iptables / ipsets

- ipsets-dynamic script with systemd
- ipsets fqdn resolving

ipsets-dynamic script with systemd

This page shows how to use a BASH script executed on an interval by a SystemD Timer for the purpose of checking entries in specified ipset lists that have a comment of the format "resolve:fqdn - note". The SystemD service runs after the netfilter-persistent.service starts and then periodically based on the timer intervals.

Assumptions

`ipset list -output json` is a new feature. Only works on Ubuntu 24.04 and not previous versions. A workaround will be to install yq and use it to convert xml to json. Quote cleanup in the comment field will be needed.

```
apt install ipset-persistent iptables-persistent netfilter-persistent
```

```
# file:/etc/iptables/ipsets
create dynamicallowlist hash:net family inet hashsize 1024 maxelem 65536 comment
add dynamicallowlist 1.2.3.4 comment "resolve:some-relevant-fqdn.bogus.com - Office - Denver, CO - ATT Fiber"
```

```
# file:/etc/iptables/rules.v4
# super simple example

*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
:WAN-IN ACCEPT [0:0]
:allowed-management - [0:0]

# allowed-management chain
-A allowed-management -s 6.7.8.9 -j ACCEPT -m comment --comment "Static endpoint just in case ipsets don't load properly"
```

```
-A allowed-management -j RETURN

-A INPUT -i lo -j ACCEPT -m comment --comment "Accept all localhost traffic"
-A INPUT -i eth0 -j WAN-IN -m comment --comment "Push inbound WAN traffic to WAN-IN chain"

-A WAN-IN -m set --match-set dynamicallowlist src -j ACCEPT -m comment --comment "Allow all traffic from managed dynamicallowlist"
-A WAN-IN -j allowed-management -m comment --comment "Legacy allowed-management chain... just in case ipsets load fails"
-A WAN-IN -p tcp -m tcp --dport 22 -j DROP -m comment --comment "Drop all other inbound SSH requests"
# list other services you want to explicitely protect

-A WAN-IN -m state --state ESTABLISHED -j ACCEPT
-A WAN-IN -m state --state RELATED -j ACCEPT
-A WAN-IN -m state --state NEW -j DROP -m comment --comment "Drop all other NEW connections"
-A WAN-IN -j DROP -m comment --comment "Drop everything else that might find its way here"
```

Components

```
mkdir -p /opt/ipsets-dynamic/
touch /opt/ipsets-dynamic/ipsets-dynamic /opt/ipsets-dynamic/ipsets-dynamic.service /opt/ipsets-dynamic/ipsets-dynamic.timer
chmod 755 /opt/ipsets-dynamic/ipsets-dynamic
```

```
apt install jq
```

```
# file:/etc/systemd/system/ipsets-dynamic.service

[Unit]
Description=Update ipset entries flagged by resolve comments
After=netfilter-persistent.service
Wants=netfilter-persistent.service

[Service]
Type=oneshot
```

```
ExecStart=/opt/ipsets-dynamic/ipsets-dynamic
Environment="PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
```

```
# file:/etc/systemd/system/ipsets-dynamic.timer

[Unit]
Description=Run ipsets-dynamic every 5 minutes

[Timer]
OnBootSec=1min
OnUnitActiveSec=5min

[Install]
WantedBy=timers.target
```

```
#!/usr/bin/env bash
# Created: 20260205

SYSLOG_TAG="ipsets-dynamic"
IPSETS=("dynamicallowlist")
#DEBUG=1

debug() {
    [[ -n "$DEBUG" ]] && echo "$@"
}

for IPSET_NAME in "${IPSETS[@]}"; do
    debug "================================"
    debug "Processing ipset: $IPSET_NAME"
    ipset list $IPSET_NAME -output json | grep -v initval | jq -c '
    .[0].members[]
    | select(.comment | contains("resolve:"))
    | {
        elem,
        comment,
        fqdn: (.comment | capture("resolve:(?<fqdn>[^ ]+)").fqdn)
        }
    ' |
    while read -r entry; do
        listed_ip=$(echo "$entry" | jq -r '.elem')
```

```
        comment=$(echo "$entry" | jq -r '.comment')
        fqdn=$(echo "$entry" | jq -r '.fqdn')

        debug "--------------------------------"
        debug "IPSET       : $IPSET_NAME"
        debug "FQDN        : $fqdn"
        debug "Comment     : $comment"
        debug "Listed IP   : $listed_ip"

        DIG_RECORD_LINE=$(dig +noall +answer "$fqdn" | grep "IN\s*A")
        if [ -n "$DIG_RECORD_LINE" ]; then
            TTL=$(echo "$DIG_RECORD_LINE" | awk '{print $2}')
            ADDRESS=$(echo "$DIG_RECORD_LINE" | awk '{print $5}')
            debug "Resolved IP : $ADDRESS ($TTL seconds TTL)"
            if [[ "$listed_ip" != "$ADDRESS" ]]; then
                debug ""
                echo "$SYSLOG_TAG - ipset update needed - $IPSET_NAME / $listed_ip -> $ADDRESS
/ $comment"
                logger -i "$SYSLOG_TAG - ipset update needed - $IPSET_NAME / $listed_ip ->
$ADDRESS / $comment"
                ipset del $IPSET_NAME $listed_ip
                if [[ $? -ne 0 ]]; then
                    debug "  Failure deleting ipset entry $IPSET_NAME / $listed_ip for $fqdn"
                    logger -i -p user.error "$SYSLOG_TAG - ipset update failure deleting entry
$IPSET_NAME / $listed_ip / $comment"
                else
                    debug "  Deleted $IPSET_NAME / $listed_ip"
                fi
                ipset add $IPSET_NAME $ADDRESS comment "$comment"
                if [[ $? -ne 0 ]]; then
                    debug "  [ERROR] ipset update failure adding entry $IPSET_NAME / $ADDRESS
for $fqdn"
                    logger -i -p user.error "$SYSLOG_TAG - ipset update failure adding entry
$IPSET_NAME / $ADDRESS / $comment"
                else
                    debug "  Added $IPSET_NAME / $ADDRESS for $fqdn"
                fi
            fi
        else
            debug "No A record found for $fqdn"
```

```
            TTL="N/A"
            ADDRESS="N/A"
        fi
        debug ""
    done
    debug ""
done

# make sure we signal a successful exit for systemd
exit 0
```

```
systemctl daemon-reload

systemctl enable --now ipsets-dynamic.timer

systemctl list-timers ipsets-dynamic.timer
watch systemctl list-timers ipsets-dynamic.timer

journalctl -u ipsets-dynamic.service -n 50
```

ipsets fqdn resolving

Concept

- Use a Sqlite3 database to store the list of entries that need to be updated.
- A script run on an interval will perform lookups for each fqdn, no more frequent than the DNS record's TTL
  - `next_lookup_dt` is set after a lookup to the current time plus the lookup's ttl in seconds
    - `UPDATE entries SET last_lookup_dt = datetime('now'), next_lookup_dt = datetime('now', '+$ttl seconds') WHERE fqdn=$fqdn and ipset=$ipset;`
  - Query for a list of entries that need to be checked due to expired ttl of last lookup
    - `SELECT * FROM entries WHERE next_lookup_dt <= datetime('now');`

We have two options for dealing with changed IPs.

The first involves creating a new ipset, swapping it with the current in-memory set, and then destroying the old ipset. After lookups are performed, the current list of IP addresses in the database would be used to create a new ipset list which would then be swapped and the old list destroyed. This would be effective only if this lookup script is the only thing manipulating the ipset list.

The second involves locating and deleting the entry that needs to be updated. This allows for additional processes to manipulate the list. The output of ipset list would be searched for an entry where the command matches the fqdn lookup. That IP address would be removed from the list using an ipset del command and then the new IP addresses added with the appropriate comment set.

Code

```
# CREATE the sqlite3 database and table(s)
sqlite3 /opt/ipsets-dynamic/ipsets-dynamic.db <<EOF
```

```
CREATE TABLE entries (
    fqdn TEXT NOT NULL,
    ipset TEXT NOT NULL,
    last_ip_address TEXT,
    last_lookup_dt DATETIME DEFAULT (datetime('now')),
    next_lookup_dt DATETIME DEFAULT (datetime('now')),
    last_change_dt DATETIME DEFAULT (datetime('now')),
    added DATETIME DEFAULT (datetime('now')),
    comment TEXT,
    PRIMARY KEY (fqdn, ipset)
);
EOF
```

Example

Assuming the ipset list exists as below:

```
ipset list unifisiteallowlist -output json | grep -v initval
```

```
# OUTPUT
[
  {
    "name" : "unifisiteallowlist",
    "type" : "hash:net",
    "revision" : 7,
    "header" : {
      "family" : "inet",
      "hashsize" : 1024,
      "maxelem" : 65536,
      "comment" : true,
      "bucketsize" : 12,
      "memsize" : 1800,
      "references" : 1,
      "numentries" : 10
    },
    "members" : [
      {
```

```
        "elem" : "1.2.3.4",
        "comment" : "resolve:some-fqdn.domain.com"
      },
   ]
  }
]
```

```
result=$(ipset list unifisiteallowlist -output json | grep -v initval | jq -r '.[].members[] | select(.comment | contains("resolve:some-fqdn.domain.com")) | .elem')
echo $result
```

```
# OUTPUT
1.2.3.4
```

The value of result would be empty if there was no match found.

```
#!/bin/bash

ENTRIESDB='/opt/ipsets-dynamic/ipsets-dynamic.db'

# PROCESS ALL RECORDS
sqlite3 -separator '|' $ENTRIESDB "SELECT fqdn, ipset, last_ip_address FROM entries;" |
while IFS='|' read -r fqdn ipset ip; do
  RECORD_LINE=$(dig +noall +answer "$fqdn" | grep "IN\s*A")
  if [ -n "$RECORD_LINE" ]; then
    TTL=$(echo "$RECORD_LINE" | awk '{print $2}')
    ADDRESS=$(echo "$RECORD_LINE" | awk '{print $5}')

    if [[ "$ip" != "$ADDRESS" ]]; then
      echo "Entry: $fqdn / $ipset / $ip / $ADDRESS / $TTL - Update needed"
      # FIND AND DELETE EXISTING IPSET ENTRY
      echo "ipset del $ipset $ip"
      # CREATE NEW IPSET ENTRY
      echo "ipset add $ipset $ADDRESS comment \"resolve:$fqdn\""
      # UPDATE DATABASE
      echo "sqlite3 $ENTRIESDB \"UPDATE entries SET last_ip_address='$ip',next_lookup_dt=datetime('now','+$TTL seconds') WHERE fqdn='$fqdn' and ipset='$ipset';\""
```

```
    else
      echo "Entry: $fqdn / $ipset / $ip / $ADDRESS / $TTL - NO CHANGE"
    fi
  else
    echo "Entry: $fqdn / $ipset / $ip / NO A RECORD FOUND"
  fi
done
```

```
#!/bin/bash

ENTRIESDB='/opt/ipsets-dynamic/ipsets-dynamic.db'

# PROCESS ONLY TTL EXPIRED RECORDS
sqlite3 -separator '|' $ENTRIESDB "SELECT fqdn, ipset, last_ip_address FROM entries WHERE
next_lookup_dt < datetime('now');" |
while IFS='|' read -r fqdn ipset ip; do
  RECORD_LINE=$(dig +noall +answer "$fqdn" | grep "IN\s*A")
  if [ -n "$RECORD_LINE" ]; then
    TTL=$(echo "$RECORD_LINE" | awk '{print $2}')
    ADDRESS=$(echo "$RECORD_LINE" | awk '{print $5}')

    if [[ "$ip" != "$ADDRESS" ]]; then
      echo "Entry: $fqdn / $ipset / $ip / $ADDRESS / $TTL - Update needed"
      # FIND AND DELETE EXISTING IPSET ENTRY
      echo "ipset del $ipset $ip"
      # CREATE NEW IPSET ENTRY
      echo "ipset add $ipset $ADDRESS comment \"resolve:$fqdn\""
      # UPDATE DATABASE
      echo "sqlite3 $ENTRIESDB \"UPDATE entries SET
last_ip_address='$ip',next_lookup_dt=datetime('now','+$TTL seconds') WHERE fqdn='$fqdn' and
ipset='$ipset';\""
    else
      echo "Entry: $fqdn / $ipset / $ip / $ADDRESS / $TTL - NO CHANGE"
    fi
  else
    echo "Entry: $fqdn / $ipset / $ip / NO A RECORD FOUND"
  fi
done
```