Using Frankencerts for Automated Adversarial Testing of Certificate Validation in SSL/TLS Implementations

Chad Brubaker Suman Jana Baishakhi Ray
Sarfraz Khurshid Vitaly Shmatikov

116033910063
黄中月

Content

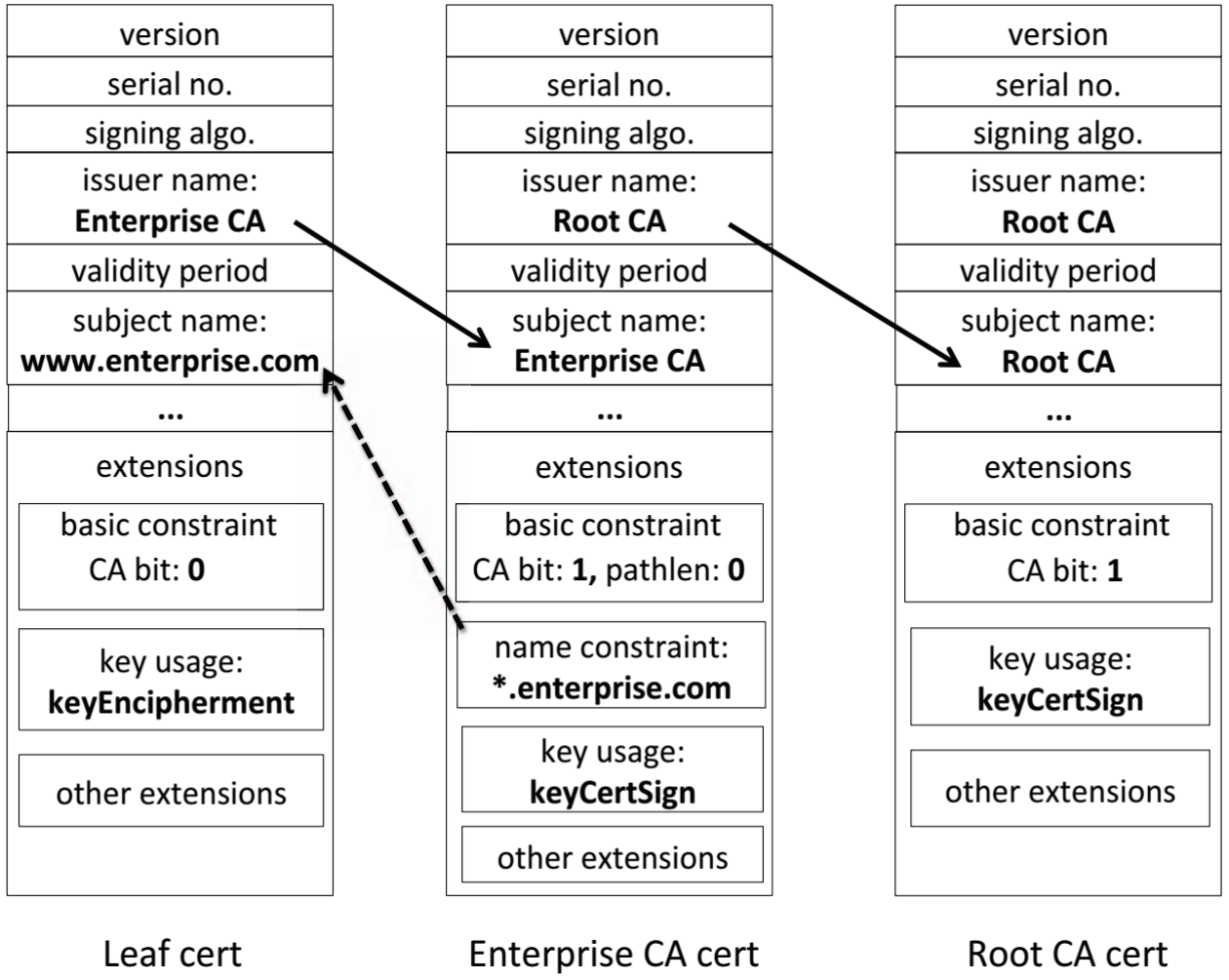
- SSL/TLS Protocol
- Implementation Correctness
- Certificate Generation
- Differential Testing
- Conclusion

SSL/TLS Protocol

- End-to-end security even if the network is insecure
  - Authentication = certificate validation
  - Confidentiality
  - Integrity

SSL/TLS Protocol

- Server authentication
  - X.509 certificate validation
    - Chain of trust
    - Basic constraints
    - Name constraints
    - Key usage
    - Hostname
    - Time
    - …



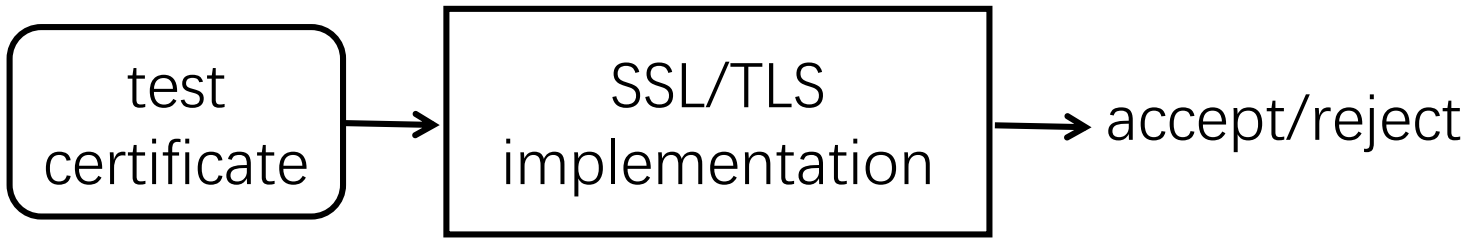
Implementation Correctness

- Problem1: generating test inputs
  - Structurally complex data = Huge input space

- Approach
  - Simple automated technique (Ex: random fuzzing)
    - A fuzzed string won't even parse as an X.509 cert
  - Manually creating certificates
    - Manually creating a high-quality suite is simply infeasible

Implementation Correctness

- Problem2: interpreting test results



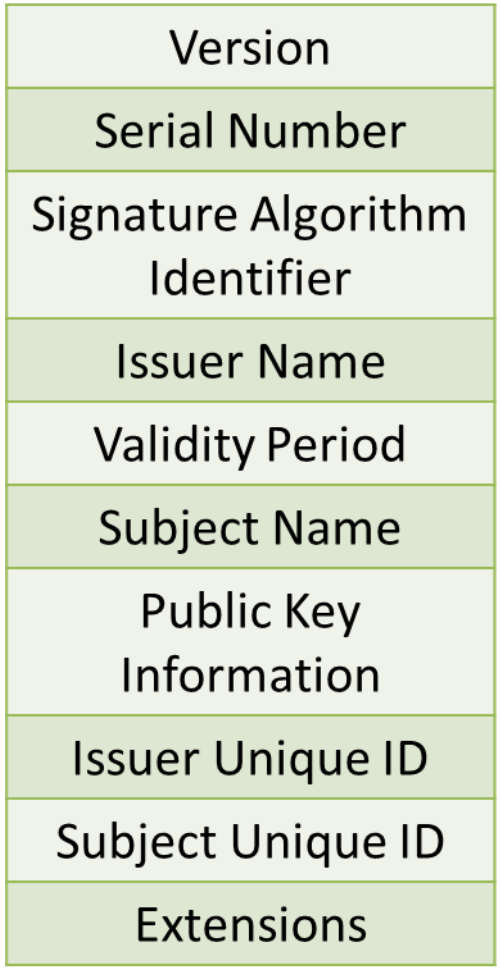
Implementation Correctness

- Problem1: generating test inputs
  - Frankencerts
- Problem2: interpreting test results
  - Differential Testing

Certificate Generation

- Requirements
  - Syntactically correct
  - Semantically bad
  - Scale to millions of certs

- X.509 certificate structure
  - Multilayered structured data
  - Syntactic constraints
    - Ex: Version must be an integer
  - Semantic constraints
    - Ex: Version must be 0, 1, or 2

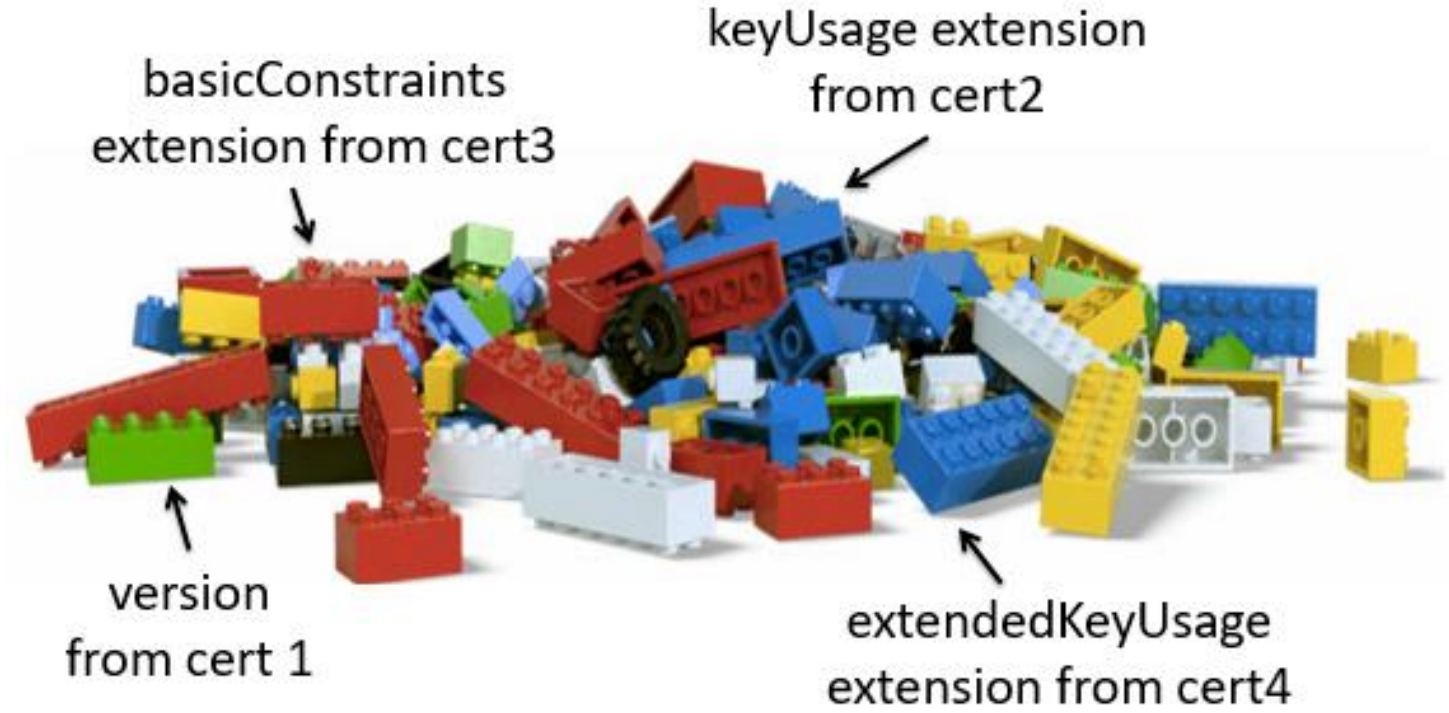


Certificate Generation

- Step 1: collect 243,246 certificates

Certificate Generation

- Step 2: generate 8,127,600 frankencerts



Certificate Generation

- Step 3: mutate a few pieces

Differential Testing

- 9 open-source SSL/TLS libraries

- 6 Web browsers

Differential Testing

- Results
  - 15 root causes
  - 208 discrepancies
  - 62,022 frankencerts

- Error Reporting
  - Expired (E)
  - Bad issuer (I)
  - Bad name (N)

Problem	Certificates triggering the problem occur in the original corpus	OpenSSL	PolarSSL	GnuTLS	CyaSSL	MatrixSSL	NSS	OpenJDK, Bouncy Castle	Browsers
Untrusted version 1 intermediate CA certificate	No	reject	reject	**accept**	reject	**accept**	reject	reject	reject
Untrusted version 2 intermediate CA certificate	No	reject	reject	reject	reject	**accept**	reject	reject	reject
Version 1 certificate with valid basic constraints	No	accept	reject	accept	accept	accept	reject	reject	Firefox: reject Opera, Chrome: accept
Intermediate CA not authorized to issue further intermediate CA certificates, but followed in the chain by an intermediate CA certificate	No	reject	reject	reject	reject	**accept**	reject	reject	reject
...followed by a leaf CA certificate	No	**reject**	**reject**	accept	**reject**	accept	**reject**	**reject**	**reject**
Intermediate CA not authorized to issue certificates for server's hostname	No	reject	reject	**accept**	**accept**	**accept**	reject	reject	reject
Certificate not yet valid	Yes	reject	**accept**	reject	reject	reject	reject	reject	reject
Certificate expired in its timezone	Yes	reject	**accept**	reject	reject	**accept**	reject	reject	reject
CA certificate not authorized for signing other certificates	No	reject	reject	**accept**	**accept**	**accept**	reject	reject	reject
Server certificate not authorized for use in SSL/TLS handshake	Yes	reject	**accept**	**accept**	**accept**	**accept**	reject	reject	reject
Server certificate not authorized for server authentication	Yes	reject	**accept**	**accept**	**accept**	**accept**	reject	reject	reject
Certificate with unknown critical extension	No	reject	reject	**accept**	**accept**	**accept**	reject	reject	reject
Certificate with malformed extension value	No	accept	reject	accept	accept	accept	reject	reject	reject
Certificate with the same issuer and subject and a valid chain of trust	No	reject	reject	**accept**	reject	**accept**	reject	reject	reject
Issuer name does not match AKI	No	reject	accept	accept	accept	accept	reject	reject	reject
Issuer serial number does not match AKI	No	reject	accept	reject	accept	accept	reject	reject	reject

Differential Testing

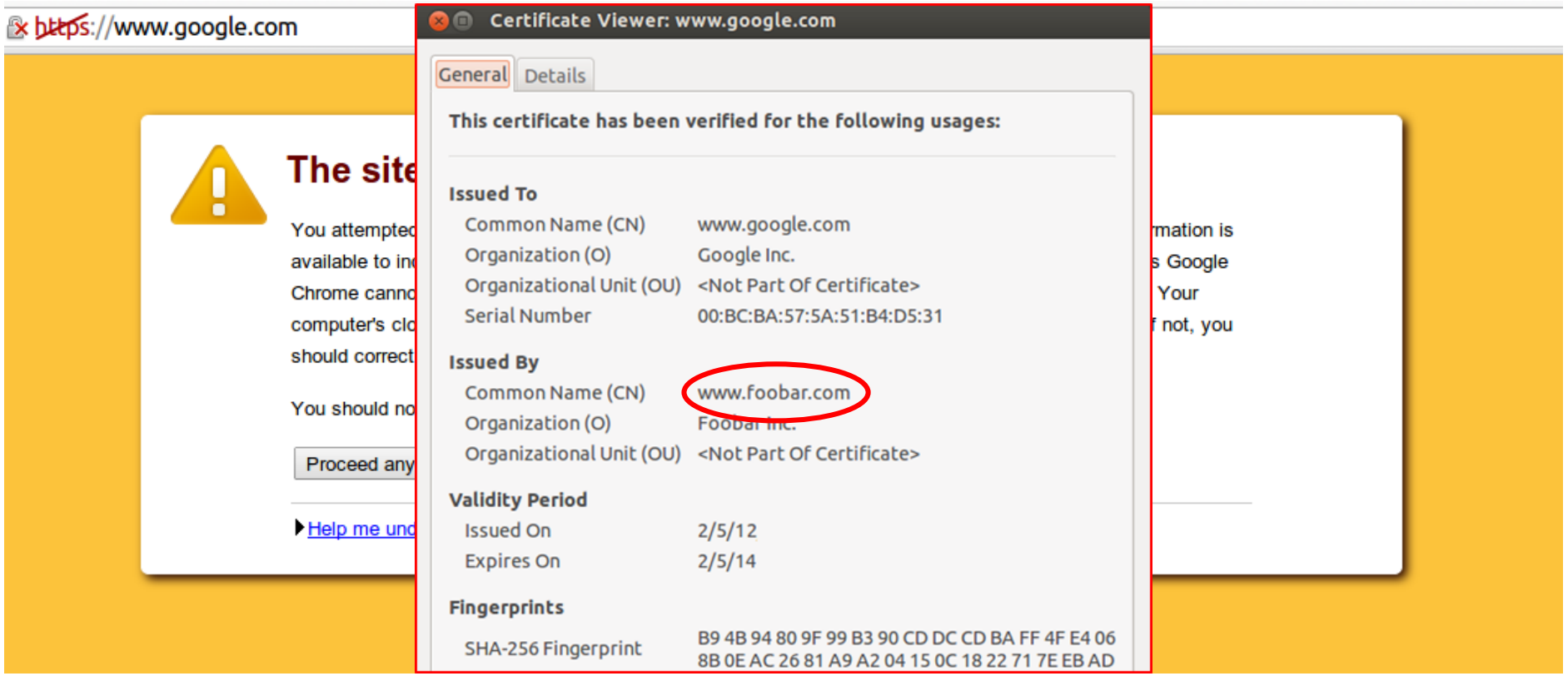
- Error Reporting

Certs	Firefox 20	Chrome 30 (Linux)	Opera 12 (Linux)	Opera 20 (Mac)	Safari 7	Chrome 30 (Mac)	IE 10	OpenSSL	PolarSSL	GnuTLS	CyaSSL	MatrixSSL	NSS
E	E	E	E	!E	!E	E	E	E	E	E	E	E	E
I	I	I	I	!I	!I	I	I	I	I	I	I	**	I
IE	IE	**E**	I#	*	**!E**	*	*	I	I	IE	**	**	**E-**
IN	IN	I	I#	!I	!I	I	IN	I-	I-	I-	I-	*-	I-
IEN	IEN	**N**	I#	*	**!E**	*	*	I-	IE-	**-	**-	**-	**E-**
N	N	N	N	+	!N	N	N	-	-	-	-	-	-
NE	NE	N	**E#**	**!E**	**!E**	N	NE	E-	**-	E-	E-	E-	E-

***** is a generic "invalid certificate" warning without a specific error message; the user cannot override this warning
+ is a generic "invalid certificate" warning without a specific error message; the user can override this warning
****** is a generic "invalid certificate" error code
all errors are shown after the user clicks the details tab
! shows a generic error message first; the reported error is shown after user clicks the details button
- the hostname check was not enabled for any of the tested clients

Differential Testing

- Ex. Google Chrome



Conclusion

- Differential testing with frankencerts is an effective technique for finding flaws in SSL/TLS implementations

- The code is available at: https://github.com/sumanj/frankencert

Thanks

Q&A