

Uplivra is in beta and online purchasing is paused. [Apply to test it](#): testers get a license on request and 10% off when purchasing opens.

INSTALL GUIDE · ACCESS CONTROL

Access Control (TACACS+ and RADIUS)

Point your network devices at Uplivra for admin logins and port access, write rules for who may do what, and keep a record auditors accept.

Uplivra Technologies LLC · Guide for Uplivra 26.10 · Updated 2 October 2026 · Latest version:
<https://uplivra.com/guides/access-control-aaa.html>

How it works

When someone logs in to a switch, or a laptop or phone connects to a port or Wi-Fi, the network device asks Access Control: "may this person or device in, and with what rights?"

1. **The question arrives** over TACACS+ (TCP 49, for admin logins and commands) or RADIUS (UDP 1812, for ports and Wi-Fi; RadSec is RADIUS inside TLS). The device proves it's one of yours with its own shared key.
2. **Who is it?** A person is checked against Access Control's own people, or your company directory (Active Directory or LDAP). A device is known by its MAC address (MAB) or its certificate (802.1X).
3. **The rules decide**, top to bottom, first match wins: allow or deny, the privilege level, the VLAN, the command set. Every decision says which rule made it and why.
4. **Every request is logged** with the answer, the rule and how long it took. Accounting from the devices (what was run, sessions that started and ended) is logged too.

Where the answering happens:

- **On the main Uplivra server** (the default, fine for one site).
- **On Access Control nodes** (below): small machines at your sites that answer from a signed copy of the rules, and keep answering when the main server or the link to it is down.

Either way, the main server is where you change rules, and it holds the complete log.

Before you start

Access Control is a module: \$5 per network device a month. Open it under **Network > Access Control**. The overview has a checklist; this guide follows it.

Two things to plan:

- **Reaching Uplivra.** Network devices reach the Uplivra server on TCP 49 (TACACS+) and UDP 1812 and 1813 (RADIUS). Allow those in the server's firewall from your management networks, for example `sudo ufw allow from 10.0.0.0/8 to any port 49 proto tcp`.
- **A way back in.** Every device keeps a local admin account for when Uplivra can't be reached. The generated commands keep one, and keep the console on local logins.

1. Network devices

Under **Network devices**, tick monitored switches, routers and firewalls, or add an address or a whole network (10.20.0.0/24). Each device gets its own random 32-character key, stored encrypted.

Reply format tells Uplivra how to tell that device someone's rights: Cisco and Arista use privilege levels; Juniper, Fortinet, Palo Alto and MikroTik use role or profile names.

RADIUS requires **Message-Authenticator** by default, to protect against the Blast-RADIUS attack (CVE-2024-3596). Turn the requirement off only for a device too old to send it, and plan to upgrade it.

2. People

There are two ways in:

- **Local people:** a user name and password typed at the device. Passwords are stored as salted PBKDF2 hashes. Use **Account ends** for contractors.
- **Directory groups:** connect an Active Directory or LDAP group to an Access Control group. Members log in with their directory password, and their access changes as soon as their group membership does.

3. Rules and command sets

Start with suggested rules adds:

- Network admins: everything, with risky commands needing approval;
- Help desk: show commands and port resets on switches;
- Auditors: read-only;
- Unknown devices on ports: monitor mode.

Change the group names to yours.

How rules work:

- Rules are checked from the top. The first one that fits decides, and nothing fits means no access. Approved temporary access is checked before every rule.
- A rule can match groups, user names, device types, tags, sites, particular devices, days and hours, maintenance windows, or "only with approved temporary access".
- A rule gives a privilege level or device role, a command set, a session time limit, and for port access a VLAN or named ACL.

Command sets are lines such as `allow show`, `deny /^debug/` or `approve reload`, checked from the top. With **approve**:

1. The device refuses the command and asks under **Temporary access**.
2. A second person approves it.
3. The person runs it again within the approval time (15 minutes by default). It works once.

Use **Try a rule** with a user, device, time and command: it shows the decision, which rule made it, every rule it skipped and why, and what would change if another rule came first.

4. Turn on the service

On the overview, tick **Answer TACACS+ and RADIUS** and save. The service starts within 30 seconds.

5. Configure a device

On a network device, click **Device commands** and pick the maker and what to set up:

- admin logins through TACACS+;
- admin logins through RADIUS;
- port access with MAB and 802.1X. Start in monitor mode where the device has one, and try one port first.

Uplivra writes the steps: a check, the commands, a test from a new session, saving, and the way back. Tick **Put the shared key in the commands** to fill it in; that's recorded. Juniper uses `commit confirmed 10`, so a mistake undoes itself.

Keep your session open until a new login through Uplivra works.

Temporary admin access

Anyone with Access Control view rights can ask for higher rights for a user on chosen devices or sites, for 30 minutes to 8 hours, with a reason and a ticket number. Someone else with **Approve temporary admin access** approves it. The clock starts then, and the access ends by itself. **End now** stops it early.

The log, dashboard and findings

- **Log:** every login, command, accounting record and refused sender, kept 400 days by default (at least 90). It can be searched and exported to CSV.
- **Overview:** success rate, response times, and failures by reason, person and device.
- **Findings:** looks at the last day against the last month:
 - failures followed by an admin login (high risk);
 - an admin login from a new address (suspicious);
 - logins from two sites at once (suspicious);
 - a first login to a device (informational);
 - unusual hours (informational);
 - bursts of refused commands (suspicious).

High-risk findings raise an alert. Findings never block anyone by themselves.

Recorded sessions

Admin sessions opened through Uplivra's browser terminal are recorded command by command against the reason given.

- **Flag** (the default) alerts when someone runs a command their rules don't allow.
- **Block** stops the command before it reaches the device.
- Anyone with **Change Access Control** can end a live session.

Only sessions through Uplivra can be watched or ended. Direct SSH to a device is covered by TACACS+ command authorization and accounting.

802.1X with certificates (EAP-TLS)

Laptops and phones can sign in to the network with a certificate, with no password to type or steal.

1. Under **Access Control > Overview > 802.1X with certificates**, paste the certificate of the CA that issues your devices' certificates. Microsoft AD CS and Intune both work, and you can paste several CAs.
2. Make sure devices trust Uplivra's web certificate. With Let's Encrypt they already do.
3. Add a rule with the method **dot1x** and the VLAN to use. The person or device is the certificate's email address, or else its common name.
4. Its groups come from **People** when you've added it there. Otherwise they come from the certificate's organizational units (OU).

To block a certificate, add that name under **People** and turn it off. Uplivra gives the switch or access point the session keys (MS-MPPE), so Wi-Fi encryption works. EAP-TLS runs over TLS 1.2.

RadSec

RadSec is RADIUS inside TLS, on TCP 2083. Cloud-managed access points use it, and it's the safe way to send RADIUS across the internet or a WAN.

1. Set the RadSec port.
2. Paste the CA that signs the devices' RadSec certificates.
3. List each device under **Network devices** by its address.

The shared secret is the fixed `radsec`, because TLS does the protecting.

Port sessions and change of authorization

Port sessions lists who and what is on switch ports and Wi-Fi now, from the RADIUS accounting the devices send. For each session you can:

- **disconnect** it;
- **reauthenticate** it, to check it again with today's rules;
- **bounce** the port (Cisco), so the device asks for a new address.

This is RFC 5176. The device must accept change of authorization from this server on UDP 3799; the Cisco 802.1X commands Uplivra makes include it. Every request is recorded.

Moving from ISE, ClearPass or NPS

Export the network device list from your old server and import it under **Network devices > Move from another RADIUS or TACACS+ server**.

- Each device keeps its shared secret, so on the device you only change the server address.
- Move a few devices at a time, using each one's fallback server.
- Message-Authenticator isn't required for imported devices at first. Turn it on for each device once it sends it.

Nodes and redundancy

A **node** is a machine (Linux or Windows, physical or virtual) that answers your network devices for Access Control. Use nodes when:

- logins must keep working if the main Uplivra server, or the link to it, is down;
- you have several sites and want each site's devices to ask something close by;
- you want two servers answering, so one can be patched or fail without anyone noticing.

How nodes work

- **The main server controls the rules.** It sends each node a **policy bundle**: the network devices and their keys, people, directory settings, groups, rules, command sets, temporary access and maintenance windows that node needs. Only the company's own data goes to its nodes.
- **Bundles are signed.** The main server signs every bundle with its own Ed25519 key. A node checks the signature before it uses a bundle and never goes back to an older one, so nobody in between can feed it different rules.
- **Nodes do the work.** Each node answers TACACS+ and RADIUS from its bundle by itself, the same way the main server would. It doesn't ask the main server per request, so it keeps working when the server can't be reached.
- **Each node has its own encrypted storage.** Its bundle and its log are kept encrypted (AES-256-GCM) with a key only that machine can open: DPAPI on Windows, a file only the service can read on Linux.
- **Logs come back to the main server.** A node checks in every 10 seconds over HTTPS, hands over its log, and picks up new rules. If the main server can't be reached, it keeps the log (up to

500,000 records) and sends it when it can. Each record carries the node and a sequence number, so the main server stores it once, however many ways it arrives.

- **Risky commands that need approval** are asked of the main server, where someone approves them. If it can't be reached, the command is refused (use the device's local break-glass account).

Pairs

Two nodes can be a **pair**. A pair shares everything:

- every log record either node writes, so each holds a copy of the other's;
- the newest rules, so a node that can't reach the main server gets them from its pair;
- what the main server has already stored, so both can tidy up;
- health: each knows if the other is up.

If one node can't reach the main server, its pair forwards its log and hands it new rules. Point devices at both nodes of a pair; the device commands do this for you.

The link between a pair is **mutual TLS 1.3**: each node shows a certificate from the main server's **Access Control CA** with the other node's name in it, so nothing else can join in. Two ways to carry it:

- **HTTPS** (default): for nodes at different sites. It works through firewalls that allow HTTPS.
- **Direct**: for nodes on the same network. One long-lived encrypted connection, with less overhead.

Open the pair port (TCP 8447 unless you change it) between the two nodes.

Add a node

1. On the main server: **Access Control > Nodes > Add a node**. Give it a name, its address (as devices reach it) and its site. Uplivra shows a one-time join code and the exact command. The code works once, for 24 hours.
2. On the node machine, install the Uplivra program, then run the command it showed, for example:

```
sudo uplivra aaa-node enroll -server https://uplivra.example.com -code upj_... -pin  
'sha256//...'  
sudo uplivra aaa-node install
```

On Windows use `uplivra.exe` in PowerShell as administrator. The pin makes sure the node is talking to your server. The node makes its own private key; the main server signs its certificate.

3. Within a few seconds the node shows as **online** with **current** rules.
4. Pair it with another node under **Change ... > Pair with**, and pick **HTTPS** or **Direct**.
5. Open TCP 49 and UDP 1812-1813 to the node from your network devices, and the pair port between the two nodes.
6. Update your devices: **Network devices > Device commands** now lists the site's nodes first (server 1 and server 2).
7. When every device points at nodes, tick **Only nodes answer devices** on the Nodes page. The main server then only keeps the rules and the log.

Check a node from its own machine with `uplivra aaa-node status`.

Certificates for 802.1X and RadSec: a node uses its own certificate from the Access Control CA (download the CA from the Nodes page for your laptops and phones to trust), or the main server's certificate if your devices already trust that.

Change of authorization: disconnect and reauthenticate requests still go from the main server, so devices must accept them from the main server's address.

Health and alerts

The Nodes page shows each node's status, requests a second now and at peak, average answer time, its pair link, whether it runs the newest rules, and how much log is waiting. An alert opens when a node stops checking in, can't reach its pair, stays on older rules, or its certificate is about to end.

Sizing

How many requests does one server answer? It depends on the kind of request.

Measured on Uplivra's own Access Control code, on a small virtual machine with 2 CPU cores (the test clients ran on the same machine, so a real server does better):

Request	Per second
TACACS+ command authorization	about 21,000

Request	Per second
TACACS+ accounting	about 20,000
TACACS+ login, password checked in the last 10 minutes	about 9,600
RADIUS MAB (phones, printers, cameras)	about 2,900
802.1X with certificates (EAP-TLS), full handshake	about 400
Login with a password not checked recently	about 18

Notes:

- **Passwords are stored slowly on purpose** (PBKDF2, 600,000 rounds), so a stolen copy can't be guessed quickly. Checking one costs about a fifth of a second of one CPU core. A password that checked out is remembered for 10 minutes (in memory only), so repeat logins are fast. People signing in with their directory password are checked by the directory.
- **802.1X costs the most:** about eight round trips with real cryptography per connection.
- **For comparison:** Cisco publishes about 150-200 EAP-TLS and 900-1,300 PAP authentications a second per ISE policy node, and 2,500-3,200 TACACS+ transactions a second on a dedicated node. Radiator reports over 4,000 EAP-TLS a second on cloud machines.

What that means for a small or mid-size business:

- A busy morning with 500 people arriving in 15 minutes is under 1 request a second on average.
- Everyone reconnecting at once after a switch restart, say 300 phones and laptops, finishes in a few seconds on one node.
- One node is plenty for speed. **The second node is for staying up**, not for speed.
- Give each node 2 CPU cores and 2 GB of memory. Add cores for heavy 802.1X (thousands of devices reconnecting at once).

What's not here yet

SCEP certificate enrollment, Entra ID and Okta lookups, EAP-TLS over TLS 1.3, and sending change of authorization from nodes are planned. Use your existing CA for certificates.