WEBADM PUBLISHING PROXY

WebADM Publishing Proxy

Push

1. Product Overview

WAProxy is an HTTP(S) reverse proxy for WebADM. While any reverse proxy should be able to fill the role, this one has been already configured by RCDevs to work securely and use all the features WebADM provides to reverse proxies. WAProxy handles basic load balancing, failover, and both server and client certificates with the least possible amount of configuration effort.

Without a WAProxy reverse proxy, WebADM end-user web applications must be accessible from anywhere its users could be: if you use OpenOTP Push Login, a user's phone must be able to access the mobile communication endpoints on your WebADM installation from the internet. This forces you to place WebADM in the DMZ (far from your LDAP directory and your databases) or to maintain exceptions to let the DMZ's applications through to WebADM, in your internal network. None of those situations are easily manageable when trying to maintain proper security.

WAProxy helps by standing in a DMZ and offering only those services that should be accessible from outside, relaying requests to a WebADM installation in your internal network.

Unless your needs truly are unusual, we recommend you to use our product as a quick way to set up a properly working proxy and go from there.

With WAProxy enabled in WebADM, you can choose which end-user applications (WebApps) are published on the WAProxy.

2. System Requirements

The current version of WAProxy runs on Linux 32bit or 64bit operating systems with glibc >= 2.5. The installation package contains all the required dependencies allowing WebADM to run on any Linux-based system without any other requirement. You should use a dedicated machine (physical or virtual) for the WAProxy, since parts of the proxy run as root.

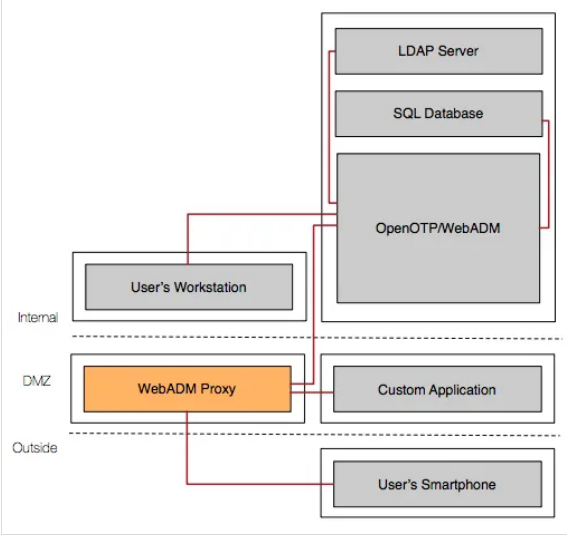
You are expected to have a running WebADM installation already (version >= 1.3), and it must be offering its services on the default 443 TCP port.

To run the WebADM Proxy, your dedicated system should fit the following requirements:

- Running a Linux distribution with glibc >= 2.5 installed (RedHat, CentOS, SUSE Debian, Ubuntu).
- At least a 1 GHz processor (two cores or vCPUs recommended). Architecture can be either x86 or x86-64.
- 512 MB of RAM.
- At the very least 50 MB of free disk space.

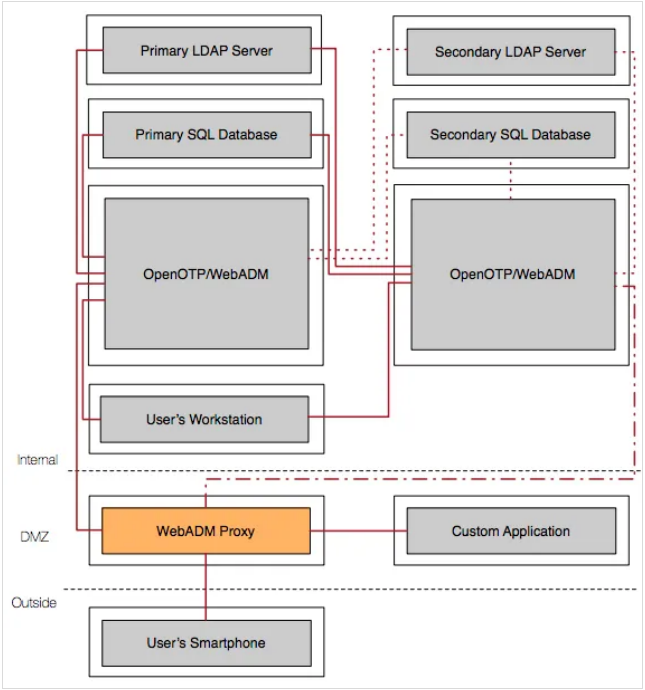
3. Deployment Scenarios

A simple case of deployment is the WAProxy in a DMZ relaying request to a single WebADM instance, such as our virtual appliance. The image on the next page illustrates such a case. White boxes represent machines, and the coloured boxes inside represent running processes. Red lines indicate communication between processes.



A more complicated case involves a WebADM cluster for the backend. To keep the high availability gained with a cluster, the proxy can send requests to the secondary server when the primary stops working. It can also distribute incoming requests over both members of the cluster, almost doubling the throughput of requests that can be treated, excluding pathological cases such as many concurrent modifications of the same user. The image on the next page illustrates such a setup. Dotted lines represent communication that happens only if a primary service has stopped working. The dot-and-dash line represents the possibility of the second server being either used only when the primary is down or during normal operation too.

> **Note**
>
> Note that instructions for setting up a cluster are in another document: the WebADM high availability guide.



4. Installation and Configuration

Installation can be performed using a repository or self-installer.

On a RedHat, CentOS or Fedora system, you can use our repository, which simplifies updates.

Add the repository:

```
yum install https://repos.rcdevs.com/redhat/base/rcdevs_release-1.1.1-1.noarch.rpm
```

Clean yum cache and install WAProxy:

```
yum clean all
yum install waproxy
```

On a Debian system, you can use our repository, which simplifies updates. Add the repository:

```
wget https://repos.rcdevs.com/debian/base/rcdevs-release_1.1.1-1_all.deb
apt-get install ./rcdevs-release_1.1.1-1_all.deb
```

Clean cache and install WAProxy:

```
apt-get update
apt-get install waproxy
```

Instead of using the repositories, you can also download the latest package on RCDevs Website. Select the package for the right architecture and execute it on your chosen host machine, as root.

After installation, you must run the WAProxy initial setup script located in /opt/waproxy/bin/setup. The setup will ask you the FQDN of the proxy: this should be the name that clients will use to connect to the proxy from the Internet. The setup script will create an HTTPs X.509 certificate based on this information. It will also ask for the IP or hostname of one of the back-end WebADM servers. The WebADM CA certificate file will be fetched from the back-end server and stored in `/opt/waproxy/conf/ca.crt`. The CA certificate will be used to authenticate users with client certificates for any WebADM application supporting this feature.

The WAProxy's configuration file is `/opt/waproxy/conf/waproxy.conf`. All options have reasonable defaults, except *server_addr1* and *server_addr2*, which must be set for the proxy to work at all.

These settings must also be the IP addresses to hostnames by which your WebADM hosts are known internally.

Note

Note that configuration changes need a restart of the WAProxy service (see next section).

Upgrades of the WAProxy will create (or overwrite) the file `/opt/waproxy/conf/waproxy.conf.default`, which you should check for any new configuration directives added by the upgrade.

> **Note**
>
> Note that WAProxy 1.1 is not compatible with configurations from WAProxy 1.0. You should uninstall WAProxy 1.0 before installing WAProxy 1.1.

4.1 Basic Configuration

Depending on what kind of WebADM backend you have, WAProxy has to be configured differently.

For a single WebADM server, you can just set server_addr1 in the configuration file. For two backend servers, you have to set *server_addr1* and *server_addr2*.

With two backend servers, you have to choose what scheme to use when relaying requests toward the backend. Setting *server_policy* to *Ordered* (the default value) tells the proxy to forward all requests to the first server and use the second only when the first stops answering properly. Setting *server_policy* to Balanced instead tells the proxy to relay the requests to both backend servers in a round-robin fashion.

4.2 Custom SSL Certificate

Refer to Trusted Certificate documentation to use your own SSL certificate with WAProxy.

4.3 SSL/TLS Ciphersuite

In default configuration different SSL/TLS version and ciphers are supported to maintain compatibility with older clients. You can enable/disable them further by using configuration settings in `/opt/waproxy/conf/waproxy.conf` (if this file doesn't exist in your environment, please create it). Please, have a look at the following documentation: Hardening your WebADM Server .

4.4 Network Configuration

Just like WebADM, WAProxy does not serve Web applications over unencrypted HTTP. Port 80 (or whatever you choose port_std to be, see later) will simply redirect the client to port 443 (or port_ssl).

You can change on what ports the proxy offers its services by setting the directives port_std and port_ssl in the configuration file.

The proxy will contact the WebADM backend over port 443.

> **Note**
>
> Note that WAProxy does not support WebADM installations that do not use the default ports.

4.5 Backend Configuration

IP of WAProxy servers

You should adapt the configuration of any WebADM server the proxy forwards to. On those servers, in `/opt/webadm/conf/webadm.conf`, you should add your WAProxy's IP addresses separated by a comma to the waproxy_proxies directive:

```
waproxy_proxies "<YOUR_WAPROXY_IP1>", "<YOUR_WAPROXY_IP2>"
```

If you have at least one or more reverse-proxy between your WAProxy server and clients, the waproxy_proxies directive should be configured with number of reverse-proxies (including WAProxy server) that are between the WebADM server and the clients, so WebADM is still able to get the actual client IP address. In that case, the directive must be configured like this:

```
waproxy_proxies "<YOUR_WAPROXY_IP> <NUMBER_REVERSE_PROXIES>", "<YOUR_WAPROXY_IP2>
<NUMBER_REVERSE_PROXIES>"
```

Push feature

If you use the OpenOTP Mobile Push applications on WebADM, you should configure them to use the proxy's OpenOTP endpoints, and not WebADM's default. (Look for "Mobile EndPoint URL" in the applications' configuration page.) The WAProxy endpoints should be `https://<public-proxy-name>/ws/openotp/`.

FIDO2 configuration

Similarly, if you use an OpenOTP login mode that includes FIDO2, please check that your public WAProxy matches `FIDO Devices->FIDO Origin or AppId` configuration setting of OpenOTP.

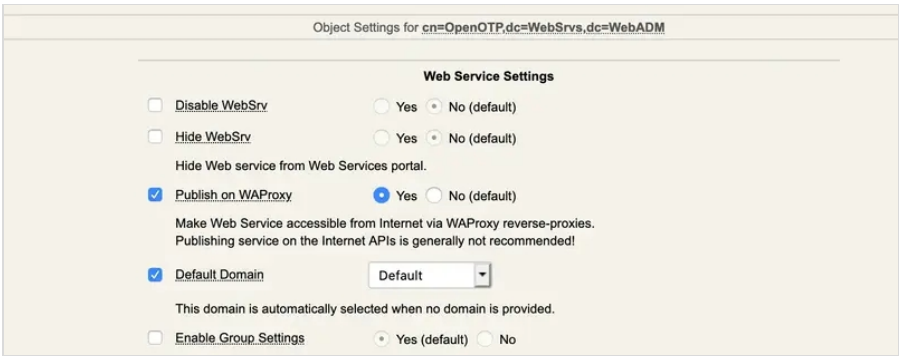
4.6 Publishing web services

By default, WAProxy does not proxy the web services provided by WebADM on port 8443. This can be enabled with the following setting in /opt/waproxy/conf/waproxy.conf file:

```
publish_websrvs Yes
```

Accessing the web services through the WAProxy requires the following:

› publish OpenOTP to WAProxy, in settings of "MFA Authentication Server (OpenOTP)":



› the use of a client certificate issued by WebADM when accessing the web service URL.

4.7 Auto-renewal of TLS certificate

If you want to enable auto-renewal of TLS certificate used by WAProxy server, you can add a cron entry running /opt/waproxy/bin/waproxy renew command. This command will renew certificate if it is near expiration, and reload rcdevs-waproxy processes so new certificate is used.

4.8 As a Last Resort

If you have the necessary knowledge of Apache HTTPd, you can change `/opt/waproxy/lib/httpd.ini` to suit your needs. This is not supported, however, and upgrades of the proxy will overwrite this file. We consider even the choice of Apache's HTTP server an implementation detail that can change over releases. If you have found yourself in a situation needing such a change, please contact us, so we can try to incorporate it to the official release.

5. Maintenance and Troubleshooting

This section should cover your common administrative tasks concerning WAProxy. If not, you can contact our commercial support or our free support on RCDevs Google Group.

5.1 Starting and Stopping

If you let the installer put the WAProxy init scripts on your machine, the proxy service should start at machine boot. You should also be able to start and stop the proxy through your distribution's usual commands, such as "systemctl start waproxy" for distributions using systemd like Red Hat Enterprise Linux 7. Alternatively, you can use `/opt/waproxy/bin/waproxy start|restart|stop`.

5.2 Upgrading or Removing

When a new version of WAProxy is released, you can download and install it as you did for your first installation. The installer will offer you the option of upgrading your installation. Be aware that, to do so, the installer will stop WAProxy. As a matter of principle, you should back up the `/opt/waproxy` directory before the upgrade. You can then restore the directory if anything breaks, restart the WAProxy service and contact us about what happened.

The installer also gives you the option of removing an existing WAProxy installation.

You can “reset” your installation by executing `/opt/waproxy/setup reset`, which removes any init, rc and log rotate scripts the installer put on the machine. This will also remove the proxy’s logs, key and certificates.

5.3 Troubles and Dealing with them

Connections that “stall” until they time out are hints that a firewall somewhere may be dropping packets (perhaps rightly so if you haven’t put the right hostnames in the configuration). Do not forget the host’s own firewall.

WAProxy was not designed to work with SELinux. If your host has SELinux enabled, you should set its mode to “Permissive”. On Red Hat 7, you can execute `setenforce Permissive`, and set “SELINUX” to “permissive” in `/etc/selinux/config` to make the change permanent.

If you have not set up a proper PKI, you might see errors in the logs such as “AH00898: Error during SSL Handshake with remote server” when trying to access a resource through the proxy. These errors can appear when the WebADM servers do not present certificates that are consistent with their names.

You can find a trace of every error happening in `/opt/waproxy/logs/waproxy.log`. Those logs are rotated if you let the installer install its log rotate script on the machine (assuming log rotate is run regularly on your host).