Determine the best server for OPNsense updates and upgrades

Wherever you are on the planet, there is a need to determine, which server out of many is accessible and which has the lowest latency.

Let's create a mirros list

```
vi mirrors.list
```

Mirrors (this one was reverse engineered from official OPNsense page)

```
https://ftp.cc.uoc.gr/mirrors/opnsense/
https://mirror-opnsense.serverbase.ch/
https://mirror.aarnet.edu.au/pub/opnsense/
https://mirror.ams1.nl.leaseweb.net/opnsense/
https://mirror.catalyst.net.nz/opnsense/
https://mirror.cedia.org.ec/opnsense/
https://mirror.cloudfence.com.br/opnsense/
https://mirror.cs.odu.edu/opnsense/
https://mirror.distly.kr/opnsense
https://mirror.dns-root.de/opnsense/
https://mirror.eliv.digital/opnsense/
https://mirror.fra10.de.leaseweb.net/opnsense/
https://mirror.hemino.net/opnsense/
https://mirror.init7.net/opnsense/
https://mirror.keiminem.com/opnsense/
https://mirror.koreapixel.kr/opnsense/
https://mirror.krfoss.org/opnsense/
https://mirror.leitecastro.com/opnsense/
https://mirror.level66.network/opnsense-dist/
https://mirror.marwan.ma/opnsense/
https://mirror.meowsmp.net/opnsense/
```

```
https://mirror.mtl2.ca.leaseweb.net/opnsense/
https://mirror.ntct.edu.tw/opnsense/
https://mirror.pangkin.com/opnsense/
https://mirror.raiolanetworks.com/opnsense/
https://mirror.serverion.com/opnsense/
https://mirror.sfo12.us.leaseweb.net/opnsense/
https://mirror.techlabs.co.kr/opnsense/
https://mirror.ueb.edu.ec/opnsense/
https://mirror.uvensys.de/opnsense/
https://mirror.venturasystems.tech/opnsense/
https://mirror.verinomi.com/opnsense/
https://mirror.vraphim.com/opnsense/
https://mirror.wdc1.us.leaseweb.net/opnsense/
https://mirror.winsub.kr/opnsense/
https://mirror.zyner.org/mirror/opnsense/
https://mirror.zzunipark.com/opnsense/
https://mirror1.isatisidc.ir/opnsense/
https://mirrors.dotsrc.org/opnsense/
https://mirrors.hopbox.net/opnsense/
https://mirrors.komogoto.com/opnsense/
https://mirrors.nycbug.org/pub/opnsense/
https://mirrors.ocf.berkeley.edu/opnsense/
https://mirrors.pku.edu.cn/opnsense/
https://opnsense-mirror.hiho.ch/
https://opnsense.aivian.org/
https://opnsense.c0urier.net/
https://opnsense.org/download/
https://pkg.opnsense.org/
https://www.mirrorservice.org/sites/opnsense.org/
```

Once that done, let's create a script to ping all of them

```
vi url-speed-rank.sh
```

bash code

```
#!/usr/bin/env bash
# url-speed-rank.sh
# Combines:
#  - ICMP RTT average per host (ping)
```

```
#  - HTTP(S) total time per URL (curl)
# Outputs a sorted table + summary with the fastest N URLs (default 3).

set -o pipefail

VERBOSE=0
TOPN=3

# Ping knobs
PING_COUNT=3
PING_TIMEOUT=1   # seconds (best-effort portable)

# Curl knobs
CURL_TIMEOUT=5  # seconds

usage() {
  cat >&2 <<EOF
Usage: $0 [-v] [-n TOPN] [-c PING_COUNT] [-p PING_TIMEOUT] [-t CURL_TIMEOUT] <urls_file>

  -v              verbose progress (stderr)
  -n TOPN         how many fastest URLs to summarize (default: ${TOPN})
  -c PING_COUNT   ping packets per host (default: ${PING_COUNT})
  -p PING_TIMEOUT ping timeout per packet, seconds (default: ${PING_TIMEOUT})
  -t CURL_TIMEOUT curl max time per URL, seconds (default: ${CURL_TIMEOUT})

Input file:
  - one URL/host per line
  - empty lines and lines starting with # are ignored
  - if scheme missing, https:// is assumed
EOF
  exit 1
}

log() { [[ "$VERBOSE" -eq 1 ]] && echo "$*" >&2; }

while getopts ":vn:c:p:t:" opt; do
  case "$opt" in
    v) VERBOSE=1 ;;
    n) TOPN="$OPTARG" ;;
    c) PING_COUNT="$OPTARG" ;;
```

```
    p) PING_TIMEOUT="$OPTARG" ;;
    t) CURL_TIMEOUT="$OPTARG" ;;
    *) usage ;;
  esac
done
shift $((OPTIND - 1))

INPUT="$1"
[[ -z "$INPUT" ]] && usage
[[ ! -r "$INPUT" ]] && { echo "Error: cannot read file '$INPUT'" >&2; exit 1; }

tmp_urls="$(mktemp)"
tmp_hosts="$(mktemp)"
tmp_ping="$(mktemp)"
tmp_results="$(mktemp)"

cleanup() { rm -f "$tmp_urls" "$tmp_hosts" "$tmp_ping" "$tmp_results"; }
trap cleanup EXIT

# 1) Normalize input → list of URLs (deduped)
#    - skip empty/comments
#    - prepend https:// if missing scheme
awk '
  /^[[:space:]]*$/ { next }
  /^[[:space:]]*#/ { next }
  {
    gsub(/^[[:space:]]+|[[:space:]]+$/, "", $0)
    url=$0
    if (url !~ /^https?:\/\//) url="https://" url
    if (!seen[url]++) print url
  }
' "$INPUT" > "$tmp_urls"

TOTAL_URLS="$(wc -l < "$tmp_urls" | tr -d ' ')"
[[ "$TOTAL_URLS" -eq 0 ]] && { echo "No URLs to test (after filtering comments/empty lines)." >&2; exit 1; }

# 2) Extract unique hosts from URLs
#    host = strip scheme, then cut at first /, :, ?, #
sed -E 's|^[a-zA-Z]+://||; s|[/?#].*$||; s|:.*$||' "$tmp_urls" | sort -u > "$tmp_hosts"
```

```
# 3) Ping each host once → avg RTT (ms) or 999999 on failure
ping_avg_ms() {
  local host="$1"
  local out

  # Try Linux-style first, then alternate (some environments differ).
  if out="$(ping -c "$PING_COUNT" -W "$PING_TIMEOUT" "$host" 2>/dev/null)"; then
    :
  elif out="$(ping -c "$PING_COUNT" -t "$PING_TIMEOUT" "$host" 2>/dev/null)"; then
    :
  else
    echo "999999"
    return 0
  fi

  # Linux: "rtt min/avg/max/mdev = 10.1/11.2/..."
  # macOS: "round-trip min/avg/max/stddev = 10.1/11.2/..."
  awk -F'/' '/round-trip|rtt/ {print $5; found=1} END{ if(!found) print "999999" }' <<<"$out"
}

log "[*] Pinging hosts ($(wc -l < "$tmp_hosts" | tr -d ' ') unique) ..."
while IFS= read -r host; do
  [[ -z "$host" ]] && continue
  log "    ping: $host"
  avg="$(ping_avg_ms "$host")"
  printf "%s\t%s\n" "$host" "$avg" >> "$tmp_ping"
done < "$tmp_hosts"

# 4) Test each URL with curl → time_total + http_code; join ping by host
log "[*] Curling URLs (${TOTAL_URLS}) ..."
while IFS= read -r url; do
  log "    curl: $url"

  host="$(sed -E 's|^[a-zA-Z]+://||; s|[/?#].*$||; s|:.*$||' <<<"$url")"
  ping_ms="$(awk -F'\t' -v h="$host" '$1==h{print $2; found=1} END{if(!found) print "999999"}' "$tmp_ping")"

  # We capture both time_total and http_code. On hard error, curl prints nothing; treat as failure.
  curl_out="$(curl \
    --silent \
    --output /dev/null \
```

```
    --location \
    --max-time "$CURL_TIMEOUT" \
    --write-out "%{time_total}\t%{http_code}" \
    "$url" 2>/dev/null || true)"

  curl_time="$(awk -F'\t' '{print $1}' <<<"$curl_out")"
  http_code="$(awk -F'\t' '{print $2}' <<<"$curl_out")"

  if [[ -z "$curl_time" || "$curl_time" == "0.000"* ]]; then
    curl_time="999.999"
    http_code="${http_code:-000}"
    status="FAIL"
  else
    # Consider 2xx/3xx as OK
    if [[ "$http_code" =~ ^2|^3 ]]; then
      status="OK"
    else
      status="HTTP_$http_code"
    fi
  fi

  # Sort key first (curl seconds), then printable columns
  # Columns: CURL_S  PING_MS  HTTP  STATUS  URL
  printf "%08.3f\t%10.2f\t%3s\t%-8s\t%s\n" \
    "$curl_time" "$ping_ms" "$http_code" "$status" "$url" >> "$tmp_results"
done < "$tmp_urls"

# 5) Print ranked table
echo
echo "Ranked (fastest → slowest) by HTTP total time:"
printf "%-10s %-10s %-4s %-8s %s\n" "CURL_S" "PING_MS" "HTTP" "STATUS" "URL"
printf "%-10s %-10s %-4s %-8s %s\n" "------" "-------" "----" "------" "---"
sort -n "$tmp_results" | awk -F'\t' '{printf "%-10s %-10s %-4s %-8s %s\n",$1,$2,$3,$4,$5}'

# 6) Summary stats + fastest TOPN URLs
echo
echo "Summary:"
unique_hosts="$(wc -l < "$tmp_hosts" | tr -d ' ')"
ping_fail_hosts="$(awk -F'\t' '$2>=999999 {c++} END{print c+0}' "$tmp_ping")"
curl_fail_urls="$(awk -F'\t' '$1>=999.999 || $4=="FAIL" {c++} END{print c+0}' "$tmp_results")"
```

```
ok_urls="$(awk -F'\t' '$4=="OK" {c++} END{print c+0}' "$tmp_results")"

# Average curl time across successful URLs (status OK)
avg_ok_curl="$(awk -F'\t' '$4=="OK"{sum+=$1; n++} END{ if(n) printf "%.3f", sum/n; else print "n/a" }'
"$tmp_results")"

echo "  URLs tested:           $TOTAL_URLS"
echo "  Unique hosts:          $unique_hosts"
echo "  OK URLs (2xx/3xx):     $ok_urls"
echo "  Curl failures/timeout: $curl_fail_urls"
echo "  Ping failures:         $ping_fail_hosts"
echo "  Avg curl time (OK):    ${avg_ok_curl}s"

echo
echo "Fastest ${TOPN} URL(s) to use for update fetching (by HTTP total time):"
sort -n "$tmp_results" | head -n "$TOPN" | awk -F'\t' '{printf "  %s  (curl=%ss, ping=%sms, http=%s,
%s)\n",$5,$1,$2,$3,$4}'
```

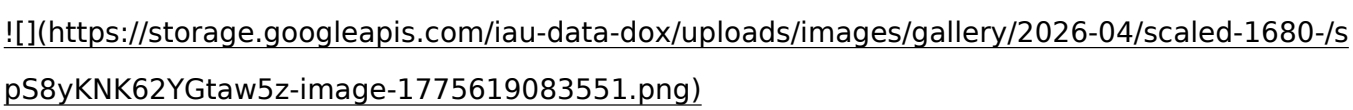
Make executale

```
chmod +x ./url-speed-rank.sh
```

And run it

```
./url-speed-rank.sh -v mirrors.list
```

Results:



Let's see result more closer

```
Ranked (fastest → slowest) by HTTP total time:
CURL_S     PING_MS    HTTP STATUS   URL
------     -------    ---- ------   ---
0000.092    999999.00 200  OK       https://mirror.verinomi.com/opnsense/
0000.221        24.05 403  HTTP_403 https://mirror.krfoss.org/opnsense/
0000.232    999999.00 200  OK       https://mirror-opnsense.serverbase.ch/
0000.233        74.23 200  OK       https://mirror.zyner.org/mirror/opnsense/
0000.234        74.60 200  OK       https://mirror.uvensys.de/opnsense/
0000.242        77.44 200  OK       https://mirror.init7.net/opnsense/
0000.245        78.98 200  OK       https://opnsense-mirror.hiho.ch/
0000.248        78.39 200  OK       https://mirrors.dotsrc.org/opnsense/
0000.256    999999.00 200  OK       https://mirror.vraphim.com/opnsense/
0000.277        74.71 200  OK       https://mirror.dns-root.de/opnsense/
0000.280        63.05 200  OK       https://mirror.fra10.de.leaseweb.net/opnsense/
0000.281        67.35 200  OK       https://mirror.serverion.com/opnsense/
0000.293        81.80 200  OK       https://opnsense.c0urier.net/
0000.318        63.73 200  OK       https://pkg.opnsense.org/
0000.336        76.12 200  OK       https://mirror.ams1.nl.leaseweb.net/opnsense/
0000.339        99.32 200  OK       https://mirror.marwan.ma/opnsense/
0000.369        28.55 404  HTTP_404 https://mirror.eliv.digital/opnsense/
0000.378        80.56 200  OK       https://mirror.raiolanetworks.com/opnsense/
0000.385        30.88 200  OK       https://mirror.techlabs.co.kr/opnsense/
0000.402        79.23 200  OK       https://mirrors.komogoto.com/opnsense/
0000.424        29.98 200  OK       https://mirror.winsub.kr/opnsense/
0000.446       145.98 200  OK       https://mirrors.nycbug.org/pub/opnsense/
0000.471    999999.00 200  OK       https://mirror.cs.odu.edu/opnsense/
0000.497        64.11 200  OK       https://opnsense.org/download/
0000.514        74.20 200  OK       https://mirror.venturasystems.tech/opnsense/
0000.579        71.70 200  OK       https://www.mirrorservice.org/sites/opnsense.org/
0000.599       141.55 200  OK       https://mirror.mtl2.ca.leaseweb.net/opnsense/
0000.601       149.73 200  OK       https://mirror.wdc1.us.leaseweb.net/opnsense/
0000.632       272.29 000  HTTP_000 https://mirror.cloudfence.com.br/opnsense/
0000.650       211.07 200  OK       https://mirrors.ocf.berkeley.edu/opnsense/
0000.738       230.49 200  OK       https://mirrors.hopbox.net/opnsense/
0000.804    999999.00 200  OK       https://ftp.cc.uoc.gr/mirrors/opnsense/
0000.814       197.82 200  OK       https://mirror.sfo12.us.leaseweb.net/opnsense/
0000.835        76.97 200  OK       https://mirror.level66.network/opnsense-dist/
0000.888    999999.00 200  OK       https://mirror.hemino.net/opnsense/
0000.929    999999.00 200  OK       https://mirrors.pku.edu.cn/opnsense/
0000.952       305.30 200  OK       https://mirror.koreapixel.kr/opnsense/
0001.005       240.86 200  OK       https://mirror.ueb.edu.ec/opnsense/
0001.033        73.53 200  OK       https://mirror.meowsmp.net/opnsense/
0001.086       345.24 200  OK       https://mirror.aarnet.edu.au/pub/opnsense/
0001.098       355.62 200  OK       https://mirror.catalyst.net.nz/opnsense/
0001.428    999999.00 200  OK       https://mirror.keiminem.com/opnsense/
0001.818       243.04 200  OK       https://mirror.cedia.org.ec/opnsense/
0001.827       312.75 200  OK       https://mirror.distly.kr/opnsense
0002.420       333.21 200  OK       https://mirror.pangkin.com/opnsense/
0002.438       308.37 200  OK       https://opnsense.aivian.org/
0004.797        28.68 200  OK       https://mirror.zzunipark.com/opnsense/
0005.002    999999.00 000  HTTP_000 https://mirror1.isatisidc.ir/opnsense/
0005.003    999999.00 000  HTTP_000 https://mirror.leitecastro.com/opnsense/
0005.004    999999.00 000  HTTP_000 https://mirror.ntct.edu.tw/opnsense/

Summary:
  URLs tested:           50
  Unique hosts:          50
  OK URLs (2xx/3xx):     0
  Curl failures/timeout: 0
  Ping failures:         11
  Avg curl time (OK):    n/as

Fastest 3 URL(s) to use for update fetching (by HTTP total time):
  https://mirror.verinomi.com/opnsense/  (curl=0000.092s, ping= 999999.00ms, http=200, OK      )
  https://mirror.krfoss.org/opnsense/  (curl=0000.221s, ping=     24.05ms, http=403, HTTP_403)
  https://mirror-opnsense.serverbase.ch/  (curl=0000.232s, ping= 999999.00ms, http=200, OK      )
```

First column indicates what time it took to load the webpage, There are hosts with 999999 ms of ping , which simply means ICMP is not permitted. Hosts with codes 404 are empty or incorrect.

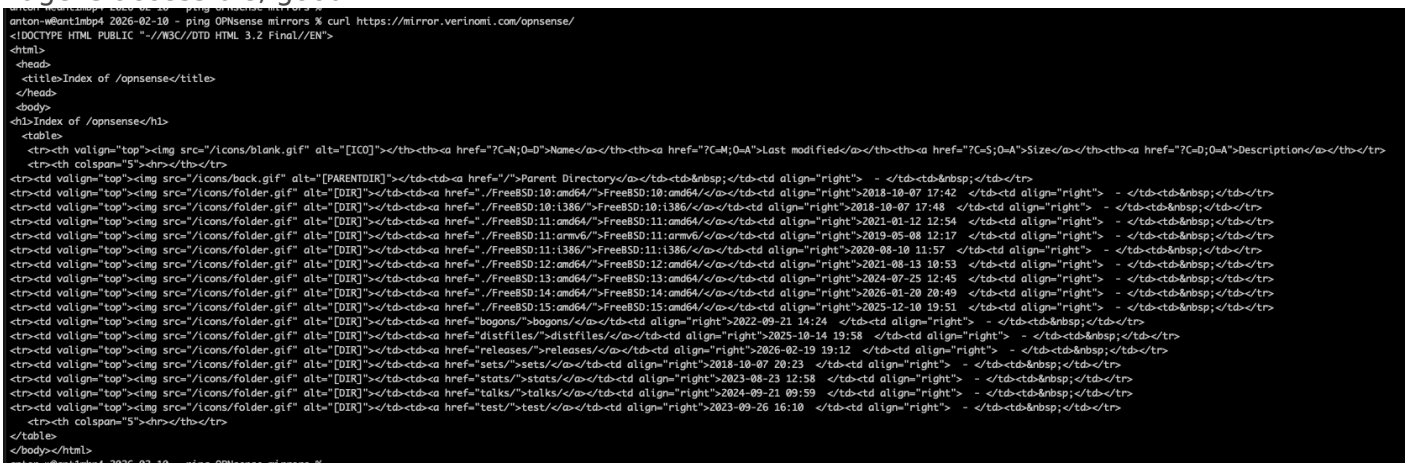
Short summary in the end gives an idea, which mirrors to focus on

Same script can be used to determine any other mirrors' list reachability.

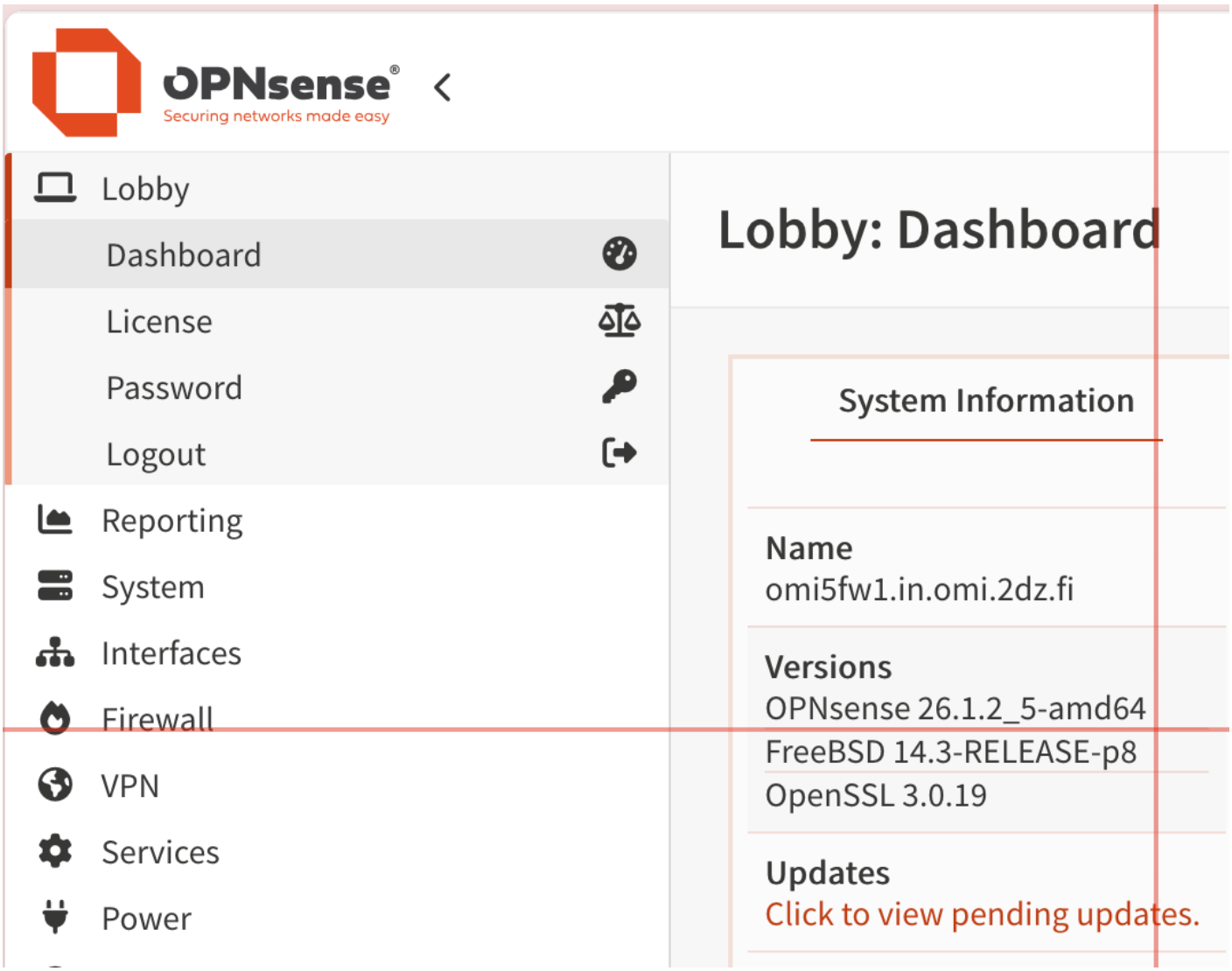
Let's perform manual verification

```
curl https://mirror.verinomi.com/opnsense/
```

Page is accessible, good.



On the day of writing (2026-04-08), current major version is 26 running on FreeBSD v14

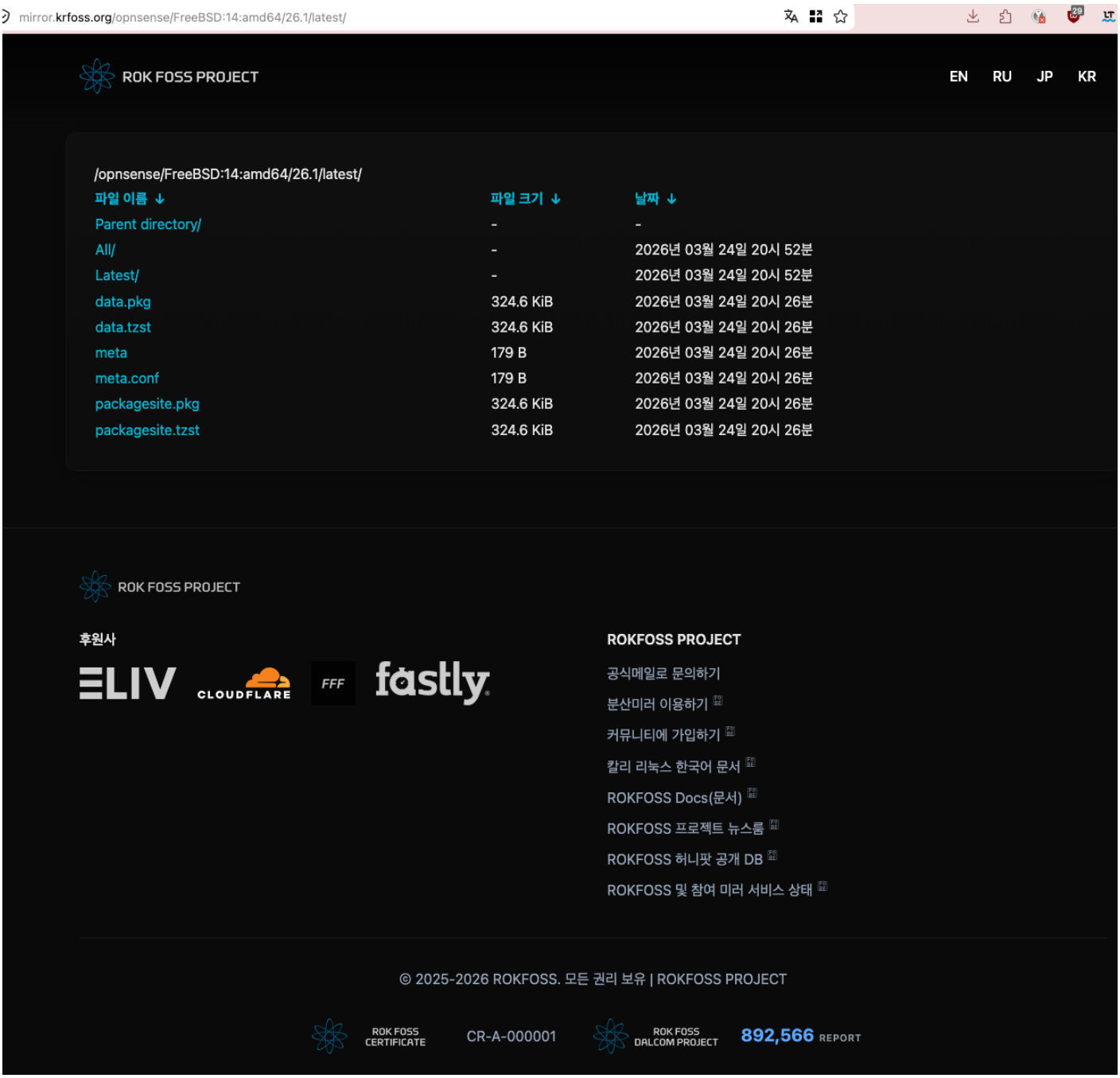


Let's check that mirror will have recent upgrades available on that paricular mirror

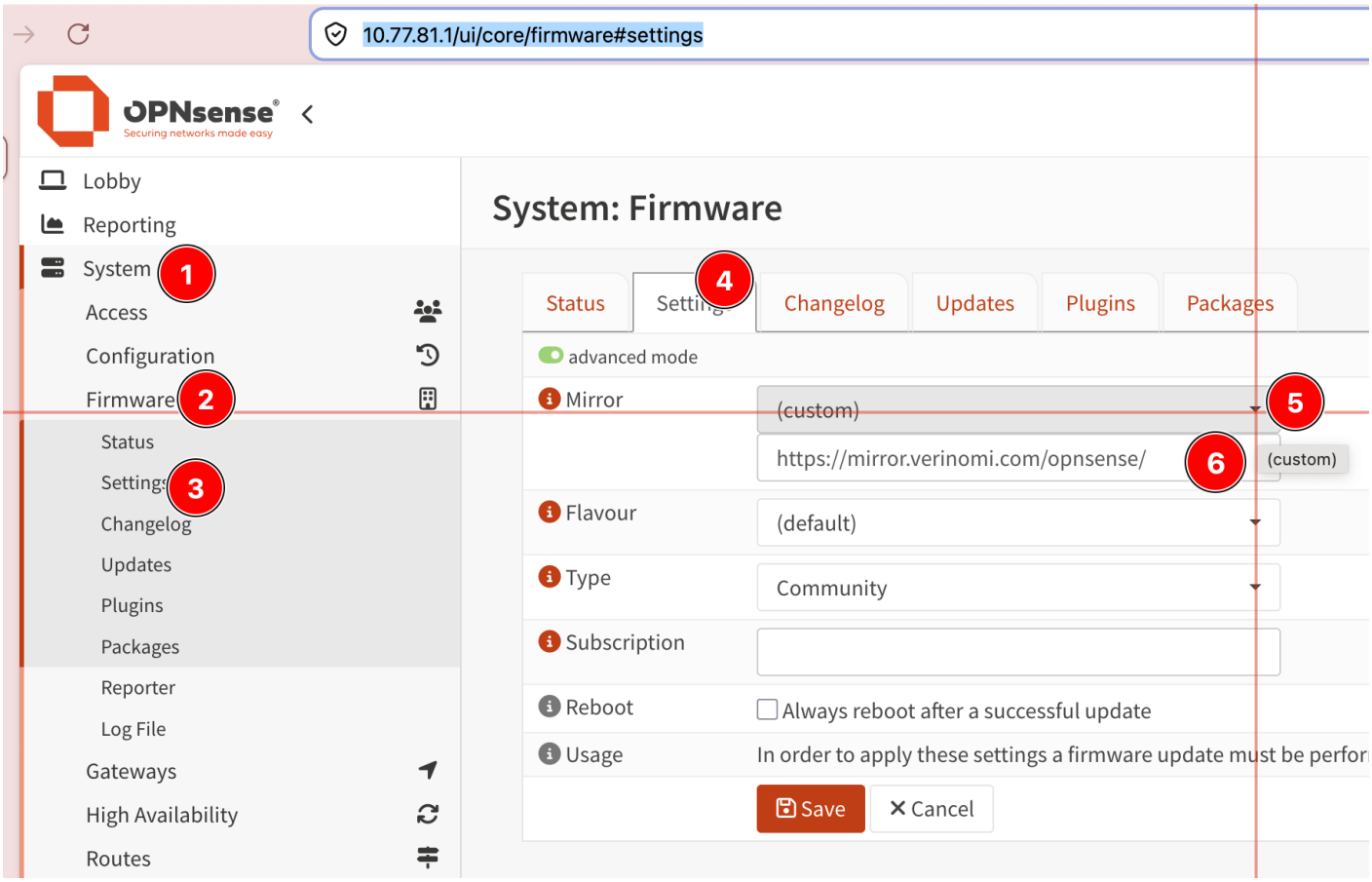
This tells me, that packages are two weeks old, which is good.

```
anton-w@ant1mbp4 2026-02-10 - ping OPNsense mirrors % curl https://mirror.verinomi.com/opnsense/FreeBSD:14:amd64/26.1/latest/
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<html>
 <head>
  <title>Index of /opnsense/FreeBSD:14:amd64/26.1/latest</title>
 </head>
 <body>
<h1>Index of /opnsense/FreeBSD:14:amd64/26.1/latest</h1>
  <table>
   <tr><th valign="top"><img src="/icons/blank.gif" alt="[ICO]"></th><th><a href="?C=N;O=D">Name</a></th><th><a href="?C=M;O=A">Last modified</a></th><th><a href="?C=S;O=A">Size</a></th><th><a href="?C=D;O=A">Description</a></th></tr>
   <tr><th colspan="5"><hr></th></tr>
<tr><td valign="top"><img src="/icons/back.gif" alt="[PARENTDIR]"></td><td><a href="/opnsense/FreeBSD:14:amd64/26.1/">Parent Directory</a></td><td> </td><td align="right">  - </td><td> </td></tr>
<tr><td valign="top"><img src="/icons/folder.gif" alt="[DIR]"></td><td><a href="All/">All/</a></td><td align="right">2026-03-24 14:52  </td><td align="right">  - </td><td> </td></tr>
<tr><td valign="top"><img src="/icons/folder.gif" alt="[DIR]"></td><td><a href="Latest/">Latest/</a></td><td align="right">2026-03-24 14:52  </td><td align="right">  - </td><td> </td></tr>
<tr><td valign="top"><img src="/icons/unknown.gif" alt="[   ]"></td><td><a href="data.pkg">data.pkg</a></td><td align="right">2026-03-24 14:26  </td><td align="right">325K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/unknown.gif" alt="[   ]"></td><td><a href="data.tzst">data.tzst</a></td><td align="right">2026-03-24 14:26  </td><td align="right">325K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/unknown.gif" alt="[   ]"></td><td><a href="meta">meta</a></td><td align="right">2026-03-24 14:26  </td><td align="right">179 </td><td> </td></tr>
<tr><td valign="top"><img src="/icons/text.gif" alt="[TXT]"></td><td><a href="meta.conf">meta.conf</a></td><td align="right">2026-03-24 14:26  </td><td align="right">179 </td><td> </td></tr>
<tr><td valign="top"><img src="/icons/unknown.gif" alt="[   ]"></td><td><a href="packagesite.pkg">packagesite.pkg</a></td><td align="right">2026-03-24 14:26  </td><td align="right">325K</td><td> </td></tr>
<tr><td valign="top"><img src="/icons/unknown.gif" alt="[   ]"></td><td><a href="packagesite.tzst">packagesite.tzst</a></td><td align="right">2026-03-24 14:26  </td><td align="right">325K</td><td> </td></tr>
   <tr><th colspan="5"><hr></th></tr>
</table>
</body></html>
anton-w@ant1mbp4 2026-02-10 - ping OPNsense mirrors %
```

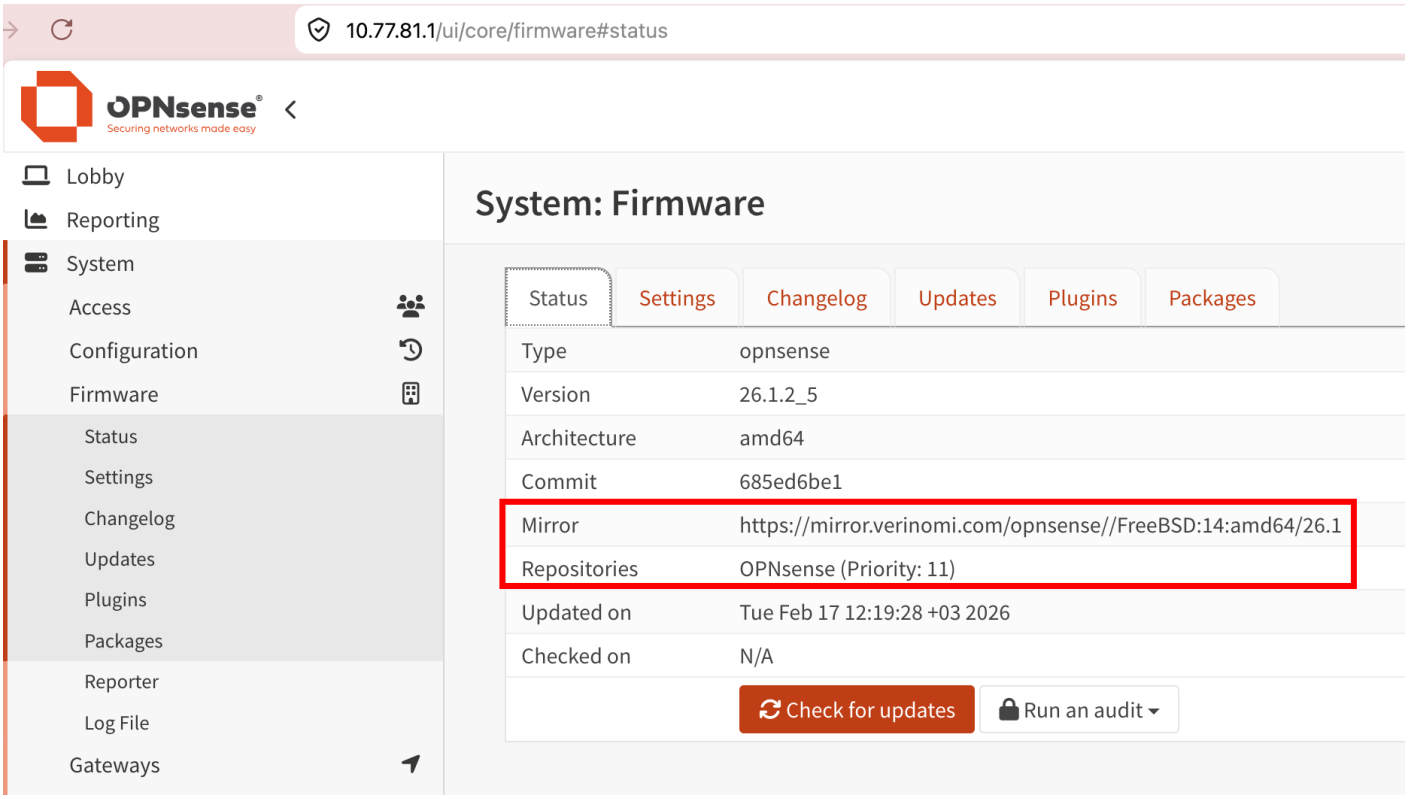
Some servers are located on CDN and protected, might be challenge to reach them or opposite (that depends on the region).



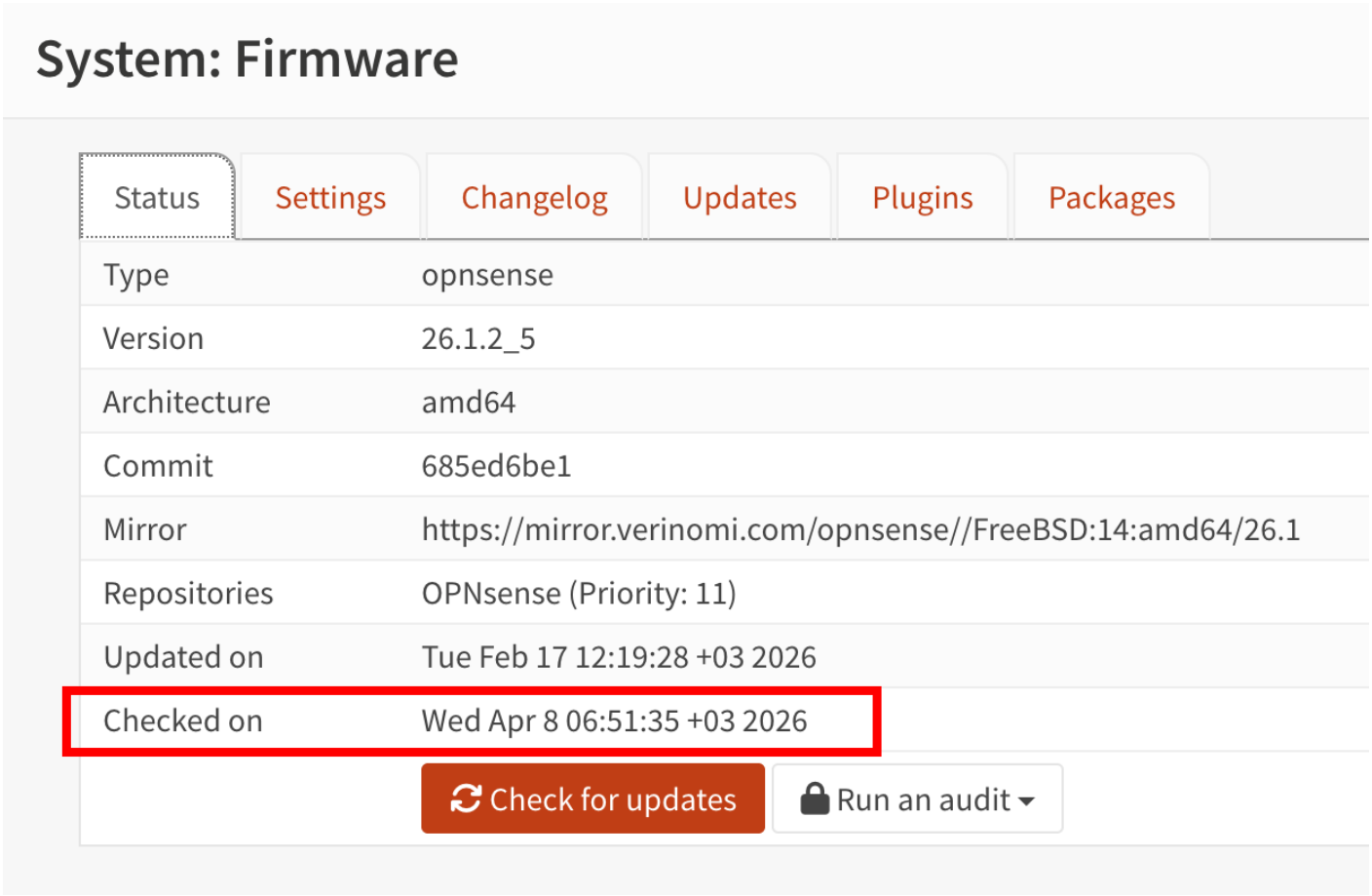
To configure selected server, navigate to System, Firmware, Settings, Mirror, choose (custom) from dropdown, type the server URL



Back to Status page will indicate that repository mirror is successfully configured



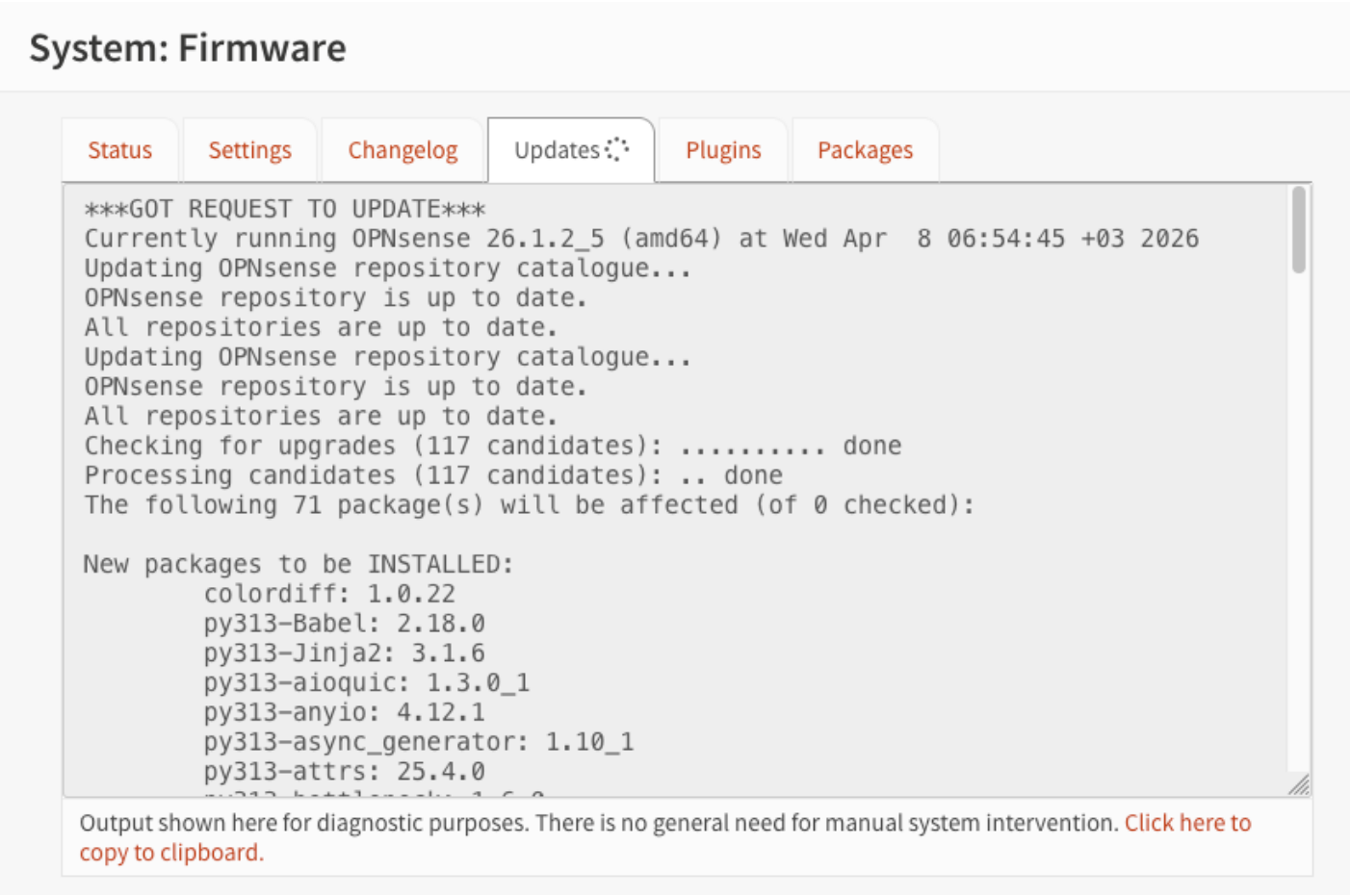
Should see dialog with fetching the packages and back to the Status, timestamp for latest update should be changed



Once you see newer base is available and packages will be upgraded, it is a good sign

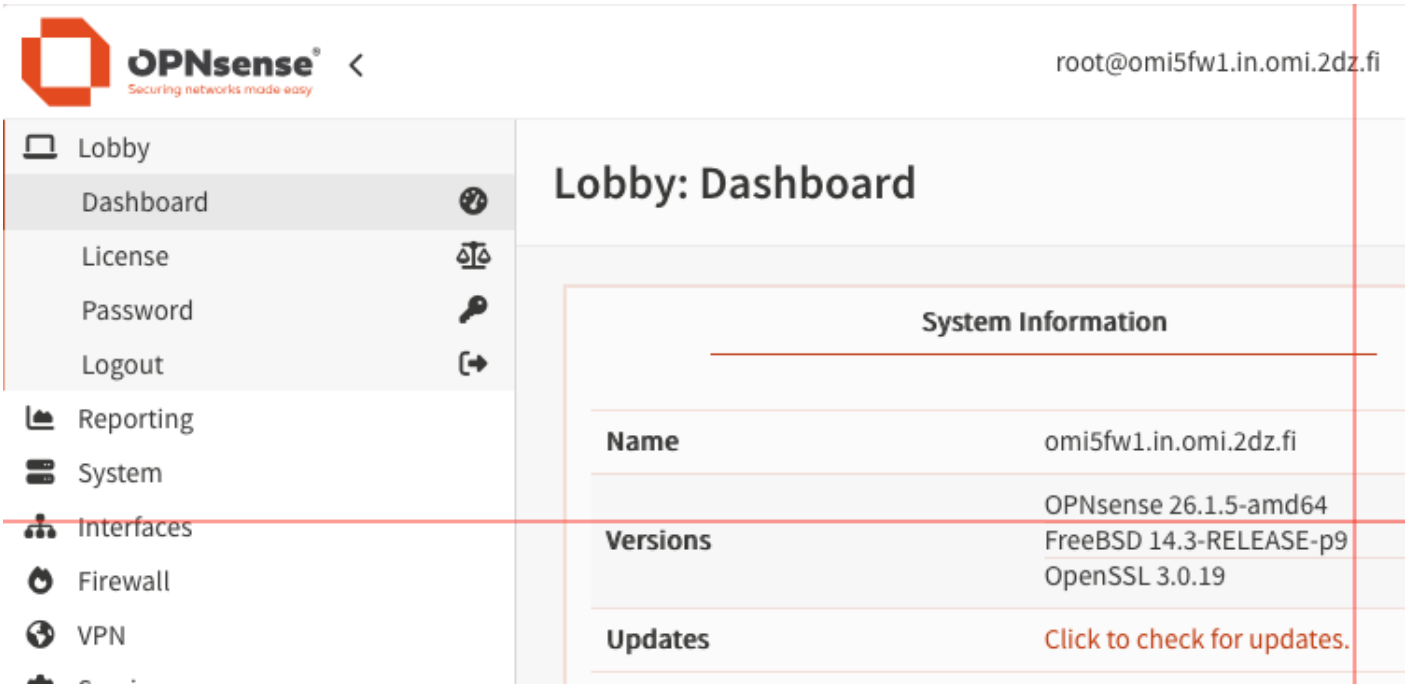

OPNsense
root@omi5fw1.in.omi.2dz.fi
Lobby
Reporting
System
Access
Configuration
Firmware
Status
Settings
Changelog
Updates
Plugins
Packages
Reporter
Log File
Gateways
High Availability
Routes
Settings
Snapshots
Trust
Log Files
Diagnostics
Interfaces
Firewall
VPN
Services
Power
Help
System: Firmware
Status
Settings
Changelog
Updates
Plugins
Packages
Package name
Current version
New version
Required action
Repository
base 26.1.1 26.1.3 upgrade OPNsense
colordiff N/A 1.0.22 new OPNsense
dnsmasq 2.92,1 2.92_2,1 upgrade OPNsense
glib 2.84.4,2 2.84.4,2 reinstall OPNsense
jansson 2.14.1 2.15.0 upgrade OPNsense
kea 3.0.2_1 3.0.2_2 upgrade OPNsense
kernel 26.1.1 26.1.3 upgrade OPNsense
libucl 0.9.3 0.9.4 upgrade OPNsense
libunistring 1.4.1 1.4.2 upgrade OPNsense
libxml2 2.15.1_1 2.15.2 upgrade OPNsense
nss 3.120 3.121 upgrade OPNsense
openldap26-client 2.6.12 2.6.13 upgrade OPNsense
opnsense 26.1.2_5 26.1.5 upgrade OPNsense
opnsense-installer 25.1_1 25.1_2 upgrade OPNsense
opnsense-lang 26.1.1 26.1.4 upgrade OPNsense
opnsense-update 26.1.1_1 26.1.3 upgrade OPNsense
perl5 5.42.0_1 5.42.1 upgrade OPNsense
php83-phpseclib 3.0.49 3.0.50 upgrade OPNsense
py313-aioquic N/A 1.3.0_1 new OPNsense
py313-anyio N/A 4.12.1 new OPNsense
py313-async_generator N/A 1.10_1 new OPNsense
py313-attrs N/A 25.4.0 new OPNsense
py313-Babel N/A 2.18.0 new OPNsense
py313-bottleneck N/A 1.6.0 new OPNsense
py313-certifi N/A 2026.1.4 new OPNsense
py313-cffi N/A 2.0.0 new OPNsense
py313-charset-normalizer N/A 3.4.4 new OPNsense
py313-cryptography N/A 46.0.5,1 new OPNsense
py313-dnspython N/A 2.8.0_1,1 new OPNsense
py313-duckdb N/A 1.5.0 new OPNsense
py313-h2 N/A 4.1.0_1 new OPNsense
py313-h11 N/A 0.16.0 new OPNsense
py313-hpack N/A 4.0.0_1 new OPNsense
py313-httpcore N/A 1.0.9 new OPNsense
py313-httpx N/A 0.28.1_1 new OPNsense
py313-hyperframe N/A 6.0.0_1 new OPNsense
py313-idna N/A 3.11 new OPNsense
py313-Jinja2 N/A 3.1.6 new OPNsense
py313-jq N/A 1.11.0 new OPNsense
py313-ldap3 N/A 2.9.1_1 new OPNsense
py313-markupsafe N/A 3.0.3 new OPNsense
py313-numexpr N/A 2.14.1 new OPNsense
py313-numpy N/A 1.26.4_12,1 new OPNsense
py313-outcome N/A 1.3.0_2 new OPNsense
py313-packaging N/A 26.0 new OPNsense
py313-pandas N/A 2.3.3,1 new OPNsense
py313-pyasn1 N/A 0.6.0 new OPNsense
py313-pyasn1-modules N/A 0.4.1 new OPNsense
py313-pycparser N/A 2.23 new OPNsense
py313-pylsqpack N/A 0.3.23 new OPNsense
py313-pyopenssl N/A 25.3.0_1,1 new OPNsense
py313-pysocks N/A 1.7.1_1 new OPNsense
py313-python-dateutil N/A 2.9.0 new OPNsense
py313-pytz N/A 2025.2_1,1 new OPNsense
py313-pyyaml N/A 6.0.3 new OPNsense
py313-requests N/A 2.32.5 new OPNsense
py313-service-identity N/A 24.2.0 new OPNsense
py313-six N/A 1.17.0 new OPNsense
py313-sniffio N/A 1.3.1 new OPNsense
py313-socksio N/A 1.0.0_1 new OPNsense

Upgrade itself

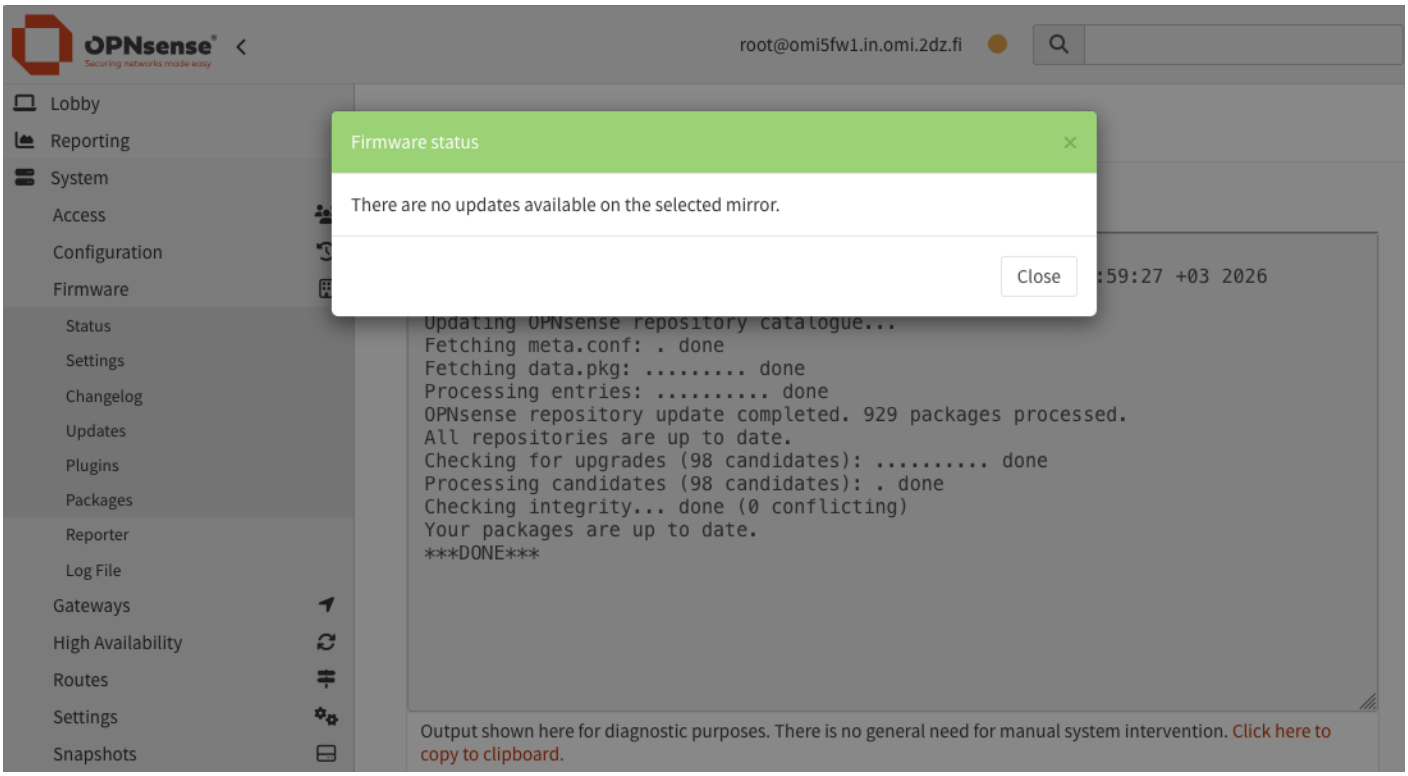


System config has an option to restart server, once upgraded, once that is enable, it will restart

After reboot.



Nothing to update. Good.



Revision #7
Created 8 April 2026 03:26:54 by Anton
Updated 13 April 2026 02:34:53 by Anton