Future of ZLint

Zakir Durumeric

CA/Browser Forum Meeting | June 10, 2025 | Toronto, CA

ZLint

ZLint is a X.509 certificate linter written in Go that checks for consistency with:

- RFCs (e.g. RFC 6818, RFC 4055, RFC 8399)
- CABF Baseline Requirements
- CABF EV SSL Certificate Guidelines
- Mozilla's PKI policy
- Apple's CT policy
- ETSI ESI (Limited)

It can be used as a command line tool or as a library integrated into CA software.

Maintained under Apache 2.0 License on Github

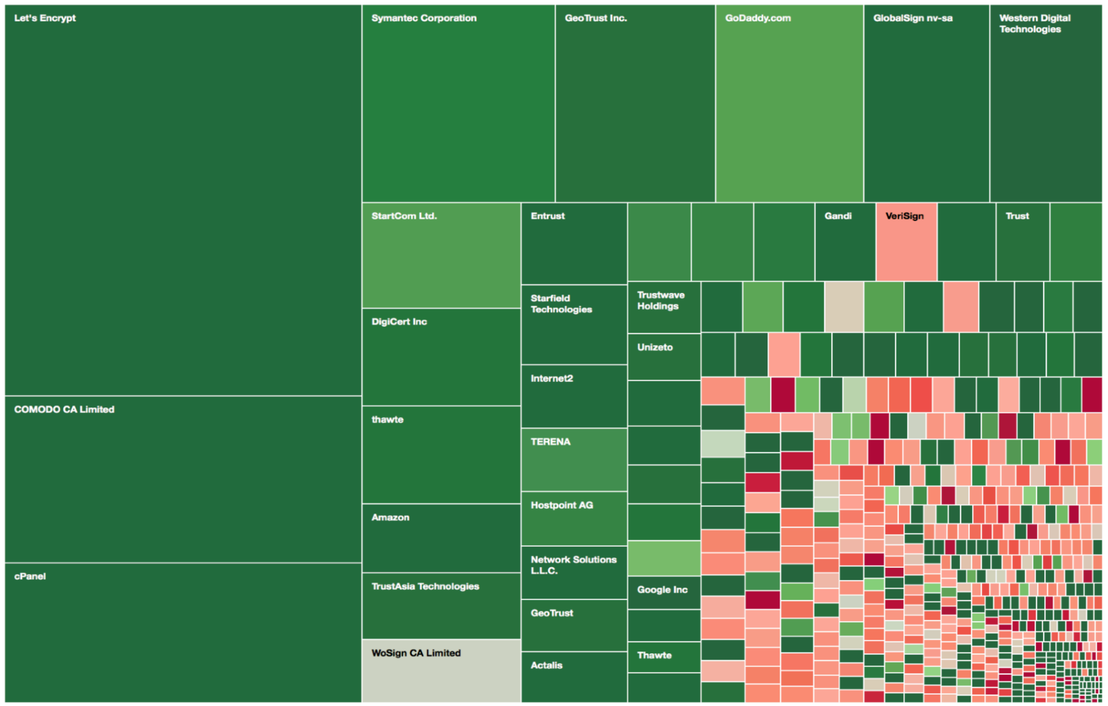
Zlint Users/Integrations

Pre-issuance linting is **strongly recommended** by the Mozilla root program. Here are some projects/CAs known to integrate with ZLint in some fashion:

- Actalis
- ANF AC
- Camerfirma
- CFSSL
- Digicert
- EJBCA
- Entrust
- Globalsign
- GoDaddy
- Google Trust Services
- Government of Spain, FNMT
- Izenpe
- Let's Encrypt and Boulder
- Microsec
- Microsoft
- Nexus Certificate Manager
- QuoVadis
- Sectigo, crt.sh, and pkimetal
- Siemens
- SSL.com
- PKI Insights
- NETLOCK
- Disig

Origin of ZLint Project

ZLint was originally developed as part of an academic research paper to quantify CA adherence to RFCs and CA/B Forum BRs



Tracking Certificate Misissuance in the Wild

Deepak Kumar*, Zhengping Wang*, Matthew Hyder*, Joseph Dickinson*, Gabrielle Beck†, David Adrian†, Joshua Mason*, Zakir Durumeric*†‡, J. Alex Halderman†, Michael Bailey*

* University of Illinois Urbana-Champaign † University of Michigan ‡ Stanford University

***Abstract*—Certificate Authorities (CAs) regularly make mechanical errors when issuing certificates. To quantify these errors, we introduce ZLint, a certificate linter that codifies the policies set forth by the CA/Browser Forum Baseline Requirements and RFC 5280 that can be tested in isolation. We run ZLint on browser-trusted certificates in Censys and systematically analyze how well CAs construct certificates. We find that the number errors has drastically reduced since 2012. In 2017, only 0.02% of certificates have errors. However, this is largely due to a handful of large authorities that consistently issue correct certificates. There remains a long tail of small authorities that regularly issue non-conformant certificates. We further find that issuing certificates with errors is correlated with other types of mismanagement and for large authorities, browser action. Drawing on our analysis, we conclude with a discussion on how the community can best use lint data to identify authorities with worrisome organizational practices and ensure long-term health of the Web PKI.**

I. Introduction

HTTPS depends on a supporting public key infrastructure (PKI) composed of hundreds of certificate authorities (CAs) that verify the identities of websites and issue digital certificates. To ensure compatibility between browsers and HTTPS-enabled websites, standards bodies like the IETF and CA/Browser Forum have developed policies that govern the digital certificates that CAs provide. Unfortunately, there is a long history of certificate authorities failing to adhere to accepted standards, due to both implementation errors and indifference. In this paper, we systematically analyze the errors that authorities make when constructing certificates and consider whether these errors can be used to predict more serious problems.

We begin by dissecting the policies set forth by RFC 5280 [14] and CA/Browser Forum Baseline Requirements [9]. We find that many aspects of certificate construction can be checked in isolation, and we codify these requirements in a set of 220 lints. We introduce and release ZLint, a Go-based linting framework that implements these checks and provides structured data on certificate construction and standards adherence.

To quantify misissuance (i.e., certificates with errors) in the Web PKI, we run ZLint on the 240 million browser-trusted certificates in Censys [17]. We find that misissuance is low in aggregate. Only 0.02% of certificates violate one of the two standards in 2017; 3.3% do not adhere to community best practices. This is a significant improvement from 2012 when more than 12% of certificates contained errors and nearly one third violated community recommendations. However, while the global misissuance rate is low, this is predominantly due to a handful of large authorities that consistently issue certificates without error. The three largest CAs by organization—Let's Encrypt, Comodo, and cPanel—signed 80% of the certificates in our dataset and have near-zero misissuance rates. Let's Encrypt, the largest CA by number of certificates issued, has a particularly stellar incident rate. Of the 37 million certificates the CA has signed, only 13 contain errors. None have warnings.

The bulk of misissuance is due to two classes of authorities. The first class is mid-sized authorities that make a variety of errors in a small percentage of their certificates. The second class is a long tail of small authorities that make the same errors in every issued certificate. Nearly half of the organizations in our dataset misissue more than 10% of certificates, and seventeen have made errors in every certificate. More than half of the errors and warnings in ZLint are triggered at least once. Most often, authorities fail to fully populate the Subject Alternative Names extension, encode the wrong type of data in the extension, or include invalid DNS names. Beyond individual certificates, we find that many organizations struggle to properly maintain OCSP/CRL responders. During our three week test period, the OCSP responders for 73 organizations (10%) failed every health check.

Next, in order to determine whether Lint data can be used to predict more serious issues, we investigate the correlation between the organizations that issue certificates containing errors, OCSP/CRL endpoint uptime, and browser removal. We find that there is weak correlation between the organizations that issue certificates with errors and OCSP availability. For authorities that have issued more than 100K certificates, there is moderate to strong correlation between ZLint-identified misissuance and browser removal. Surprisingly, while there is discussion about large CAs with high error profiles, there is no correlation between the small authorities making errors and discussion in the community (e.g., in the Mozilla Developer Security Policy mailing list).

Our results indicate that large authorities are making progress in correctly issuing certificates. However, there remains a long tail of small authorities that fail to follow community standards and misissue most certificates. Most of these small authorities are not being actively discussed. We hope that by shedding light on these practices, we motivate the community to investigate struggling authorities and prompt discussion on whether lint data can be systemically used to help prevent future PKI incidents. Finally, by releasing ZLint, we hope to

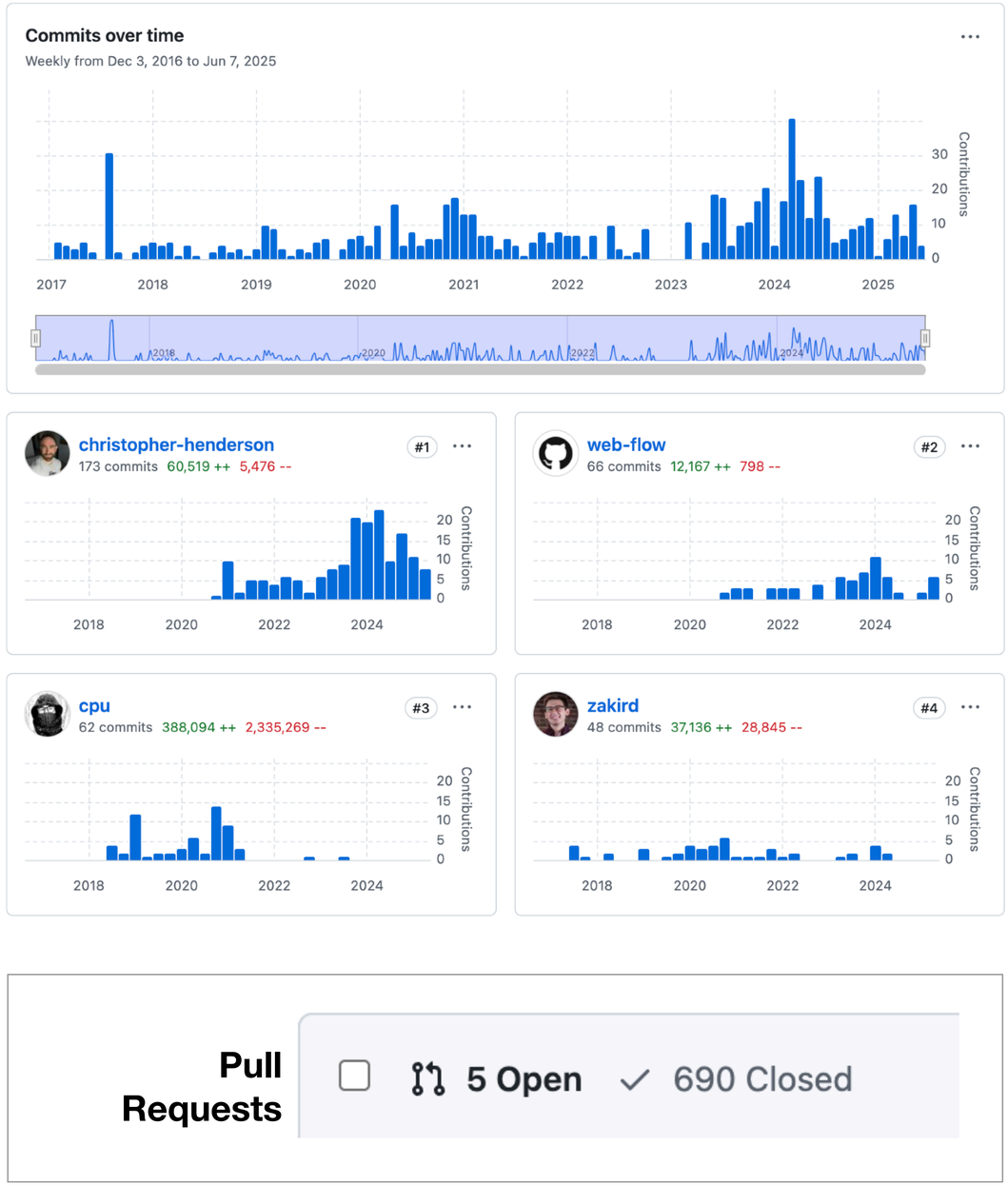
Project Evolution

After release, we initially did well keeping up with changing BRs and solidifying both code base and testing (h/t Daniel McCarney @cpu)

Beyond BRs + RFCs, we added support for:

- CABF EV SSL Certificate Guidelines
- ETSI ESI (Partial)
- Mozilla's PKI policy
- Apple's CT policy

Today, most development is best effort work from Chris Henderson (supported by Digicert) and ad-hoc contributions from others



Project Challenges

We have fallen behind in several important areas:

- Baseline Requirements 2.0+ and Certificate Profiles
- ASN.1-Based Lints
- ETSI Requirements

We have several broader project challenges as well:

- We don’t have on-time-releases corresponding to policy changes
- Most code contributions are below bar and require significant effort to merge
  - Many PRs don’t match policies, are broken, and are poorly implemented
  - Led to burnout from many of the community maintainers

Where do we go from here?

A strong open source community has yet to emerge to maintain ZLint at the standard that we suspect is needed by the WebPKI

If we want continuously released up-to-date linter for the ecosystem:

- ZMap Foundation is happy to serve as an organizational home, but we cannot support continued development by ourselves
- There would need to be funding from CAs to support an FTE resource and/or significantly different level of engagement

Evolution to additionally support lints for CPS?

We are seeking feedback from this community on how you would like to see ZLint evolve

Zakir Durumeric
Stanford University

zakir@cs.stanford.edu