DESENVOLVIMENTO E IMPLEMENTAÇÃO DE SERVIÇOS DE LINUX (DHCP-KEA E DNS-BIND)

Formador: Dário Quental

SÍNTESE

Este Projeto consistiu na realização de dois *scripts Bash* com o objetivo da configuração de uma infraestrutura de rede completa: um automatiza a gestão de nomes (**DNS**) e o outro automatiza a gestão de endereços (**DHCP-KEA**), garantindo a estabilidade e a segurança do servidor.

Sérgio Correia

0925GRSC

Índice

Resumo Executivo

O presente projeto teve como objetivo a implementação e automatização de dois serviços essenciais numa infraestrutura de rede: um servidor DNS (BIND) e um servidor DHCP (Kea), ambos instalados em sistemas CentOS Stream 10. Com o intuito de garantir consistência, rapidez de implementação e facilidade de manutenção, foram desenvolvidos scripts em Bash que automatizam todo o processo de instalação, configuração e validação dos serviços.

O servidor DNS foi configurado como servidor *autoritativo* e de *cache*, incluindo a criação das zonas de resolução direta e inversa, a definição de servidores encaminhadores para consultas externas e a ativação de mecanismos de validação e registo de atividade. O servidor DHCP, baseado no Kea, foi configurado para atribuição dinâmica de endereços IPv4, definindo a sub-rede, máscara, *gateway*, DNS, tempo de concessão e reservas estáticas, com gestão centralizada em ficheiro de configuração no formato JSON.

Foram aplicadas medidas de segurança adequadas ao contexto operacional, nomeadamente a utilização de *firewalld*, *SELinux* em modo *enforcing*, fail2ban para proteção contra tentativas de acesso indevido e organização cuidadosa do espaço de endereçamento de forma a separar endereços estáticos de dinâmicos. A implementação foi validada através de testes práticos em clientes Linux, bem como pela análise de *logs* e verificação do correto funcionamento dos serviços.

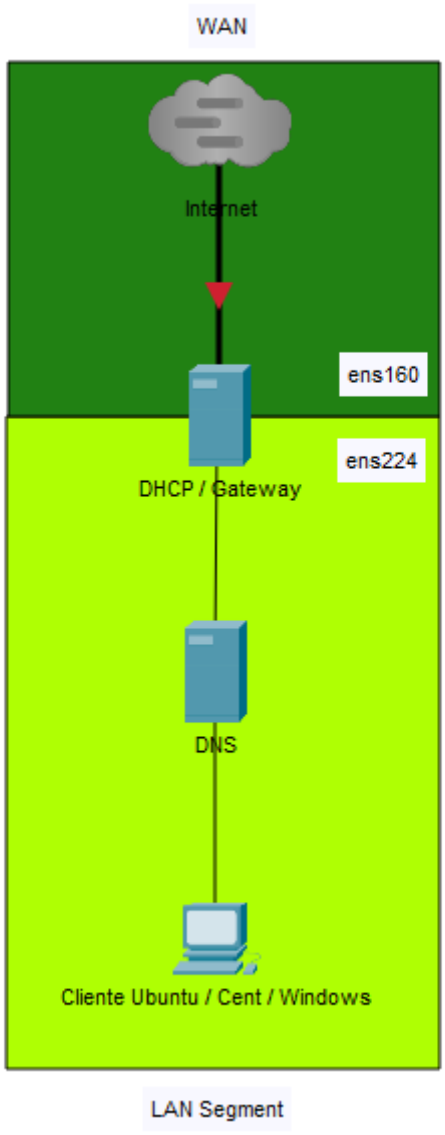
Como resultado, obteve-se uma solução funcional, segura e totalmente automatizada, acompanhada de documentação técnica e scripts reutilizáveis, adequada para utilização em ambientes de produção, formação ou laboratório.

Metodologia

A metodologia seguiu uma abordagem faseada baseada na separação entre a rede WAN e o segmento LAN.

Começou pela configuração do servidor que assume o papel de **Gateway/DHCP**, com duas interfaces de rede: uma ligada à Internet (ens160) e outra à rede interna (ens224), assegurando tanto a conectividade externa como a distribuição de endereços IP aos clientes.

Em seguida, procedeu-se à instalação e configuração do **servidor DNS** dentro do segmento LAN, garantindo a resolução interna e o encaminhamento de pedidos externos. Por fim, realizou-se a ligação de um cliente (Linux/Windows) à mesma rede, validando a receção do endereço via DHCP e a resolução de nomes através do DNS, confirmando o funcionamento integrado da infraestrutura.



Comandos Utilizados nos Scripts

Script DNS-BIND

Este Script instala e configura o DNS-BIND, cria as zonas de resolução direta e inversa (Forward e Reverse Zones), aplica as definições de rede, ativa o serviço e valida se o DNS está a funcionar corretamente.

```
set -e
```

O que faz: O set -e é um comando a correr em Scripts para que este pare a execução se houver algum erro em algum dos comandos presentes no Script.

Passo 1 – Definir Permissões do Script

```
chmod 775 config_dns_bind.sh
```

O que faz: Define permissões de leitura, escrita e execução para o proprietário e grupo, e leitura e execução para outros.

Passo 2 – Recolha de Informação do Utilizador

```
echo ""
read -p "Introduza o domínio (ex: empresa.local): " DOMINIO
read -p "Introduza o IP do servidor de Classe C (ex: 192.168.0.10): " IP_SERVIDOR_DNS
read -p "Introduza o IP do Servidor DHCP/NAT: " IP_FORWARDER
sleep 0.5

nmcli device status | grep ethernet

read -p "Indique a interface PRINCIPAL ÚNICA (ex: ens224): " LAN_INTERFACE # INTERFACE ÚNICA
sleep 0.5

echo ""
echo "Informações recolhidas com sucesso!"
sleep 0.5
```

O que faz: Solicita ao utilizador os dados necessários para configurar o servidor DNS.

Passo 3 – Configuração da Interface LAN com IP Estático

```
sudo nmcli connection modify "$LAN_INTERFACE" ipv4.addresses "$IP_SERVIDOR_DNS/24"
echo "Alterar a LAN_INTERFACE para IP $IP_SERVIDOR_DNS/24"
sleep 0.5

# O que faz o ipv4.method manual: Desativa DHCP e força configuração manual de IP.

sudo nmcli connection modify "$LAN_INTERFACE" ipv4.method manual
echo "Definir método de IP para manual"
sleep 0.5

# O que faz o ipv4.gateway: Define o gateway padrão (router) para acesso à rede externa.

sudo nmcli connection modify "$LAN_INTERFACE" ipv4.gateway "$IP_FORWARDER"
echo "Definir gateway para $IP_FORWARDER"
sleep 0.5

# O que faz o connection up: Reload da interface LAN com as novas configurações aplicadas.

sudo nmcli connection up "$LAN_INTERFACE"
echo "Aplicar configurações na interface $LAN_INTERFACE"
sleep 0.5

echo "Configuração da interface LAN concluída."
sleep 0.5
```

O que faz: Configura a interface de rede principal com um IP estático fornecido pelo utilizador.

Passo 4 – Teste de Conectividade à Internet

```
echo ""
echo "Teste de conectividade antes da instalação."

nmcli con mod ens224 ipv4.dns "8.8.8.8"
nmcli con up ens224

ping -c 3 8.8.8.8

echo "Conectividade confirmada!"
sleep 0.5
echo ""
```

O que faz: Verifica se o servidor consegue aceder à Internet antes de instalar pacotes.

Passo 5 – Instalação do Serviço BIND

```
echo "A instalar BIND..."

sudo dnf install -y bind bind-utils

echo "BIND instalado com sucesso!"
sleep 0.5

localhost="127.0.0.1"
sudo nmcli con mod ens224 ipv4.dns "$localhost"
```

O que faz: Instala o servidor DNS BIND e as ferramentas associadas.

Passo 6 - Instalação e Configuração do Fail2Ban para Proteger o BIND

```
echo "A instalar EPEL..."
sudo dnf install -y epel-release

echo "A instalar Fail2Ban e firewalld integration..."

# Fail2Ban e fail2ban-firewalld estão agora no EPEL

sudo dnf install -y fail2ban fail2ban-firewalld

echo "Fail2Ban instalado com sucesso!"
sleep 0.5
```

O que faz: Instala o Fail2Ban e configura uma *jail* específica para proteger o servidor DNS contra ataques.

Passo 6.1 - Criar Jail (Regra) para o BIND DNS

```
        echo "A criar jail 'bind-dns' em /etc/fail2ban/jail.d/bind-dns.conf..."

        IP_TO_IGNORE="127.0.0.1/8 $IP_SERVIDOR_DNS"

        sudo tee /etc/fail2ban/jail.d/bind-dns.conf >/dev/null << EOF
[bind-dns]
enabled  = true
port     = domain
protocol = udp,tcp
filter   = named-refused
logpath  = /var/log/named/security.log
maxretry = 10
findtime = 60
bantime  = 3600
ignoreip = $IP_TO_IGNORE
action   = firewallcmd-ipset
EOF
```

O que faz: Define os parâmetros de banimento para o serviço DNS.

Passo 6.2 - Iniciar e Habilitar o Serviço Fail2Ban

```
echo "A iniciar e habilitar o serviço Fail2Ban..."

# Reiniciar named para garantir que as novas configs de log são carregadas

sudo systemctl restart named

sudo systemctl enable --now fail2ban

echo "Fail2Ban configurado para proteger o BIND/DNS."
sleep 0.5
```

O que faz: Ativa e inicia o serviço Fail2Ban, garantindo que ele arranca automaticamente no *boot*.

Passo 7 – Configurar DNS da interface LAN para localhost

```
echo ""
echo "A definir DNS da interface principal ($LAN_INTERFACE) para 127.0.0.1 (Loopback)..."
sleep 0.5

sudo nmcli connection modify "$LAN_INTERFACE" ipv4.dns "127.0.0.1"
echo "Aplicar configuração de DNS na interface $LAN_INTERFACE..."
sleep 0.5

sudo nmcli connection up "$LAN_INTERFACE"
echo "DNS da interface $LAN_INTERFACE definido para localhost."
sleep 0.5

echo "Limpeza de rede concluída."
sleep 0.5
echo ""
```

O que faz: Define o servidor DNS da interface LAN para o próprio servidor (localhost).

Passo 8 - Extrair Octetos do IP para Criação da Zona Reversa

```
OCTETO_1=$(echo "$IP_SERVIDOR_DNS" | cut -d'.' -f1)
OCTETO_2=$(echo "$IP_SERVIDOR_DNS" | cut -d'.' -f2)
OCTETO_3=$(echo "$IP_SERVIDOR_DNS" | cut -d'.' -f3)
OCTETO_4=$(echo "$IP_SERVIDOR_DNS" | cut -d'.' -f4)

REVERSE_ZONE_ID="${OCTETO_3}.${OCTETO_2}.${OCTETO_1}.in-addr.arpa"

SERIAL_DATE=$(date +%s)

echo "Domínio: $DOMINIO"
echo "IP Servidor DNS: $IP_SERVIDOR_DNS"
echo "Zona Reversa: $REVERSE_ZONE_ID"
echo "Serial Date: $SERIAL_DATE"
sleep 0.5
echo ""

echo "A criar diretório e ficheiros de logs..."
```

O que faz: Divide o endereço IP em 4 partes (octetos) para poder criar a zona de resolução inversa.

Passo 8.1 – Criação de Diretórios e Ficheiros de Logs

```
sudo mkdir -p /var/log/named
sudo chown named:named /var/log/named
sudo chmod 755 /var/log/named
sudo touch /var/log/named/bind_queries.log
sudo touch /var/log/named/security.log
sudo chown named:named /var/log/named/bind_queries.log
sudo chown named:named /var/log/named/security.log
echo "Ajustando contexto SELinux para os logs do BIND..."
sudo restorecon -Rv /var/log/named

echo "Diretório e ficheiros de logs criados e permissões/SELinux ajustados."
sleep 0.5
echo ""
```

O que faz: Cria o diretório e ficheiros necessários para armazenar os logs do BIND.

Passo 8.2 – Validação de Variáveis

```
echo "A validar variáveis necessárias..."

if [ -z "$DOMINIO" ]; then
    echo "ERRO CRÍTICO: Variável DOMINIO não está definida!"
    exit 1
fi

if [ -z "$IP_SERVIDOR_DNS" ]; then
    echo "ERRO CRÍTICO: Variável IP_SERVIDOR_DNS não está definida!"
    exit 1
fi

if [ -z "$OCTETO_1" ] || [ -z "$OCTETO_2" ] || [ -z "$OCTETO_3" ] || [ -z "$OCTETO_4" ]; then
    echo "ERRO CRÍTICO: Octetos não estão definidos!"
    exit 1
fi

if [ -z "$SERIAL_DATE" ]; then
    echo "ERRO CRÍTICO: Variável SERIAL_DATE não está definida!"
    exit 1
fi

echo "DOMINIO: $DOMINIO"
echo "IP_SERVIDOR_DNS: $IP_SERVIDOR_DNS"
echo "OCTETOS: $OCTETO_1.$OCTETO_2.$OCTETO_3.$OCTETO_4"
echo "SERIAL_DATE: $SERIAL_DATE"
echo "REVERSE_ZONE_ID: $REVERSE_ZONE_ID"

echo ""
echo "Todas as variáveis validadas com sucesso!"
sleep 0.5
echo ""
```

O que faz: Garante que todas as variáveis necessárias existem antes de criar os ficheiros de zona.

Passo 9 – Criar Ficheiro de Zona Direta (Forward Zone)

```
        echo "A criar Forward Zone..."

        sudo tee /var/named/${DOMINIO}.db >/dev/null << EOF
\$TTL 86400
@ IN SOA ${DOMINIO}. root.${DOMINIO}. (
        ${SERIAL_DATE}  ; Serial
        3600            ; Refresh
        1800            ; Retry
        604800          ; Expire
        86400 )         ; Minimum TTL
    IN NS     ns.${DOMINIO}.
ns IN A       ${IP_SERVIDOR_DNS}
@ IN A       ${IP_SERVIDOR_DNS}
EOF
```

O que faz: Cria o ficheiro que resolve nomes de domínio para endereços IP (nome -> IP).

Explicação dos registos DNS:

\$TTL 86400: Time To Live - tempo (em segundos) que os registos são guardados em cache (24 horas).

SOA: Start of Authority - define o servidor autoritativo para esta zona.

Serial: Número de versão da zona (incrementa a cada alteração).

Refresh: Tempo que servidores secundários esperam antes de verificar atualizações.

Retry: Tempo de espera se o refresh falhar.

Expire: Tempo após o qual a zona expira se não houver contacto com o servidor primário.

NS: Name Server - define qual o servidor de nomes para este domínio.

A: Address Record - associa um nome a um endereço IPv4.

@: Representa o próprio domínio (empresa.local).

Passo 10 – Criar Ficheiro de Zona Inversa (Reverse Zone)

```
        echo "A criar zona inversa (Reverse Zone)..."

        sudo tee /var/named/${OCTETO_3}.${OCTETO_2}.${OCTETO_1}.db >/dev/null << EOF
\$TTL 86400
@ IN SOA ${DOMINIO}. root.${DOMINIO}. (
        ${SERIAL_DATE}  ; Serial
        3600            ; Refresh
        1800            ; Retry
        604800          ; Expire
        86400 )         ; Minimum TTL
    IN NS     ns.${DOMINIO}.
${OCTETO_4} IN  PTR   ns.${DOMINIO}.
EOF
```

O que faz: Cria o ficheiro que resolve endereços IP para nomes de domínio (IP -> nome).

Passo 11 – Configurar named.conf (Ficheiro Principal do BIND)

```
    sudo tee /etc/named.conf >/dev/null << EOF
// named.conf do servidor autoritativo e de caching
options {
    listen-on port 53 { any; };
    listen-on-v6 port 53 { none; };
    directory "/var/named";
    dump-file "/var/named/data/cache_dump.db";
    statistics-file "/var/named/data/named_stats.txt";
    memstatistics-file "/var/named/data/named_mem_stats.txt";
    allow-query { any; };

    forwarders {
        8.8.8.8;
    };
    forward only;

    recursion yes;

    managed-keys-directory "/var/named/dynamic";
    pid-file "/run/named/named.pid";
    session-keyfile "/run/named/session.key";
};

logging {
    channel default_log {
        file "/var/log/named/bind_queries.log" versions 3 size 5m;
        severity info;
        print-time yes;
    };
    channel security_log {
        file "/var/log/named/security.log" versions 3 size 5m;
        severity info;
        print-time yes;
    };
    category queries { default_log; };
    category security { security_log; };
    category client { security_log; };
};

zone "." IN {
    type hint;
    file "named.ca";
};

include "/etc/named.rfc1912.zones";
include "/etc/named.root.key";
EOF
```

O que faz: Cria o ficheiro de configuração principal do servidor BIND com todas as opções necessárias.

Explicação das opções do named.conf:

listen-on port 53 { any; }: Escuta pedidos DNS em todas as interfaces na porta 53.

listen-on-v6 port 53 { none; }: Desativa IPv6.

directory "/var/named": Diretório onde ficam os ficheiros de zona.

allow-query { any; }: Permite consultas DNS de qualquer cliente.

forwarders { 8.8.8.8; }: DNS externo usado para consultas que este servidor não conhece.

forward only: Força o uso do forwarder (não tenta resolver sozinho domínios externos).

recursion yes: Permite que o servidor faça consultas recursivas (essencial para clientes).

logging: Configuração de logs para registar todas as consultas DNS.

Passo 12 – Adicionar Zonas Personalizadas ao named.conf

```
        echo "A adicionar zonas personalizadas ao named.conf..."

        sudo tee -a /etc/named.conf >/dev/null << EOF

zone "${DOMINIO}" IN {
    type master;
    file "${DOMINIO}.db";
    allow-update { none; };
};

zone "${REVERSE_ZONE_ID}" IN {
    type master;
    file "${OCTETO_3}.${OCTETO_2}.${OCTETO_1}.db";
    allow-update { none; };
};
EOF
```

O que faz: Adiciona as definições das zonas direta e inversa ao ficheiro de configuração.

Passo 13 – Definir Permissões dos Ficheiros da Zona

```
echo "A definir permissões dos ficheiros de zona..."

sudo chown named:named /var/named/${DOMINIO}.db
sudo chown named:named /var/named/${OCTETO_3}.${OCTETO_2}.${OCTETO_1}.db

echo "Permissões definidas."
sleep 0.5
echo ""
```

O que faz: Altera o proprietário dos ficheiros de zona para o utilizador "named".

Passo 14 – Validar Configuração do BIND (named.conf)

```
echo "A validar named.conf..."

if sudo named-checkconf; then
    echo "named.conf está OK!"
else
    echo "Erro 16 no named.conf! Verifique a configuração."
    exit 1
fi

sleep 0.5
```

O que faz: Verifica se o ficheiro de configuração principal tem erros de sintaxe.

Passo 15 – Validar Zona Direta

```
echo "A validar forward zone..."

if sudo named-checkzone ${DOMINIO} /var/named/${DOMINIO}.db; then
    echo "Forward zone está OK!"
else
    echo "Erro 17 na forward zone! Verifique o ficheiro."
    exit 1
fi

sleep 0.5
```

O que faz: Verifica se o ficheiro de zona direta tem erros de sintaxe ou inconsistências.

Passo 16 – Validar Zona Inversa

```
echo "A validar reverse zone..."

if sudo named-checkzone ${REVERSE_ZONE_ID} /var/named/${OCTETO_3}.${OCTETO_2}.${OCTETO_1}.db; then
    echo "Zona inversa está OK!"
else
    echo "Erro 18 na zona inversa! Verifique o ficheiro."
    exit 1
fi

sleep 0.5
echo ""
```

O que faz: Verifica se o ficheiro de zona inversa tem erros de sintaxe ou inconsistências.

Passo 17 – Configurar Firewall

```
echo "A configurar firewall..."

sudo firewall-cmd --permanent --add-service=dns

sudo firewall-cmd --reload

echo "Firewall configurada com sucesso!"
sleep 0.5
echo ""
```

O que faz: Adiciona regras à firewall para permitir tráfego DNS (porta 53 TCP/UDP).

Passo 18 – Iniciar e Habilitar o Serviço BIND

```
echo -n "A iniciar o Serviço BIND..."
for i in {1..50}; do
    printf "\rA carregar: [ %-50s ]" "$(printf '=%.0s ' $(seq 1 $i))"
    sleep 0.1
done

echo ""

sudo systemctl enable --now named

sleep 0.5

sudo systemctl status named

echo ""
echo "DNS CONFIGURADO COM SUCESSO!"
echo ""
sleep 0.5
;;
```

O que faz: Inicia o servidor DNS BIND e configura-o para arrancar automaticamente no boot, com uma barra de Progresso extra.

Passo 19 – Menu de Gestão de Registos DNS

```
read -p "Introduza o IP do Servidor DNS (ex: 192.168.0.10): " IP_SERVIDOR_DNS
read -p "Introduza o domínio (ex: empresa.local): " DOMINIO

# O que faz: Re-extrair octetos e rede para uso nos passos 21 e 22

OCTETO_1=$(echo "$IP_SERVIDOR_DNS" | cut -d'.' -f1)
OCTETO_2=$(echo "$IP_SERVIDOR_DNS" | cut -d'.' -f2)
OCTETO_3=$(echo "$IP_SERVIDOR_DNS" | cut -d'.' -f3)
OCTETO_4=$(echo "$IP_SERVIDOR_DNS" | cut -d'.' -f4)

REVERSE_ZONE_ID="${OCTETO_3}.${OCTETO_2}.${OCTETO_1}.in-addr.arpa"

# O que faz: Define a sub-rede para validação da rede

IP_REDE_SERVIDOR="${OCTETO_1}.${OCTETO_2}.${OCTETO_3}"

# O que faz: Tenta obter o serial date para gestão (Se o ficheiro existir, senão usa o timestamp atual)

if [ -f /var/named/${DOMINIO}.db ]; then
    SERIAL_DATE=$(grep -m 1 '; Serial' /var/named/${DOMINIO}.db 2>/dev/null | awk '{print $1}')
    if [ -z "$SERIAL_DATE" ]; then
        SERIAL_DATE=$(date +%s)
    fi
else
    echo "Aviso 19: O ficheiro de zona direta não foi encontrado. Usando Serial Date atual."
    SERIAL_DATE=$(date +%s)
fi

echo ""
echo "IP DNS para validações: $IP_SERVIDOR_DNS (Sub-rede: $IP_REDE_SERVIDOR.0)"
sleep 0.5
echo ""
echo "Deseja adicionar ou consultar registos no DNS? (y/n): "
read -p "Resposta: " GERIR_REGISTOS
```

O que faz: Permite ao utilizador adicionar ou consultar registos DNS após a configuração inicial.

Passo 19.1 – Loop do Menu Principal

```
while true; do
    echo ""
    echo "========================================="
    echo "   GESTÃO DE REGISTOS DNS"
    echo "========================================="
    echo ""
    echo "Escolha o tipo de operação:"
    echo "1 - Adicionar Nome para IP (zona direta)"
    echo "2 - Adicionar IP para Nome (zona inversa)"
    echo "3 - Ver registos existentes"
    echo "4 - Sair"
    echo ""
    read -p "Opção: " OPCAO_GESTAO
```

O que faz: Mantém o menu ativo até o utilizador escolher sair.

Passo 21.1.1 – Adicionar Registo à Zona Direta (A Record)

```
echo ""
echo "--- Adicionar Nome para IP (zona direta) ---"
echo ""
read -p "Nome do host (ex: pc1, servidor, router): " NOME_HOST
read -p "Endereço IP (ex: 192.168.1.20): " IP_HOST
```

O que faz: Adiciona um novo nome de host que aponta para um IP.

Passo 21.1.2 – Validar se o IP pertence à mesma Rede

```
IP_REDE_HOST=$(echo "$IP_HOST" | cut -d'.' -f1-3)
IP_REDE_SERVIDOR=$(echo "$IP_SERVIDOR_DNS" | cut -d'.' -f1-3)

if [[ "$IP_REDE_HOST" != "$IP_REDE_SERVIDOR" ]]; then
    echo ""
    echo "AVISO: O IP $IP_HOST não está na mesma rede que o servidor ($IP_REDE_SERVIDOR.0/24)!"
    read -p "Deseja continuar mesmo assim? (y/n): " CONTINUAR
    if [[ "$CONTINUAR" != "y" && "$CONTINUAR" != "Y" ]]; then
        echo "Operação cancelada."
        continue
    fi
fi
```

O que faz: Extrai os primeiros 3 octetos do IP fornecido e compara com o IP do servidor.

Passo 21.1.3 - Incrementar o Serial da Zona

```
NOVO_SERIAL=$(date +%s)

sudo sed -i "s/${SERIAL_DATE}/${NOVO_SERIAL}/" /var/named/${DOMINIO}.db
SERIAL_DATE=$NOVO_SERIAL
```

O que faz: Atualiza o número de série para que outros servidores DNS saibam que a zona mudou.

Passo 21.1.4 – Adicionar o Registo A ao Ficheiro da Zona

```
echo "${NOME_HOST}  IN  A       ${IP_HOST}" | sudo tee -a /var/named/${DOMINIO}.db >/dev/null

echo ""
echo "Registo adicionado com sucesso!"
echo "${NOME_HOST}.${DOMINIO} → ${IP_HOST}"
```

O que faz: Anexa uma nova linha ao ficheiro com o registo DNS tipo A.

Passo 21.1.5 – Validar a Zona após alteração

```
if sudo named-checkzone ${DOMINIO} /var/named/${DOMINIO}.db >/dev/null 2>&1; then

    sudo rndc reload
    echo "Zona recarregada com sucesso!"

else

    echo "Erro 21.1.5! Zona direta inválida após alteração!"
    echo "A reverter alterações..."

    # Remover a última linha adicionada em caso de erro

    sudo sed -i '$ d' /var/named/${DOMINIO}.db
fi

sleep 0.5
;;
```

O que faz: Verifica se o ficheiro de zona está correto após a adição do novo registo.

Passo 21.1.6 – Adicionar Registo à Zona Inversa (PTR Record)

```
echo ""
echo "--- Adicionar IP para Nome (zona inversa) ---"
echo ""
read -p "Último octeto do IP (ex: para 192.168.0.20, digite 20): " ULTIMO_OCTETO
read -p "Nome completo do host (ex: pc1.${DOMINIO}): " NOME_COMPLETO
```

O que faz: Adiciona um registo que permite resolver IP para nome.

Passo 21.1.7 – Adição do ponto final

```
if [[ ! "$NOME_COMPLETO" =~ \.$ ]]; then
    NOME_COMPLETO="${NOME_COMPLETO}."
fi
```

O que faz: Garante que o FQDN (Fully Qualified Domain Name) termina com ponto.

Passo 21.1.8 – Incrementação da Serial da Zona Inversa (Reverse Zone)

```
NOVO_SERIAL=$(date +%s)
sudo sed -i "s/${SERIAL_DATE}/${NOVO_SERIAL}/" /var/named/${OCTETO_3}.${OCTETO_2}.${OCTETO_1}.db
SERIAL_DATE=$NOVO_SERIAL
```

O que faz: Atualiza o número de série para que outros servidores DNS saibam que a zona mudou.

Passo 21.1.9 – Adicionar o Registo PTR ao Ficheiro da Zona Inversa

```
echo "${ULTIMO_OCTETO}  IN  PTR   ${NOME_COMPLETO}" | sudo tee -a /var/named/${OCTETO_3}.${OCTETO_2}.${OCTETO_1}.db >/dev/null

echo ""
echo "Registo inverso adicionado com sucesso!"
echo "  ${OCTETO_1}.${OCTETO_2}.${OCTETO_3}.${ULTIMO_OCTETO} -> ${NOME_COMPLETO}"
```

O que faz: Anexa uma nova linha ao ficheiro com o registo DNS tipo PTR.

Passo 21.1.10 – Validar a Zona Inversa após alteração

```
if sudo named-checkzone ${REVERSE_ZONE_ID} /var/named/${OCTETO_3}.${OCTETO_2}.${OCTETO_1}.db >/dev/null 2>&1; then
    sudo rndc reload
    echo "Reverse zone recarregada com sucesso!"

else
    echo "Erro 21.1.10! Reverse zone inválida após alteração!"
    echo "  A reverter alterações..."
    sudo sed -i '$ d' /var/named/${OCTETO_3}.${OCTETO_2}.${OCTETO_1}.db
fi

sleep 0.5
;;
```

O que faz: Verifica se o ficheiro de zona inversa está correto após a adição do novo registo.

Passo 21.1.11 – Ver Registos existentes

```
sudo cat /var/named/${DOMINIO}.db | grep -v "^;" | grep -v "^$" | grep -E "IN\s+(A|NS|CNAME|MX)"

echo ""
echo "--- Zona Inversa (${REVERSE_ZONE_ID}) ---"
sudo cat /var/named/${OCTETO_3}.${OCTETO_2}.${OCTETO_1}.db | grep -v "^;" | grep -v "^$" | grep "PTR"

echo ""
read -p "Prima ENTER para continuar..."
;;


echo ""
echo "A sair da gestão de registos..."
break
;;


echo ""
echo "Opção inválida! Tente novamente."
sleep 0.5
;;
```

O que faz: Mostra os registos DNS atualmente configurados nas zonas.

Passo 22 – Menu de Verificações Opcionais

```
echo ""
echo "Deseja executar verificações finais? (y/n): "
read -p "Resposta: " FAZER_VERIFICACOES
```

O que faz: Oferece ao utilizador opções para testar o servidor DNS configurado.

Passo 22.1 – Menu de Verificações DNS

```
if [[ "$FAZER_VERIFICACOES" == "y" || "$FAZER_VERIFICACOES" == "Y" ]]; then
    echo ""
    echo "1) Testar resolução direta (nome -> IP);"
    echo "2) Testar resolução inversa (IP -> nome);"
    echo "3) Ver status do serviço;"
    echo "4) Testar conectividade Internet;"
    echo "5) Sair."
    echo ""
    read -p "Escolha uma opção (1-5): " OPCAO_VERIFICACAO_DNS
```

O que faz: Apresenta várias opções de teste para o servidor DNS.

Passo 22.1.1 – Estrutura Case para Opções de Verificação

```
case $OPCAO_VERIFICACAO_DNS in
```

O que faz: Executa diferentes comandos de teste baseados na escolha do utilizador.

Passo 22.1.2 – Teste de Resolução Direta

```
1)
    echo ""
    echo "--- Teste de Resolução Direta ---"

    # 22.1.2 - Teste de resolução direta
    # O que faz: Consulta o servidor DNS para resolver nomes de domínio para endereços IP.
    # O que faz o dig: Ferramenta de query DNS que consulta registos DNS.

    dig ${DOMINIO}
    dig ns.${DOMINIO}
    ;;
```

O que faz: Consulta o servidor DNS para resolver nomes de domínio para endereços IP.

Passo 22.1.3 – Teste de Resolução Inversa

```
2)
    echo ""
    echo "--- Teste de Resolução Inversa ---"

    # 22.1.3 - Teste de resolução inversa
    # O que faz: Consulta o servidor DNS para resolver endereços IP para nomes de domínio.

    # O que faz o -x: Opção do dig que realiza uma query de resolução inversa (IP para nome).

    dig -x ${IP_SERVIDOR_DNS}
    ;;
```

O que faz: Consulta o servidor DNS para resolver endereços IP para nomes de domínio.

Passo 21.1.4 – Ver status do serviço BIND

```
3)
    echo ""
    echo "--- Status do Serviço BIND ---"

    # 22.1.4 - Ver status do serviço BIND
    # O que faz: Mostra o estado atual do serviço BIND (named).

    sudo systemctl status named
    ;;
```

O que faz: Mostra o estado atual do serviço BIND (named).

Passo 21.1.5 – Teste de Conectividade à Internet via DNS

```
4)
    echo ""
    echo "--- Teste de Conectividade Internet ---"

    # 22.1.5 - Teste de conectividade à Internet via DNS
    # O que faz: Testa se o forwarder 8.8.8.8 está a funcionar corretamente.

    dig google.com
    ;;
```

O que faz: Testa se o forwarder 8.8.8.8 está a funcionar corretamente.

Passo 21.1.6 – Sair do Menu de Verificações

```
5)
    echo ""
    echo "A sair sem verificações."

    # 22.1.6 - Sair do menu de verificações
    # O que faz: Encerra o loop do menu de verificações.

    break
    ;;
```

O que faz: Encerra o loop do menu de verificações.

Passo 21.1.7 – Opção Inválida

```
*)
    echo ""
    echo "Opção inválida. A sair sem verificações."

    # 22.1.7 - Opção inválida
    # O que faz: Encerra o loop do menu de verificações em caso de entrada inválida.

    break
    ;;
```

O que faz: Encerra o loop do menu de verificações em caso de entrada inválida.

Restantes Opções

```
3)
    echo ""
    echo "A sair do script. Adeus!"
    exit 0
    ;;

*)
    echo ""
    echo "Opção inválida. A sair do script."
    exit 1
    ;;
```

Passo 23 – Mensagens Finais e Comandos Úteis

```
echo ""
echo "========================================"
echo "   COMANDOS ÚTEIS PARA O FUTURO"
echo "========================================"
echo ""
echo "- Testar DNS interno: dig ${DOMINIO}"
echo "- Testar Internet: dig google.com"
echo "- Ver logs: tail -f /var/log/named/bind_queries.log"
echo "- Status: systemctl status named"
echo "- Reiniciar BIND: sudo systemctl restart named"
echo ""
echo "Recomenda-se um reboot do sistema para garantir que todas as alterações tenham efeito."
echo "Para reiniciar o sistema, execute: reboot"
```

O que faz: Exibe comandos úteis para o utilizador após a conclusão do script.

Script DHCP-KEA

O script do DHCP (Kea) realiza a instalação do serviço e a criação do ficheiro de configuração em formato JSON, onde são definidas a sub-rede, o pool de endereços dinâmicos, o gateway, o servidor DNS e eventuais reservas estáticas.

Depois, ativa o serviço, aplica as regras necessárias na firewall e valida o funcionamento, verificando a atribuição de leases a clientes na rede e os respetivos registos de log.

```
set -e
```

O que faz: O set -e é um comando a correr em Scripts para que este pare a execução se houver algum erro em algum dos comandos presentes no Script.

```
chmod 775 config_kea.sh
```

O que faz: Define permissões de leitura, escrita e execução para o proprietário e grupo, e leitura e execução para outros.

Passo 1 – Instalação do *Service*

```
sudo dnf install -y kea
```

O que faz: Instala o servidor DHCP KEA usando o gestor de pacotes DNF. Ao contrário de DHCP tradicional (dhcpd), o KEA vai usar o dnf para instalação.

Passo 2 – Criação do Backup Config

```
if [ -f /etc/kea/kea-dhcp4.conf ] && [ ! -e /etc/kea/kea-dhcp4.conf.backup ]; then
    sudo cp /etc/kea/kea-dhcp4.conf /etc/kea/kea-dhcp4.conf.backup
fi
```

O que faz: Cria uma cópia de segurança do ficheiro de configuração original do KEA DHCP4, caso este exista e ainda não tenha sido feito um backup.

Passo 3 - Validação do IP da Máquina

```
while true; do

    read -p "Digite o IP desejado para o Servidor (Inserir unicamente IPs de Classe C): " IP_SERVIDOR

    TERCEIRO_OCTETO=$(echo "$IP_SERVIDOR" | cut -d'.' -f3)
    QUARTO_OCTETO=$(echo "$IP_SERVIDOR" | cut -d'.' -f4)

    if [[ ! $IP_SERVIDOR =~ ^192\.168\.[0-9]{1,3}\.[0-9]{1,3}$ ]]; then
        echo "Erro 1! IP deve começar com 192.168.x.x."

    elif (( TERCEIRO_OCTETO < 1 || TERCEIRO_OCTETO > 254 )); then
        echo "Erro 2! O 3º octeto deve estar entre 1 e 254."

    elif (( QUARTO_OCTETO < 2 || QUARTO_OCTETO >= 254 )); then
        echo "Erro 3! O 4º octeto deve estar entre 1 e 254."

    else
        echo "IP válido!"
        break
    fi

done
```

O que faz: Verifica se o IP pertence à Classe C (192.168.x.x). Depois, garante que o terceiro octeto do IP não é 0 nem 255.

Passo 4 - Inserção e Validação de IPs

```
VERIFICACAO=""

while [ "$VERIFICACAO" != "y" ] && [ "$VERIFICACAO" != "Y" ]; do
```

O que faz: Solicita ao utilizador todos os IPs necessários (Servidor, Range DHCP, Gateway e DNS), valida-os em tempo real e permite que, caso a validação final não seja confirmada, o utilizador volte a inserir todos os valores novamente.

Passo 4.1 – Solicitar o Escopo de IPs Desejado e Gateway/DNS

```
read -p "Qual vai ser o início do range DHCP (4º octeto)? " OCTETO_INICIO_RANGE
read -p "Qual vai ser o fim do range DHCP (4º octeto)? " OCTETO_FIM_RANGE
read -p "Inserir o nome do domínio (ex: empresa.local): " DOMAIN_NAME
read -p "Inserir o IP do Servidor DNS BIND (o IP estático na LAN Segment): " IP_DNS_BIND
```

O que faz: Pede ao utilizador apenas o 4º octeto do range, gateway e DNS, para formar os IPs completos.

Passo 4.2 – Extrair a Subrede do Servidor

```
IP_SUBNET_SERVIDOR_C=$(echo "$IP_SERVIDOR" | cut -d'.' -f1-3)
```

O que faz: Usa o cut para extrair os primeiros três octetos do IP do servidor, formando a sub-rede.

Passo 4.3 – Criar IPs Completos

```
IP_RANGE_INICIO="${IP_SUBNET_SERVIDOR_C}.${OCTETO_INICIO_RANGE}"
IP_RANGE_FIM="${IP_SUBNET_SERVIDOR_C}.${OCTETO_FIM_RANGE}"
IP_REDE="${IP_SUBNET_SERVIDOR_C}.0"
```

O que faz: Extrai os octetos individuais do IP do servidor e do DNS para facilitar as validações subsequentes.

Passo 4.4 – Octetos Individuais para Validações

```
IP_DNS="${IP_DNS_BIND}"
OCTETO_IP_DNS=$(echo "$IP_DNS" | cut -d'.' -f4)
```

O que faz: Extrai os octetos individuais do IP do servidor e do DNS para facilitar as validações subsequentes.

Passo 4.5 - Cálculo do *Broadcast*

```
IP_BROADCAST="${IP_SUBNET_SERVIDOR_C}.255"
```

O que faz: Calcula o IP de *broadcast* com base no 4º octeto da *gateway*. Se a *gateway* for .1, o *broadcast* será .255, e vice-versa.

Passo 4.6 - Validação do Range DHCP

```
if (( OCTETO_INICIO_RANGE >= OCTETO_FIM_RANGE )); then
    echo "Erro 6! Início do range ($OCTETO_INICIO_RANGE) deve ser menor que o fim ($OCTETO_FIM_RANGE)."
    continue
fi
```

O que faz: Garante que o início do range DHCP é menor que o fim e que o IP do servidor não está dentro do range DHCP.

Passo 4.7 - Validação 2: IP do Servidor fora do range DHCP

```
if (( QUARTO_OCTETO_SERVIDOR >= OCTETO_INICIO_RANGE && QUARTO_OCTETO_SERVIDOR <= OCTETO_FIM_RANGE )); then
    echo "Erro 6! O IP do Servidor não pode estar dentro do range DHCP."
    continue
fi
```

O que faz: Garante que o 4º octeto do IP do servidor não esteja dentro do range DHCP definido.

Passo 4.8 - Validação 3: IP do DNS fora do range DHCP

```
echo -n "[ "

for i in {1..40}; do
    echo -n "="
    sleep 0.2
done

echo " ]"

echo "Resumo dos IPs configurados:"
echo "IP Servidor: $IP_SERVIDOR"
echo "Range DHCP: $IP_RANGE_INICIO - $IP_RANGE_FIM"
echo "IP Gateway: $IP_GATEWAY"
echo "IP DNS: $IP_DNS"
echo "IP Broadcast: $IP_BROADCAST"
echo "IP de Rede: $IP_REDE"
```

O que faz: Garante que o 4º octeto do IP do DNS não esteja dentro do range DHCP definido.

Passo 4.9 – Mostrar Resumo para Configuração

```
echo -n "[ "

for i in {1..40}; do
    echo -n "="
    sleep 0.05
done

echo " ]"

echo ""

echo "Resumo dos IPs configurados (Sub-rede: $IP_SUBNET_SERVIDOR_C):"
echo "-------------------------------------------------------"
echo "IP Servidor Estático: $IP_SERVIDOR (Fora do range DHCP)"
echo "IP Gateway/Router:    $IP_GATEWAY"
echo "IP DNS BIND:          $IP_DNS"
echo "Range DHCP (Início): $IP_RANGE_INICIO"
echo "Range DHCP (Fim):     $IP_RANGE_FIM"
echo "IP Broadcast:         $IP_BROADCAST"
echo "IP de Rede:           $IP_REDE"
echo "Domain Name:         $DOMAIN_NAME"
echo "-------------------------------------------------------"
```

O que faz: Pede ao utilizador para confirmar se os IPs estão corretos. Se a resposta for "y" ou "Y", o loop termina; caso contrário, o utilizador pode reinserir os valores.

Passo 4.10 – Solicitar Confirmação Final

```
read -p "Validação básica concluída! Está tudo correto? (y/n): " VERIFICACAO
```

O que faz: Pede ao utilizador para confirmar se todos os valores estão corretos antes de prosseguir.

Passo 5 – Configuração do Acesso à Internet

```
read -p "Deseja que os clientes tenham acesso à Internet? (y/N): " ACESSO_INTERNET
```

 # O que faz: Pergunta ao utilizador se os clientes DHCP devem ter acesso à Internet. Se sim, configura o gateway e os servidores DNS apropriados.

Passo 5.1 – Definir o Gateway para a rede DHCP

```
GATEWAY="$IP_SERVIDOR"
```

O que faz: Define o gateway para os clientes DHCP como o IP do servidor, assumindo que este atuará como gateway para a rede LAN.

Passo 5.2 - Configurar o DNS do Servidor/Gateway para Acesso à Internet

```
read -p "Insira o IP de DNS EXTERNO para o Servidor/Gateway (Ex: 8.8.8.8 ou 1.1.1.1): " IP_DNS_EXTERNO
```

O que faz: Pede ao utilizador para inserir o IP de um servidor DNS externo (Google ou Cloudflare) para que o servidor/gateway possa resolver nomes de domínio na Internet.

Passo 5.3 - Validação do DNS Externo

```
if [[ "$IP_DNS_EXTERNO" != "8.8.8.8" && "$IP_DNS_EXTERNO" != "1.1.1.1" ]]; then
    echo "Aviso! Valor inválido para DNS Externo. O Servidor/Gateway usará 1.1.1.1 por omissão."
    IP_DNS_EXTERNO="1.1.1.1"
fi
```

O que faz: Verifica se o IP inserido é um dos servidores DNS públicos válidos (Google ou Cloudflare).

Passo 5.4 - Configurar o DNS Primário para os Clientes DHCP

```
read -p "Insira o IP do seu Servidor BIND/DNS na LAN Segment (Será o DNS Primário dos clientes): " IP_DNS_BIND_PRIMARIO

IP_DNS="$IP_DNS_BIND_PRIMARIO"

echo "Acesso à Internet ativado."
echo "Clientes DHCP receberão: Gateway ($GATEWAY) e DNS ($IP_DNS)."
```

O que faz: Pede ao utilizador para inserir o IP do servidor BIND/DNS na LAN, que será usado como DNS primário pelos clientes DHCP.

Passo 5.5 – Configuração para operar sem Acesso à Internet

```
ACESSO=0
GATEWAY=""
IP_DNS_EXTERNO=""

read -p "Insira o IP do seu Servidor BIND/DNS na LAN Segment (Será o único DNS para os clientes): " IP_DNS_BIND_PRIMARIO

IP_DNS="$IP_DNS_BIND_PRIMARIO"
echo "Apenas atribuição de IP local. Clientes DHCP receberão DNS ($IP_DNS)."
```

O que faz: Configura o servidor DHCP para operar sem acesso à Internet, apenas com DNS local (BIND).

Passo 6 – Deteção e Configuração da Placa de Rede

```
read -p "Qual é a interface WAN (NAT)? " WAN_IF
read -p "Qual é a interface LAN (para DHCP)? " LAN_IF

echo ""
echo "A configurar interface WAN ($WAN_IF) para obter IP por DHCP..."
sleep 0.5

sudo nmcli connection add type ethernet ifname "$WAN_IF" con-name WAN ipv4.method auto || true
sudo nmcli connection up WAN || sudo nmcli connection up "$WAN_IF"
```

O que faz: Deteta automaticamente a interface de rede ativa e pede confirmação ao utilizador.

Passo 6.1 – Mostrar IP obtido pela WAN

```
WAN_IP=$(ip -4 addr show "$WAN_IF" | grep inet | awk '{print $2}' | head -n1)
echo "Interface WAN configurada com IP: ${WAN_IP:-"A aguardar DHCP..."}"
```

O que faz: Exibe o endereço IP obtido pela interface WAN após a configuração DHCP.

Passo 6.2 – Configurar LAN com IP do Utilizador

```
echo ""
echo "A configurar interface LAN ($LAN_IF) com IP $IP_SERVIDOR/24..."
sleep 0.5
sudo nmcli connection add type ethernet ifname "$LAN_IF" con-name LAN ipv4.method manual ipv4.addresses "$IP_SERVIDOR/24" || true
sudo nmcli connection modify LAN ipv4.gateway "$IP_GATEWAY" ipv4.dns "$IP_DNS"
sudo nmcli connection up LAN || sudo nmcli connection up "$LAN_IF"

echo ""
echo "Interfaces configuradas:"
sleep 0.5
ip -4 addr show "$WAN_IF"
ip -4 addr show "$LAN_IF"
```

O que faz: Configura a interface LAN com o IP estático fornecido pelo utilizador, juntamente com o gateway e o servidor DNS.

Passo 6.3 – Ativar IP Forwarding

```
echo ""
echo "Ativando IP forwarding..."
sleep 0.5
sudo sysctl -w net.ipv4.ip_forward=1
echo "net.ipv4.ip_forward=1" | sudo tee -a /etc/sysctl.conf > /dev/null
```

O que faz: Ativa o encaminhamento de IP no sistema, permitindo que o tráfego de rede seja roteado entre diferentes interfaces.

Passo 6.4 – Configurar NAT (Masquerading)

```
echo ""
echo "Aplicando regras de NAT..."
sleep 0.5
sudo iptables -t nat -A POSTROUTING -o "$WAN_IF" -j MASQUERADE
sudo iptables -A FORWARD -i "$LAN_IF" -o "$WAN_IF" -j ACCEPT
sudo iptables -A FORWARD -i "$WAN_IF" -o "$LAN_IF" -m state --state RELATED,ESTABLISHED -j ACCEPT
```

O que faz: Configura regras de NAT (Network Address Translation) usando iptables para permitir que os dispositivos na rede LAN acedam à Internet através da interface WAN.

Passo 6.5 – Tornar as Regras Persistentes

```
echo ""
echo "A guardar as regras de NAT para persistirem após reboot."
sleep 0.5
sudo dnf install -y iptables-services
sudo service iptables save
sudo systemctl enable iptables
```

O que faz: Instala o pacote iptables-services para permitir a persistência das regras de iptables após reboot.

Passo 6.6 – Firewall

```
echo ""
echo "A configurar os acessos à firewall..."
sleep 0.5
sudo firewall-cmd --permanent --add-service=dhcp
sudo firewall-cmd --permanent --add-masquerade
sudo firewall-cmd --reload

echo ""
echo "Interfaces e NAT configurados com sucesso!"
sleep 0.5
echo "WAN ($WAN_IF) -> Internet via NAT"
echo "LAN ($LAN_IF) -> IP fixo $IP_SERVIDOR/24 + DHCP KEA"
```

O que faz: Configura a firewall (firewalld) para permitir o serviço DHCP e ativar o masquerading, garantindo que os clientes possam comunicar com o servidor e aceder à Internet.

Passo 7 – Instalação do Fail2Ban

```
echo "A instalar EPEL..."
sudo dnf install -y epel-release

echo "A instalar Fail2Ban e firewalld integration..."
sleep 0.5

sudo dnf install -y fail2ban fail2ban-firewalld

echo "Fail2Ban instalado com sucesso!"
sleep 0.5
```

O que faz: Instala o Fail2Ban e a integração com o firewalld para proteger o servidor contra tentativas de acesso não autorizadas.

Passo 8 – Edição do Config do DHCP (KEA)

```
read -p "Tempo de concessão dos leases desejado, em segundos (Ex: 3600 para 1 hora): " LEASE_TIME

RENEW_TIMER=$((LEASE_TIME / 2))
REBIND_TIMER=$((LEASE_TIME * 7 / 8))

sudo mkdir -p /etc/kea
sudo chmod 755 /etc/kea
sudo chown root:root /etc/kea

sudo mkdir -p /var/log/kea
sudo chmod 755 /var/log/kea
sudo chown root:root /var/log/kea

if [ "$ACESSO" == "1" ]; then
    # Acesso à Internet ativado
    IP_GATEWAY="$IP_SERVIDOR"
    sudo tee /etc/kea/kea-dhcp4.conf << DHCP
{
    "Dhcp4": {
        "interfaces-config": {
            "interfaces": [ "${LAN_IF}" ]
        },
        "expired-leases-processing": {
            "reclaim-timer-wait-time": 10,
            "flush-reclaimed-timer-wait-time": 25,
            "hold-reclaimed-time": 3600,
            "max-reclaim-leases": 100,
            "max-reclaim-time": 250,
            "unwarned-reclaim-cycles": 5
        },
        "renew-timer": ${RENEW_TIMER},
        "rebind-timer": ${REBIND_TIMER},
        "valid-lifetime": ${LEASE_TIME},
        "option-data": [
            {
```

```
                "name": "domain-name-servers",
                "data": "${IP_DNS_BIND}"
            },
            {
                "name": "domain-name",
                "data": "${DOMAIN_NAME}"
            },
            {
                "name": "domain-search",
                "data": "${DOMAIN_NAME}"
            }
        ],
        "subnet4": [
            {
                "id": 1,
                "subnet": "${IP_REDE}/24",
                "pools": [
                    { "pool": "${IP_RANGE_INICIO} - ${IP_RANGE_FIM}" }
                ],
                "option-data": [
                    {
                        "name": "routers",
                        "data": "${IP_GATEWAY}"
                    }
                ]
            }
        ],
        "loggers": [
            {
                "name": "kea-dhcp4",
                "output-options": [
                    { "output": "/var/log/kea/kea-dhcp4.log" }
                ],
                "severity": "INFO",
                "debuglevel": 0
            }
        ]
    }
}
DHCP
```

O que faz: Escreve o ficheiro de configuração JSON do Kea DHCPv4 com os parâmetros fornecidos.

```
else
    # Acesso à Internet desativado
    sudo tee /etc/kea/kea-dhcp4.conf << DHCP
{
    "Dhcp4": {
        "interfaces-config": {
            "interfaces": [ "${LAN_IF}" ]
        },
        "expired-leases-processing": {
            "reclaim-timer-wait-time": 10,
            "flush-reclaimed-timer-wait-time": 25,
            "hold-reclaimed-time": 3600,
            "max-reclaim-leases": 100,
            "max-reclaim-time": 250,
            "unwarned-reclaim-cycles": 5
        },
        "renew-timer": 900,
        "rebind-timer": 1800,
        "valid-lifetime": 3600,
        "option-data": [
            {
                "name": "domain-name",
                "data": "${DOMAIN_NAME}"
            }
        ],
        "subnet4": [
            {
                "id": 1,
                "subnet": "${IP_REDE}/24",
                "pools": [
                    { "pool": "${IP_RANGE_INICIO} - ${IP_RANGE_FIM}" }
                ]
            }
        ],
        "loggers": [
            {
                "name": "kea-dhcp4",
                "output-options": [
                    { "output": "/var/log/kea/kea-dhcp4.log" }
                ],
                "severity": "INFO",
                "debuglevel": 0
            }
        ]
    }
}
DHCP
```

O que faz: Desativa o acesso à Internet.

Passo 8.1 – Permissões dos Ficheiros de Configuração e Log do KEA

```
sudo chmod 644 /etc/kea/kea-dhcp4.conf
sudo chown root:root /etc/kea/kea-dhcp4.conf

sudo touch /var/log/kea-dhcp4.log
sudo chmod 644 /var/log/kea-dhcp4.log
sudo chown root:root /var/log/kea-dhcp4.log

echo "A validar as configurações do Kea DHCP4..."
if ! sudo kea-dhcp4 -t /etc/kea/kea-dhcp4.conf; then
    echo "Erro: Configuração inválida detetada. Por favor, reveja o ficheiro de configuração."
    exit 1
    sleep 0.5
fi
```

O que faz: Define as permissões corretas para o ficheiro de configuração do Kea DHCPv4 e o ficheiro de log, garantindo que apenas o utilizador root tenha acesso de escrita.

Passo 9 – Configuração do Fail2Ban para o KEA

```
sudo tee /etc/fail2ban/filter.d/kea-dhcp.conf >/dev/null << 'EOF'
# ============================================================================ #
# FILTRO FAIL2BAN PARA KEA DHCP (JSON)
# ============================================================================ #

[Definition]

# Usa ADDR para capturar o IP do cliente (quando presente) ou outro identificador.
# Corresponde a logs como: {"message":"DHCPDISCOVER from 00:00:00:00:00:00 via eth0"}
failregex = ^.*"message":\s*"DHCPDISCOVER from (?:<ADDR>|\S+) via \S+".*$
            ^.*"message":\s*"DHCPREQUEST .*from (?:<ADDR>|\S+) via \S+".*$

ignoreregex =

[Init]
# Padrão de data que corresponde ao formato JSON do KEA, essencial para o findtime funcionar.
datepattern = ^"timestamp":\s*"%%Y-%%m-%%d\s+%%H:%%M:%%S\.%%f"
EOF

echo "Ficheiro de filtro /etc/fail2ban/filter.d/kea-dhcp.conf criado."
sleep 0.5
```

O que faz: Configura o Fail2Ban para monitorizar os logs do Kea DHCPv4 e bloquear IPs que apresentem comportamento suspeito, como múltiplas tentativas de DHCPDISCOVER.

Passo 10 – Jail

```
echo ""
echo "A configurar regras de proteção..."

sudo tee /etc/fail2ban/jail.d/kea-dhcp.conf >/dev/null << EOF
# ================================================================================ #
# JAIL KEA DHCP - Configuração Final
# ================================================================================ #

[kea-dhcp]
enabled  = true
filter   = kea-dhcp
port     = 67,68
protocol = udp
logpath  = /var/log/kea/kea-dhcp4.log
backend  = polling
maxretry = 20
findtime = 120
bantime  = 7200

# IPs que NUNCA serão bloqueados: Certificar-se de que a variável é resolvida.
ignoreip = 127.0.0.1/8 ${IP_SERVIDOR}

action   = firewallcmd-ipset
EOF

echo "Configuração criada: /etc/fail2ban/jail.d/kea-dhcp.conf"
sleep 0.5
```

O que faz: Cria uma jail personalizada no Fail2Ban para o Kea DHCP, definindo as regras de monitorização e bloqueio.

Passo 11 – Iniciar o Fail2Ban

```
echo ""
echo "A iniciar o Fail2Ban..."

sudo systemctl enable --now fail2ban

echo "Fail2Ban ativo!"
sleep 0.5

echo "Proteção ativa contra ataques DHCP."
echo "IPs maliciosos serão bloqueados automaticamente."
sleep 0.5
```

O que faz: Ativa e inicia o serviço Fail2Ban, garantindo que ele arranca automaticamente no boot.

Passo 12 – Add do Service à Firewall

```
sudo firewall-cmd --permanent --add-service=dhcp
echo "Serviço adicionado à Firewall."
sleep 0.5

sudo firewall-cmd --runtime-to-permanent
echo "Alterações temporárias aplicadas permanentemente na firewall."
sleep 0.5

sudo systemctl restart firewalld
echo "Serviço firewalld reiniciado."
sleep 0.5
```

O que faz: Configura a firewall (firewalld) para permitir o serviço DHCP (kea/dhcp), garantindo que os clientes podem comunicar com o servidor.

Passo 13 – *Restart* dos *Services*

```
sudo kea-dhcp4 -t /etc/kea/kea-dhcp4.conf
echo "A iniciar o serviço Kea ..."
sleep 0.5

sudo systemctl enable --now kea-dhcp4
echo "Serviço Kea DHCP4 iniciado e pronto para iniciar no arranque."
sleep 0.5

sudo systemctl restart kea-dhcp4
echo "Serviço Kea DHCP4 reiniciado."
sleep 0.5

echo "Recomenda-se um reboot do sistema para garantir que todas as alterações tenham efeito."
echo "Para reiniciar o sistema, execute: reboot"
sleep 0.5
```

O que faz: Inicia o serviço do servidor DHCP (dhcpd) e garante que ele arranca automaticamente no boot. O sudo journalctl é usado para mostrar os logs do serviço, confirmando que o DHCP está ativo e a funcionar.

Passo 14 – Configuração Final

```
while true; do
    echo ""
    echo "Deseja executar verificações finais?"
    echo "1) Verificar status do serviço KEA;"
    echo "2) Ver leases atribuídos;"
    echo "3) Ver últimas linhas do log;"
    echo "4) Verificar status do Fail2Ban;"
    echo "5) Sair."
    echo ""
    read -p "Escolha uma opção (1-5): " OPCAO_VERIFICACAO
```

O que faz: Oferece ao utilizador a opção de executar verificações finais, como verificar o status do serviço, listar os leases atribuídos e visualizar as últimas linhas do log.

Passo 14.1 – Verificar o Status do KEA

```
1)
    echo ""
    echo "--- Status do Serviço Kea DHCP4 ---"

    # 14.1 - Verificar status do serviço KEA
    # O que faz: Mostra o status atual do serviço Kea DHCPv4.

    sudo systemctl status kea-dhcp4
    ;;
```

O que faz: Mostra o status atual do serviço Kea DHCPv4.

Passo 14.2 – Ver Leases Atribuídos

```
2)
    echo ""
    echo "--- Leases Atribuídos ---"

    # 14.2 - Ver leases atribuídos
    # O que faz: Exibe a lista de leases atualmente atribuídos pelo servidor DHCP.

    if [ -f /var/lib/kea/kea-leases4.csv ]; then
        cat /var/lib/kea/kea-leases4.csv
    else
        echo "Ainda não existem leases atribuídos."
    fi
    ;;
```

O que faz: Exibe a lista de leases atualmente atribuídos pelo servidor DHCP.

Passo 14.3 - Ver as Últimas linhas do Log

```
3)
    echo ""
    echo "--- Últimas 10 linhas do Log ---"

    # 14.3 - Ver últimas linhas do log
    # O que faz: Mostra as últimas 10 linhas do ficheiro de log do Kea DHCPv4.

    if [ -f /var/log/kea/kea-dhcp4.log ]; then
        tail -n 10 /var/log/kea/kea-dhcp4.log
    else
        echo "Ficheiro de log ainda não existe."
    fi
    ;;
```

O que faz: Mostra as últimas 10 linhas do ficheiro de log do KEA DHCPv4.

Passo 14.4 – Verificar o Status do Fail2Ban

```
4)
    echo ""
    echo "--- Status do Fail2Ban ---"

    # 14.4 - Verificar status do Fail2Ban
    # O que faz: Mostra o status atual do serviço Fail2Ban e detalhes da jail do Kea DHCP.

    sudo systemctl status fail2ban
    echo ""
    echo "--- Jail KEA DHCP ---"
```

O que faz: Mostra o status atual do serviço Fail2Ban e detalhes da jail do Kea DHCP.

Passo 14.5 – Ver Detalhes da Jail

```
# 14.5 - Ver detalhes da jail do KEA DHCP
# O que faz: Exibe informações específicas sobre a jail do Kea DHCP no Fail2Ban.

sudo fail2ban-client status kea-dhcp
;;
```

O que faz: Exibe informações específicas sobre a jail do Kea DHCP no Fail2Ban.

Restantes Opções

```
5)
    echo ""
    echo "A sair do menu de verificações."
    break
    ;;
*)
    echo ""
    echo "Opção inválida. Por favor, escolha entre 1 e 5."
    ;;
```

Passo 15 – Mensagem Final

```
echo ""
echo "========================================"
echo "   COMANDOS ÚTEIS PARA O FUTURO"
echo "========================================"
echo ""
echo "- Ver leases: cat /var/lib/kea/kea-leases4.csv"
echo "- Ver logs KEA: tail -f /var/log/kea/kea-dhcp4.log"
echo "- Status KEA: systemctl status kea-dhcp4"
echo "- Status Fail2Ban: sudo fail2ban-client status kea-dhcp"
echo "- Ver IPs banidos: sudo fail2ban-client status kea-dhcp"
echo "- Desbanir IP: sudo fail2ban-client set kea-dhcp unbanip <IP>"
echo ""
echo "Recomenda-se um reboot do sistema para garantir que todas as alterações tenham efeito."
echo "Para reiniciar o sistema, execute: reboot"
```

O que faz: Exibe uma mensagem final com comandos úteis para o utilizador.

Checklist de Segurança

Sistema Operativo

- Sistema CentOS 10 atualizado;

Firewall (firewalld)

- firewalld ativa e em execução.
- Porta 53/TCP e UDP aberta para o serviço DNS.
- Porta 67/UDP aberta para o serviço DHCP.
- Apenas serviços essenciais expostos à rede interna.
- Regras permanentes aplicadas e verificadas (firewall-cmd --list-all).

DHCP

- Sub-rede, máscara, gateway e DNS definidos corretamente.
- Pool de endereços dinâmicos sem conflito com IPs estáticos.
- Reserva de IP para o próprio servidor confirmada.
- Ficheiro de leases verificado (/var/lib/kea/dhcp4.leases).
- Log de atividade validado (tail -f /var/log/kea-dhcp4.log).
- Serviço ativo e configurado para iniciar no arranque (systemctl enable kea-dhcp4).

DNS

- Zonas direta e inversa configuradas e validadas (named-checkzone).
- Servidor a responder em modo autoritativo e de cache.
- Forwarders para resolução externa configurados.
- Logging de consultas ativo para auditoria.
- Serviço ativo e configurado para iniciar no arranque (systemctl enable named).

Geral / Boas Práticas

- Backups dos ficheiros originais de configuração guardados.
- Scripts revistos antes da execução (sem linhas inseguras ou comandos destrutivos).
- Testes realizados com clientes Linux e Windows.
- Documentação atualizada após implementação.

Dificuldades

Durante a fase de desenvolvimento e implementação dos *scripts* de servidor (BIND/DNS e KEA/DHCP), o projeto enfrentou desafios notáveis que exigiram uma reestruturação do código e aprofundamento na lógica de rede.

A principal dificuldade incidiu sobre a otimização da experiência do utilizador e a eficiência da entrada de dados. O sistema foi modificado para simplificar a inserção de endereços IP de rede: em vez de pedir o IP completo repetidamente, o script foi reestruturado para solicitar apenas o 4º octeto dos IPs de rede (como *ranges* DHCP e *gateways*), reconstruindo os endereços completos internamente a partir da sub-rede do servidor.

No que respeita à portabilidade do sistema, foi identificado um ponto crítico na dependência de nomes de interfaces de rede fixos (como ens160). Para garantir que os *scripts* pudessem ser executados em qualquer ambiente de Máquina Virtual (VM) de forma universal, foi necessário reestruturar a fase inicial para solicitar os nomes das interfaces diretamente ao utilizador, assegurando a correta configuração do IP estático e do acesso temporário à Internet.

Por fim, o projeto envolveu a integração de medidas de segurança avançadas, nomeadamente o Fail2Ban, tanto para proteger o DNS contra ataques de negação de serviço quanto para proteger o serviço DHCP (KEA). A complexidade desta etapa reside na necessidade de criar filtros e *jails* personalizadas capazes de interpretar os registos de log em formatos específicos (como JSON no KEA), garantindo que apenas o comportamento malicioso é bloqueado sem afetar o serviço legítimo.

Tais desafios resultaram numa solução automatizada não só funcional, mas também mais segura, flexível e eficiente em termos de usabilidade.

Anexos

Conclusão

O presente projeto alcançou com sucesso o seu objetivo fundamental: automatizar a instalação e configuração de serviços essenciais de infraestrutura de rede num ambiente CentOS, nomeadamente o Servidor de Nomes (BIND/DNS) e o Servidor de Atribuição de Endereços (KEA/DHCP).

Os *scripts* desenvolvidos representam soluções completas que vão além da mera instalação de pacotes. Através da linguagem Bash, foi possível centralizar a configuração de redes complexas, como a definição de IPs estáticos, o encaminhamento de tráfego (NAT) e a gestão de zonas DNS, com a garantia de que a segurança é uma prioridade.

Em termos de desempenho e usabilidade, o projeto superou desafios críticos, como a adaptação dinâmica a diferentes interfaces de rede e a otimização da interação do utilizador para simplificar a inserção de dados. A implementação rigorosa do Fail2Ban em ambos o serviço assegura que o ambiente configurado não é apenas funcional, mas também está robustamente protegido contra potenciais ataques de negação de serviço.

Em suma, a automatização provou ser um método eficaz para garantir a rapidez, a consistência e a resiliência na implementação de serviços de rede, fornecendo uma base sólida e segura para o ambiente de servidor.