Comparison of resiliency on content management software systems

Aleksi Suomalainen, *CTO, Necunos Ltd.*

1 Introduction

Current content management platforms suffer from multiple issues regarding data integrity and content consistency. Necunos has made a novel solution to make content issuance more streamlined and highly resilient to both adversary modification and availability.

Threaths of current world are varied and dangerous. It is easy for state or professional adversaries to prevent information flow from factual source to information consumers with a simple distributed denial of service attack. Only trigger to engage an attack like this is money, rest is done in an automated and destructive way through botnets.

In this paper, author shall provide data regarding our novel solution compared to a mainstream solution which can, from this data, be deduced to be inferior. Our solution comprises of service made in just static files, with a separate publishing system, without any database in the served files. This also means that no executable code is run in the webserver host.

Additional hosting needs are available on demand, rather than by default unlike in many other non-resilient mainstream solutions. Costs for hosting information can thus be dramatically reduced, rather than applying additional server racks when traffic exceeds expectations.

2 Measurements

Diagnostics were run against our solution and Wordpress. A high quality diagnostic tool was used for this, namely a web server load tool wrk2. Both runs were using same parameters and were run on the same host without any pre-existing load being on the host. Also content was exactly the same on both platforms, meaning that articles and posts used exactly the same titles and formatting.

Command line parameters used:

```
wrk2 -R2000 --latency -t12 -c600 -d30s
http://127.0.0.1
```

With details:
-R2000: 2000 total requests per second
–latency, produce latency percentile spectrum data
-t 12, use 12 threads
-c600, keep 600 concurrent connections open
-d30s, run a 30 second test

Host CPU is Intel(R) Xeon(R) CPU E3-1240 v5 @ 3.50GHz, mainboard X11SAE-M and 32GiB System Memory. Docker was used to run both platforms.

Both diagnostic runs produced percentile spectrum dataplot which was then plotted with GNUPlot to preserve maximum reproduceability. Percentile spectrums are simple logarithmic dataplots which determine where most of datapoints are within certain predetermined intervals.

• *Aleksi Suomalainen is working as CTO at Necunos Ltd.. E-mail: aleksi.suomalainen@necunos.com*

2.1 Wordpress

Following data was gathered from the Wordpress host run. On average, the latency was 7.89 seconds, with 4.33 second standard deviation and maximum of 26.69 seconds. 70.89% of deviation. Requests per second was, on average 71.92, with 28.17 requests per second standard deviation, with maximum 110 requests per second. 75.00% of deviation.

In a 30.04 second run of diagnostic, 18157 requests were generated, 496.73MB of data was read. The HTTP server used with Wordpress had 14818 non-2xx or 3xx responses, meaning that the server failed to reply with a successful response code 81.61% of the test run.

Simple latency distribution:

Percentage	Latency
50.000%	7.34s
75.000%	9.74s
90.000%	14.68s
99.000%	20.25s
99.900%	23.92s
99.990%	29.64s
99.999%	29.70s
100.000%	29.70s

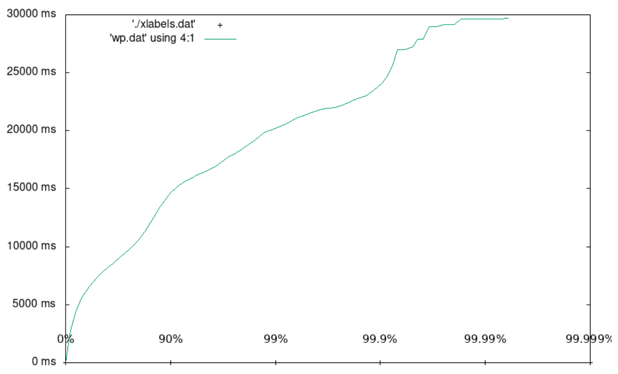

Fig. 1. Wordpress load percentile spectrum

2.2 Resilient Content Management

Necunos developed Resilient Content Management platform produced the following information. Latency was on average, 1.40 milliseconds, with 1.13 milliseconds standard deviation and maximum of 9.76 milliseconds of latency. 89.40% deviation. Requests per second were on average, 175.54 with 227.31 standard deviation and maximum of 4550 requests per second. 98.73% deviation.

During the 30.22 seconds of diagnostic run, 59709 requests were generated. All requests were replied to with success and 368.93MB was read.

Simple latency distribution:

Percentage	Latency
50.000%	1.14ms
75.000%	1.54ms
90.000%	2.40ms
99.000%	6.31ms
99.900%	8.23ms
99.990%	9.47ms
99.999%	9.77ms
100.000%	9.77ms

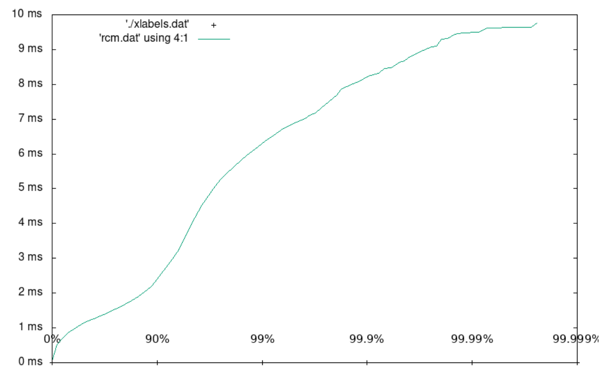

Fig. 2. Resilient Content Management percentile spectrum

3 Discussion

It is important to note that all requests were done on non-encrypted connections, namely HTTP without SSL encryption. This does induce additional penalty to performance but as can be seen, it is already slow even without. Resilient Content Management can handle the incurred penalty.

With the previous results, following observations must be made. First of all, any default installation of Wordpress Docker image is unable to respond to unreasonable workload in time. This is quite critical since Wordpress is widely used in context of spreading information.

Second, a three orders of magnitude decrease in latency is important with increasing number of attacks directed towards information providers and their services. This means that all requests for information can be handled successfully even in a hostile denial-of-service attack. Quick response times mean that attackers cannot manipulate large request referrals as easily.

Third, the hosting needs of our software solution are very minimal and require no proxying unless more capable hosting is required. Databases, server side code execution and dependencies generated by current solutions are not and will never be sustainable in the world of information influence.

3.1 Acknowledgments

Necunos would like to thank Faktabaari for sponsoring and launching this product on their site.

4 References

Faktabaari, https://faktabaari.fi
Wordpress, https://wordpress.org
WRK2, https://github.com/giltene/wrk2