From Web Accelerator to WAF

scalabilty **and** security

Marco d'Itri
<md@seeweb.it>
@md@linux.it

Seeweb s.r.l.

MSP Day - 7 giugno 2024

Who is Marco

Marco d'Itri

- Involved in Internet things in Italy since the mid '90s (we may have met on Usenet...).
- A Debian Developer for 27 years (mutt, inn, ppp, netbase, hotplug, udev, systemd...).
- I also wrote the `whois` command used by all Linux distributions.
- Employed by Seeweb, an italian cloud infrastructure, hosting and colocation provider.
- Designed and manages the Seeweb network (and other services).
- Designed and manages the Seeweb SOC.

seeweb

What is Seeweb

A cloud services provider in Italy

- 1998: founded as a pure hosting provider, after an experience as an ISP.
- 2005: opens a second data center in the Via Caldera Campus in Milano.
- 2010: first in Italy to provide cloud infrastructure.
- 2015: creates DHH S.p.A., a company listed on the Milano stock exchange which invests in cloud computing companies in the emerging markets of Europe.

Seeweb owns 4 data centers in Frosinone and Milano.

DHH is also present in Switzerland, Slovenia, Croatia, Serbia and Bulgaria.

seeweb

Content

seeweb

What is a caching reverse proxy? What is a WAF?

Reverse proxy

A specialized HTTP server which actually gets the data from *another* HTTP server.

Add caching...

A caching proxy is able to remember the data received from the backend and return it much more quickly.

Add ACLs and content inspection...

A Web Application Firewall is a reverse proxy which can block requests or replies depending on their content.

seeweb

Why is caching needed?

Caching is **essential** for non-trivial web sites!

When pages are served by PHP then it's too late!

There are orders of magnitude of difference between the latencies of a response from the cache or from the backend.

Caching is needed for scalability and also to mitigate some DoS attacks.

seeweb

The Web Accelerator

The Seeweb Web Accelerator is an HTTP reverse proxy delivered on the Seeweb public cloud infrastructures as a fully managed single-tenant appliance.

Product history

- 2012: initial offering to customers.
- 2016: integrated nginx and Let's Encrypt in the configuration system.
- 2021: upgraded to Varnish 6.
- 2021: implemented an optional WAF.

Development has been driven mostly by customers' needs.

seeweb

The technology

Caching

Varnish

TLS termination

nginx

WAF

ModSecurity with custom rules.

Configuration and deployment system

YAML, Perl, Template::Toolkit, Moose and Moo.

Observability

Munin, Prometheus.

Configuration and deployment system

The appliances are installed by an automated procedure. Their configurations are managed in a git repository and remotely deployed with a single command.

The deployment system manages the configurations of Varnish, nginx, the Let's Encrypt ACME client and the WAF from a single YAML file.
Normal operations do not require accessing the appliance with SSH.

Appliances are stateless and not backed up by design: if needed, in a few minutes they can be automatically reinstalled and their configuration deployed.

seeweb

Modular configuration

It allows to manage most customers with a simple declarative configuration and automatically benefit from updates to the standard modules.

Abstracted configuration

- General template.
- CMS-specific modules (e.g. Wordpress, Magento...).
- Function-specific modules (e.g. mobile detection, WebP negotiation...).

With optional customer-specific modules

- For custom VCL (e.g. complex routing, manipulating headers).

seeweb

An example configuration

Each customer is configured with a simple YAML file:

```yaml
backends:
  - host: 192.0.2.10

frontends: vm4635.cloud.seeweb.it

version: 6

modules: [ selective_caching, wordpress ]

ssl:
  sites:
    - vhosts: blog.seeweb.it
      cert_file: WILDCARD.seeweb.it

variables:
  hsts_enable: 1
```

Content

seeweb

Caching: Varnish

Varnish

An HTTP reverse proxy with advanced features for caching responses.

Varnish Configuration Language (VCL)

All policies are defined using an imperative domain-specific language.

VCL allows creating highly detailed custom policies.

Added value: customers benefit from standardized and well tested configurations which implement the best practices for caching.

seeweb

Caching strategies

Default: the standard semantics of HTTP

- Responses cannot be cached if they receive or set a cookie. But everything does!

"Standard" caching

- Ignore cookies for objects which are presumed static: images, Javascript, CSS, etc...

"Forced" caching

- Force caching for everything, with exceptions like the administration area or the user preferences pages.
- This requires knowledge of how the accelerated CMS works!

seeweb

TLS termination

nginx

- Another HTTP reverse proxy.
- Something was needed because Varnish does not implement TLS.
- So we choose nginx because it can provide other features (like ModSecurity).

TLS certificates

They can be installed statically or requested automatically from Let's Encrypt.

Added value: customers benefit from an highly tuned configuration which implements the most recent TLS features both for optimized performances and modern security.

seeweb

Content

seeweb

Web Application Firewalls

What is a WAF?

- An application level firewall which monitors and blocks HTTP traffic.
- Interposed between a server and the (possibly malicious) world.

ModSecurity

- World's most used WAF.
- An add-on module for web servers (here: nginx).
- Operates by rules defined with a declarative event-driven language.

seeweb

Why use a WAF?

- Satisfy compliance or business requirements.
- Have a second layer of defense against configuration mistakes.
- Protect from 0-day (unknown) attacks.
- Protect insecure software that cannot be patched.

Some reasons to run insecure web applications:

- Fixes not yet available or not compatible with other software.
- Fixing old unmaintained software may be too much expensive.
- Change management procedures not compatible with the harsh reality of 0-day attacks.
- You do not actually know that they are vulnerable!

seeweb

A WAF should block unwanted traffic

Custom WAF

- Operates strictly on the basis of allow lists.
- Expensive to configure, requires a deep knowledge of the protected application.

Generic WAF

- Generic rules try to block bad traffic.
- Usually no customization is needed.

We choose the second approach and developed a custom ruleset.

seeweb

Our ModSecurity rules

Scoring rule set

- Generic rules detect the basic characteristics of exploit attempts.
- Constraints and a scoring-based approach minimizes false positives.

Customization is still possible

- Custom *virtual patch* rules can prevent exploiting known vulnerabilities.
- Rules causing false positives can be disabled.

seeweb

Content

seeweb

Load balancing

Load balancing in the Web Accelerator:

- Can direct traffic to multiple backends.
- Can detect when a backend is broken, stop using it and retry the request.
- Can route requests to different backends depending on complex criteria.

Even if all backends are down it can still serve stale pages from the cache: many outages become invisible to users.

seeweb

High availability

High availability

- Local: two redundant appliances can be deployed in a data center.
- Geographical: appliances in different data centers can share the traffic and be one the backup of the other.

Anycast architecture

Having data centers and peerings both in Milano and Rome allows to serve local users with a lower latency.

seeweb

Emergency caching and traffic scrubber

Sometimes customers receive unexpected levels of traffic, both legitimate and from DoS attacks.

The emergency cache is an almost standard Web Accelerator

- Receives the traffic with no need for DNS or backend changes by using flowspec on the border routers.
- Will magically proxy to the backend traffic for unrelated virtual hosts for which it does not have a TLS certificate.
- Implements a variety of tecniques to stop malicious bots.
- Can be enabled in a few minutes.

seeweb

Modules

Optional features:

- Detection of mobile browsers.
- Block bad bots.
- Filter cookies or the Vary header.
- Edge Side Includes (ESI).
- Geoblocking or geoallowing.
- Disallow hotlinking.
- Negotiation of WebP support.
- Fail2ban.
- JWT authorization of requests.
- Integration with Botguard.

seeweb

Future features

- Customer console to access the WAF logs (?).
- haproxy for TLS termination?
- What else do you need?

seeweb

Any questions?

https://www.linux.it/~md/text/wa-mspday2024.pdf

(Google ... Marco d'Itri ... I feel lucky)

seeweb