` indica para não resolver nomes;
4. *ifconfig*: em alguns casos, a lista de entradas *ARP* pode não conter a *MAC* do *host*. Quando o comando *ARP* é utilizado para o *localhost*[11] é um exemplo, neste caso, é usado o *ifconfig* para apanhar o endereço *MAC* do *localhost*;
5. *nmblookup*: faz parte do pacote *smbclient* do *SAMBA*[12], que no ScanNet tem a função de resolver nomes de *hosts Windows*®;
6. *host*: ferramenta pertencente ao pacote *bind-utils*[13], que realiza a resolução do *IP* em nome do *host*, por consulta a servidores *DNS*.

Foi desenvolvido uma função externa, para checar a existência das ferramentas necessárias para a utilização correta do ScanNet, assim como versões desejadas. A aplicação se localiza em `src/dependencias.sh`, verificando versões do *Nmap*, *host* e *nmblookup*. As versões requeridas são, para o *Nmap*, a versão 3.XX, para o *nmblookup* (*smbclient*) 3.X.X e para o *host* (*bind-utils*) versão 9. A checagem de dependências é executada juntamente com o ScanNet. Para desativar a checagem, é necessários remover ou comentar[14] a linha 19 do `scannet.sh`.

4.1 Procura de *Hosts* Acessíveis

O processo de busca a *hosts* ativos se dá em duas fases. Na primeira, é buscado na rede local com auxílio do *Nmap*, os *IPs* ativos, e sua saída é redirecionada para um arquivo temporário. A segunda fase será analisar o arquivo temporário, buscando apenas os *IPs* dos *hosts* ativos, como visto na Figura 2.

[10] *Nmap*: http://www.insecure.org/
[11] Maquina local, a maquina ao qual se está trabalhando.
[12] http://www.samba.org, software livre e aberto para serviços a clientes *SMB*/*CIFS*
[13] *BIND(Berkeley Internet Name Domain)* http://www.isc.org/index.pl?/sw/bind/
[14] O símbolo # em *scripts Shell Bash*, é interpretado como comentário

```
#Buscando o Conteúdo da Variável Rede no arquivo de configuração.
REDE=`cat $CONF | grep -w REDE | cut -d "=" -f 2`

#Executando o Nmap, na rede declarada, e redireccionando a saída
#para um arquivo temporário.
nmap -sP $REDE 2>> $SAIDA/error.log | grep "appears to be up." > /tmp/scan.tmp

#Busca o IP na lista de Hosts Acessíveis, um a um, dentro de um laço que percorre
#o arquivo temporário linha a linha, incrementando a variável CTH
IP=`sed --silent "$CTH"p /tmp/scan.tmp |  awk -F '(' '{print $2}' | cut -d ")" -f 1`

#Em casos onde o Nmap não apresenta o IP seguido do Nome do Host, a função acima
#retorna Nulo, a condição abaixo checa essa possibilidade e busca novamente o IP
#forma no arquivo temporário.
if [ $IP ]
 then
  echo -n
 else
  IP=`sed --silent "$CTH"p /tmp/scan.tmp |  awk '{print $2}'`
fi
```

Figura 2: Buscando *Hosts*

4.2 Obtenção do Endereco *MAC* do *Host*

O endereço *MAC* está contido na lista de entradas *ARP*. Essa lista é mantida enquanto existe atividade de comunicação entre o *host* local e um *host* remoto, dentro da mesma máscara de rede[15]. A lista de antradas *ARP*, não contém a entrada do *localhost*, e nem de *hosts* fora do intervalo de sua máscara de rede. O segundo caso é desconsiderado, uma vez que o intuito do ScanNet é atuar em redes locais, mas, no caso do *localhost*, o endereço *MAC* é obtido através do comando `ifconfig`.

Nos testes, caso o ScanNet não consiga adquirir a *MAC*, a variável *ARP* conterá `--`. Existe uma condição que, se a variável *ARP* contém `--`, o processo adiante é abortado, será gerado então um registro de erro para o *host* em questão. A rotina de busca pelo endereço *MAC*, pode ser visto na Figura 3.

4.3 Resolução de *IP* em Nome

A resolução de *IPs* em nomes é o processo mais crítico no ScanNet. Por ser modular e usar fontes diversas para resolver os nomes, na ordem de prioridade selecionada, tornou-se a tarefa mais complexa da aplicação. A resolução pode ser feita de cinco formas:

***DNS*:** utilizando a função `src/dns.sh`, usa os servidores *DNS* para tentar resolver o *IP* corrente em nome;

***NIS*:** utilizando a função `src/nis.sh`, usa o arquivo `/etc/hosts` ou servidor *NIS*, para resolver o *IP* corrente em nome;

***SMB*:** utilizando a função `src/smb.sh`, usa o *smbclient* com o comando `nmblookup -A`, para tentar resolver o *IP* corrente em nome;

[15] Usada para delimitar intervalos de endereços possíveis de rede.

```
#Buscando a MAC pelo comando arp
ARP=`arp -n $IP | grep $IP | awk '{print $3}'`

#Checando se o IP que está sendo processado, pode ser o IP Local para obter
#sua MAC pelo ifconfig uma vez que o arp não tem entradas para Hosts Locais.
if [ $ARP = '--' ]
then
 ARPTEMP=`ifconfig | grep -w -B 1 $IP | grep HW | cut -d "H" -f 2 | awk '{print $2}'`
  if [ $ARPTEMP ]
  then
   ARP=$ARPTEMP
  else
   echo -n
  fi
else
 echo -n
fi
```

Figura 3: Endereço *MAC*

INTERATIVO=sim: em caso do ScanNet não achar nenhum nome nas tentativas de resolução, solicitará ao usuário que digite um nome para o *IP* corrente;

INTERATIVO=nao: em caso do ScanNet não achar nenhum nome nas tentativas de resolução, esta opção usa o Nome Máscara, previamente configurado no `scannet.conf`, mais o número de sequência de cadastro do *host* para nomeá-lo.

As funções `src/dns.sh`, `src/nis.sh` e `src/smb.sh` são externas, sendo chamadas em ordem de declaração no arquivo `scannet.conf`. Caso não seja encontrado um nome na primeira resolução declarada, será buscado na segunda, assim por diante, até ser achado algum nome ou cair na interatividade. Na Figura 4, é apresentado o trecho de código que lida com a leitura da ordem de execução declarada e execução das funções externas.

4.4 Geração de Arquivos de Configurações

O ScanNet é modular, isso possibilita a livre construção de novos módulos de geração de arquivos de configuração, alteração de antigos, adaptações, etc. Os módulos são construídos utilizando as variáveis de referência, como apresentado na Figura 5.

A partir das variáveis de referência, os módulos podem ser construídos, utilizando os dados que serão passados a cada módulo pelo ScanNet. Um módulo constitui na utilização das variáveis de referência para modelagem ou apresentação na tela dos dados obtidos. Um exemplo de módulo que apresenta os dados simultaneamente na tela é o `modulos/verbose.sh`, como pode ser visto na Figura 6.

Após a criação de cada módulo, para ser executado, ele deve ser declarado no arquivo de configurações. Seguindo o exemplo do `modulos/verbose.sh`, seria declarado `MODULO=modulos/verbose.sh`. No caso de haver mais de um módulo a ser executado pelo do ScanNet, eles devem ser declarados exatamente como o `modulos/verbose.sh`, em linhas separadas. Com isso, a aplicação principal lê o conteúdo de MODULO=, em cada linha, e a executa para cada *host* com nome encontrado.

```
#Exemplo de declaração de Ordem no arquivo de configuração:
ORDEM=dns,smb,nis
#Rotina para buscar no arquivo de configurações os métodos de resoluções na orden declarada
#checar a validade das declarações, e armazenar a ordem de prioridade de resolução.
CPREF=1
while [ $CPREF -le 3 ]
do
 PREFTESTE=`cat $CONF | grep -w ORDEM | cut -d "=" -f 2 | cut -d "," -f $CPREF | cut -b 1-3`
if [ $PREFTESTE ]
then
 if [ $PREFTESTE = 'nis' ]
 then
  echo -n
   else if [ $PREFTESTE = 'dns' ]
        then
         echo -n
          else if [ $PREFTESTE = 'smb' ]
               then
                echo -n
               else
                echo -e '\tParametro de Ordem de Preferencia: '$PREFTESTE', invalido!'
                echo -e '\tVerifique o arquivo scannet.conf em: Preferencias por Nome de Hosts'
                echo -e '\tE digite os Parametros de Ordem corretos, da forma correta.'
                exit 1
               fi
        fi
 fi
else
 echo -n
fi
 CPREF=$((CPREF+1))
done
PREF1=`cat $CONF | grep -w ORDEM | cut -d "=" -f 2 | awk -F "," '{print $1}'`
PREF2=`cat $CONF | grep -w ORDEM | cut -d "=" -f 2 | awk -F "," '{print $2}'`
PREF3=`cat $CONF | grep -w ORDEM | cut -d "=" -f 2 | awk -F "," '{print $3}'`
#As variáveis PREF* são utilizadas para execução das funções externas
[ $PREF1 ] && ORD1=`source $SRC$PREF1.sh $IP`
[ $PREF2 ] && ORD2=`source $SRC$PREF2.sh $IP`
[ $PREF3 ] && ORD3=`source $SRC$PREF3.sh $IP`
#As variáveis obtidas são armazenadas em ORD1, ORD2 e ORD3
#Se as mesmas se encontram nulas, o sistema verifica o INTERATIVO,
#e Gera Sequencialmente o Nome ou Solicita ao Usuário.
if [ $ORD1 ]
then
 NHOST=$ORD1
 else if [ $ORD2 ]
      then
       NHOST=$ORD2
       else if [ $ORD3 ]
            then
             NHOST=$ORD3
             else if [ $INTERAT = 'sim' ]
                  then
                   echo -en '\tDigite um Nome de Host para o IP '$IP': '
                   read NHOST
                  else
                   NHOST=$NOMESEQ$CTH
                  fi
            fi
      fi
fi
```

Figura 4: Resolução de *IP* em Nome

```
# Variáveis de Referência:
# $1 = Diretório onde será armazenado os resultados
# $2 = IP do Host
# $3 = MAC do Host
# $4 = Nome do Host
# $5 = Domínio do Host
# $6 = Sequência de Hosts Achados
# $7 = Sequência de Hosts Cadastrados
# $8 = Módulo em Execução
# $9 = Rede Sendo Scaneada
```

Figura 5: Variáveis para Construção de Módulos

```
#!/bin/sh
#
# Módulo para exibição das informações simultâneamente.
# Variáveis de Referência:
# $1 = Diretório onde será armazenado os resultados
# $2 = IP do Host
# $3 = MAC do Host
# $4 = Nome do Host
# $5 = Domínio do Host
# $6 = Sequência de Hosts Achados
# $7 = Sequência de Hosts Cadastrados
# $8 = Módulo em Execução
# $9 = Rede Sendo Scaneada

echo -e '\tHost: '$7' \t'$2' \t'$3' \t'$4''
```

Figura 6: Exemplo de Módulo

5 Testes Realizados

O Scanet foi testado nas distribuições: Red Hat 7.0, Red Hat 8.0, Fedora Core 1, Conectiva Linux 9, Conectiva Linux 10. Para isso, ele foi configurado com os seguintes parâmetros no arquivo de configurações scannet.conf:

- `REDE=10.0.0.0/24`
- `DOMINIO=HMRP.ORG.BR`
- `NOMESEQ=HMRP`
- `ORDEM=dns,smb,nis`
- `INTERATIVO=sim`
- `SRC=src/`
- `SAIDA=saidas/`
- `MODULO=modulos/verbose.sh`
- `MODULO=modulos/dhcp.sh`
- `MODULO=modulos/hosts.sh`
- `MODULO=modulos/dns.sh`

- `MODULO=modulos/firewall.sh`

Em seqüência, ele foi executado para a rede declarada. O módulo `modulos/verbose.sh` apresentou a impressão em tela das informações do *host* encontrado, como mostrado na Figura 7.

```
Scaneando Rede: 10.0.0.0/24
Buscando Informacoes em: 4 Host(s) Encontrado(s) OnLine(s)
        Host: 1         10.0.0.1        00:00:E2:33:E0:A7       SRV01
        Host: 2         10.0.0.5        00:0B:CD:D3:01:65       SRVAUT
        Host: 3         10.0.0.15       00:60:67:79:D2:0E       COMPRAS-01
        Host: 4         10.0.0.16       00:0F:EA:9D:5E:0F       TECINF-01
```

Figura 7: Executando ScanNet

O módulo `MODULO=modulos/dhcp.sh`, gerou um arquivo em `saidas/dhcpd.txt`, nos moldes do `/etc/dhcpd.conf` como foi previamente programado para fazê-lo (Figura 8).

```
host SRV01 {
        hardware ethernet 00:00:E2:33:E0:A7;
        fixed-address 10.0.0.1;
}
host SRVAUT {
        hardware ethernet 00:0B:CD:D3:01:65;
        fixed-address 10.0.0.5;
}
host COMPRAS-01 {
        hardware ethernet 00:60:67:79:D2:0E;
        fixed-address 10.0.0.15;
}
host TECINF-01 {
        hardware ethernet 00:0F:EA:9D:5E:0F;
        fixed-address 10.0.0.16;
}
```

Figura 8: Arquivo `dhcpd.txt`

O módulo `MODULO=modulos/hosts.sh`, gerou na mesma pasta, o arquivo `hosts.txt`, nos moldes do `/etc/hosts` o qual foi programado para fazê-lo (Figura 9).

```
10.0.0.1        SRV01.HMRP.ORG.BR       SRV01
10.0.0.5        SRVAUT.HMRP.ORG.BR      SRVAUT
10.0.0.15       COMPRAS-01.HMRP.ORG.BR  COMPRAS-01
10.0.0.16       TECINF-01.HMRP.ORG.BR   TECINF-01
```

Figura 9: Arquivo `hosts.txt`

O módulo `MODULO=modulos/dns.sh`, responsável pela geração de entradas no padrão dos arquivos de configuração de *DNS*, gerou o arquivo `saidas/dns.txt` como como mostrado na Figura 10.

```
SRV01    IN      A       10.0.0.1
SRVAUT   \index{}IN      A       10.0.0.5
COMPRAS-01       IN      A       10.0.0.15
TECINF-01        IN      A       10.0.0.16
```

Figura 10: Arquivo `dns.txt`

O módulo `MODULO=modulos/firewall.sh`, responsável pela geração do *script* de regras para o *Iptables*, gerou o arquivo `firewall.txt`, como pode ser visto na Figura 11. As regras consistem em negar o acesso *INPUT*, e liberar os *hosts* com suas respectivas *MACs* encontrados ativos na rede.

```
#!/bin/sh
iptables -F
iptables -X
iptables -t nat -F
iptables -t nat -X
iptables -t filter -P INPUT DROP
iptables -t filter -P OUTPUT ACCEPT
iptables -t filter -P FORWARD DROP
iptables -A INPUT -i lo -j ACCEPT
iptables -A INPUT -s 10.0.0.1 -m mac --mac-source 00:00:E2:33:E0:A7 -j ACCEPT
iptables -A INPUT -s 10.0.0.5 -m mac --mac-source 00:0B:CD:D3:01:65 -j ACCEPT
iptables -A INPUT -s 10.0.0.15 -m mac --mac-source 00:60:67:79:D2:0E -j ACCEPT
iptables -A INPUT -s 10.0.0.16 -m mac --mac-source 00:0F:EA:9D:5E:0F -j ACCEPT
```

Figura 11: Arquivo `firewall.txt`

6 Conclusão

Tornar os problemas mais simples e rápidos de serem resolvidos, com soluções práticas, maleáveis, livres e de código fonte aberto, é uma forma de deixar o trabalho manual e repetitivo de lado e dar atenção a outras ameaças que realmente podem trazer problemas futuros. O trabalho de pesquisar individualmente computador a computador e gerar os arquivos `dhcpd.conf`, `/etc/hosts`, *DNS* e *Firewall*, como foi mostrado aqui, dependendo da amplitude da rede, pode levar dias. A realização desse trabalho traz, indiscutivelmente, mais segurança para o ambiente e, com a utilização do ScanNet, se tornou fácil.

A proposta de um aplicativo modular, em uma linguagem de fácil assimilação, facilita o engajamento de novos personagens no desenvolvimento, melhoria e correção de erros da aplicação. Ainda nova e com possíveis erros, que podem ser corrigidos com a rápida alteração em seus códigos, por pessoas que tenham um conhecimento intermediário no desenvolvimento de aplicativos em *Shell Bash*.

O ScanNet pode ser obtido em `http://www.lopan.eti.br/` e encontra-se licenciado sob a licença GPL. Apesar de sua alteração ser livre, o primeiro autor deste trabalho

solicita que alterações efetuadas nele sejam informadas por *e-mail*, para melhor controle e funcionalidade de futuras versões.

Referências

CAMARGO, H. A. *Automação de Tarefas*. Lavras: UFLA/FAEPE, 2005.

CONSORTIUM, I. S. *Dynamic Host Configuration Protocol*. [S.l.], 2004. Disponível em: <http://www.isc.org/index.pl?/sw/dhcp/>.

FOUNDATION, F. S. *GNU Bourne-Again SHell*. [S.l.], 2002. Disponível em: <http://maconlinux.net/linux-man-pages/en/bash.1.html>.

KIRCH, O.; DAWSON, T. *Linux Network Administrator's Guide*. Sebastopol: O'Reilly, 2000.

KUKUK, T. *Homepage of the Linux NIS/NIS+ Projects*. [S.l.], 2003. Disponível em: <http://www.linux-nis.org/>.

LANGFELDT, N. *DNS HOWTO*. [S.l.], 2001. Disponível em: <http://www.linux.org-/docs/ldp/howto/DNS-HOWTO.html>.

PLUMMER, D. C. *An Ethernet Address Resolution Protocol*. IETF, 1982. Disponível em: <ftp://ftp.rfc-editor.org/in-notes/rfc826.txt>.

SILVA, G. M. da. *Firewall iptables*. [S.l.], 2005. Disponível em: <http://focalinux-.cipsga.org.br/guia/avancado/ch-fw-iptables.htm>.

UCHÔA, J. Q. *Segurança Computacional*. 2. ed. Lavras: UFLA/FAEPE, 2005. (Curso de Pós Graduação "Lato Sensu" (Especialização) a Distância em Administração em Redes Linux).

VUKSAN, V. *DHCP mini-HOWTO)*. [S.l.], 2000. Disponível em: <http://www.ibiblio-.org/pub/Linux/docs/HOWTO/DHCP>.