SVDSA uptime monitor: setup

For the SVDSA tech committee.

This sets up the chapter's uptime monitor on the chapter's GitLab and Cloudflare accounts. It checks the chapter's sites every 5 minutes, shows the results at `status.siliconvalleydsa.org`, and can send push notifications. Do the steps in order.

Before you start: Owner on the `svdsa` GitLab group, a login on the official Cloudflare account that can create API tokens, Git plus Bun or Node on your computer, and about an hour.

`uptime-config` is public on purpose. It holds only domain names, redirect targets and Cloudflare resource IDs, which do nothing without an API token. Helena's real address stays out of it and goes in a Worker secret (step 17).

GITLAB

1. Import `uptime-config`

1. Open New project → **Import project** → **Repository by URL**.
2. **Git repository URL:** `https://github.com/cinderblock/svdsa-uptime-config.git`
3. **Project URL:** group `svdsa`, slug `uptime-config`. **Visibility:** Public.
4. Select **Create project**.

It's a one-time copy and is the chapter's from then on. Nothing deploys yet.

> **DONE WHEN**
>
> ☐ `svdsa/uptime-config` shows files

2. Fill in `uptime-tracker`

The empty `svdsa/uptime-tracker` project already exists, so push into it:

```
git clone https://github.com/cinderblock/uptime-tracker.git
cd uptime-tracker
git push git@gitlab.com:svdsa/uptime-tracker.git master
```

Then **Settings → General → Visibility → Public → Save changes**.

DONE WHEN

- [ ] Both projects show their files in a logged-out browser window

3. Optional: give Cameron access to the engine

If you want Cameron to push engine fixes to the chapter's copy, add `@cinderblock` to `uptime-tracker` members as **Maintainer**. Pushing there deploys nothing; `uptime-config` pins an exact engine commit. Skip it and you pull engine updates yourselves when you want them.

DONE WHEN

- [ ] Done, or skipped

CLOUDFLARE

Log in to the dashboard and select the official SVDSA account.

4. Check the Workers plan

Look under **Workers & Pages → Plans**. The monitor works on either.

	Workers Free	**Workers Paid ($5/month)**
If someone floods the page	Every Worker on the account stops until 5 PM PDT / 4 PM PST (00:00 UTC)	Workers keep running; usage is billed
Included	100k requests a day	10M requests a month, then $0.30 per million
Spending cap	Not needed	None available

A flood of 1,000 requests a second for a full day would cost about $24 on Paid. Requests blocked by the rule in step 20 aren't billed.

On Paid, add **Notifications → Add → Budget Alert**. It only emails.

DONE WHEN

- [ ] Plan: Free / Paid
- [ ] Budget alert added (Paid only)

5. Copy the account ID

Press **Ctrl+K**, type `Copy account ID`, and select it. You'll paste it in step 10.

> **DONE WHEN**
> - [ ] Copied

6. Check the `workers.dev` subdomain

Under **Workers & Pages**, look for **Your subdomain**. If none is set, pick one the chapter is happy to keep. A new one can show error 1042 for a few minutes.

> **DONE WHEN**
> - [ ] Subdomain set

7. Create the database

Storage & databases → D1 SQL database → Create Database. Name it `svdsa-uptime`, keep the default location, and copy its **Database ID**. Don't create tables; the deploy does that.

> **DONE WHEN**
> - [ ] Database ID copied

8. Create the KV store

Storage & databases → Workers KV → Create instance. Name it `svdsa-uptime` and copy its **ID**.

> **DONE WHEN**
> - [ ] KV ID copied

9. Create the API token

1. **Manage Account → Account API Tokens → Create Token → Create Custom Token.** An account token keeps working if its creator leaves.
2. **Name:** `gitlab uptime-config deploy`
3. **Permissions**, Account level, Edit: **Workers Scripts**, **D1**, **Workers KV Storage**.
4. **Account Resources:** the official SVDSA account only.
5. **Continue to summary → Create Token**, and copy it. It's shown once.

Don't use the "Edit Cloudflare Workers" template; it adds zone rights the deploy doesn't need.

DONE WHEN

- [ ] Token copied

10. Add the GitLab CI variables

In uptime-config CI/CD settings → **Variables**, add:

Key	Value	Options
`CLOUDFLARE_API_TOKEN`	the token from step 9	Masked and hidden, Protected
`CLOUDFLARE_ACCOUNT_ID`	the ID from step 5	Protected

Protected variables only reach protected branches, and `master` is protected by default.

DONE WHEN

- [ ] Both added

FIRST DEPLOY

11. Generate the push keys

```
bunx web-push generate-vapid-keys
```

(`npx` works too.) Keep the **Public Key** for step 12. Save the **Private Key** in VaultWarden as `VAPID_PRIVATE_KEY`.

DONE WHEN

- [ ] Keys generated

12. Fill in `wrangler.toml` and commit

In `svdsa/uptime-config`, open `wrangler.toml` with **Edit → Web IDE** and fill in the three blanks:

```
VAPID_PUBLIC_KEY = ""   # Public Key from step 11

[[d1_databases]]
database_id = ""        # Database ID from step 7

[[kv_namespaces]]
id = ""                 # KV ID from step 8
```

Commit to `master`. That starts the first deploy.

DONE WHEN

- [ ] Committed

13. Watch the pipeline

Open **Build → Pipelines** in `uptime-config`. If it stops for **identity verification**, follow GitLab's prompt and retry the job.

DONE WHEN

- [ ] Pipeline passed

14. Check the status page

Open the `workers.dev` address shown under **Workers & Pages → svdsa-uptime**. Results appear within 5 minutes. There's no admin login yet.

DONE WHEN

- [ ] Status page loads with results

SECRETS

Each secret goes in **Workers & Pages → svdsa-uptime → Settings → Variables and Secrets → Add**, **Type: Secret**, then **Deploy**. Later deploys keep them.

15. Add the login and push key

Add all three in one go; the Worker won't start with only one of the first two.

Name	Value
`DASHBOARD_PASSWORD`	An admin passphrase. Save it in VaultWarden
`AUTH_SECRET`	Any long random string, such as from `openssl rand -base64 48`
`VAPID_PRIVATE_KEY`	The Private Key from step 11

DONE WHEN

- [ ] Added and deployed

16. Check login and notifications

Log in through **Admin** on the status page. Then click a 🔔 bell on your phone or laptop, allow notifications, and check that the device shows on the admin **Subscriptions** page.

DONE WHEN

- [ ] Login works
- [ ] Device listed

17. Add the Helena direct check

This checks Helena at its real address, bypassing Cloudflare's proxy. The page shows only the display name; nobody sees the address, admins included.

See what Helena answers on plain HTTP:

```
curl -sI -H "Host: siliconvalleydsa.org" http://HELENA_IP/
```

Write this, replacing the address and choosing the last field from the table:

```
[
  {
    "id": "helena",
    "name": "Helena (origin, direct)",
    "panel": "Silicon Valley DSA",
    "address": "HELENA_IP",
    "kind": "http",
    "host": "siliconvalleydsa.org",
    "redirect": "https://siliconvalleydsa.org/"
  }
]
```

`curl` shows	Last field
A 301 or 302 to `https://…`	`"redirect": "https://siliconvalleydsa.org/"`
`HTTP/1.1 200`	`"expect": 200`
No answer	Replace `kind`, `host` and the last field with `"kind": "tcp", "port": 443`

Add it as the secret `SECRET_SERVICES`, and save a copy in VaultWarden.

> **Limits:** The check can't verify Helena's HTTPS certificate, only that it answers. If Helena's firewall accepts only Cloudflare's IP ranges, the check shows down; use the `tcp` kind or tell Cameron.

DONE WHEN

- [ ] Helena shows with a **DIRECT** label

GO LIVE

18. Connect the status domain

1. In `siliconvalleydsa.org` → **DNS → Records**, search for `status`. Remove or rename any existing record first.
2. **Workers & Pages → svdsa-uptime → Settings → Domains & Routes → Add → Custom Domain.**

3. Enter `status.siliconvalleydsa.org` → **Add Custom Domain**. It takes a few minutes.

DONE WHEN

- [ ] status.siliconvalleydsa.org loads

19. Turn off the `workers.dev` address

Now that the custom domain works, close the old address, so there's one public address and step 20 covers it. In `wrangler.toml` on `uptime-config`, add this near the top and commit:

```
workers_dev = false
```

DONE WHEN

- [ ] Pipeline passed, and the `workers.dev` address no longer loads

20. Suggested: add the flood rule

The monitor already limits each visitor, but those requests still reach the Worker. This rule stops a flood before it gets there, which matters most on Workers Paid, where requests are billed. On the Free zone plan a rule can't target a hostname, so it also covers the main site's homepage; 20 loads in 10 seconds is far beyond normal browsing. If `siliconvalleydsa.org` already has a rate-limiting rule, skip this step, since Free allows only one.

1. `siliconvalleydsa.org` → **Security → Security rules → Create rule → Rate limiting rules.**
2. **Name:** `Status page flood guard`
3. **Edit expression**, paste:

   ```
   (http.request.uri.path eq "/") or (http.request.uri.path eq "/api/poll")
   ```

4. **Rate:** 20 requests per 10 seconds, per IP. **Action:** Block for 10 seconds.
5. **Deploy.**

DONE WHEN

- [ ] Deployed, and the status page and main site still load (or skipped)

AFTERWARDS

To	Do
Add or change a monitored site	Edit `src/config.ts` in `uptime-config` and commit
Change the Helena check	Update the `SECRET_SERVICES` secret
Replace the API token	Repeat step 9, update the GitLab variable, delete the old token
Update the engine	Move `uptime-config`'s `vendor/uptime-tracker` submodule to a newer commit and commit

If a pipeline fails, the site keeps running the last good version.