

CSE227 – Graduate Computer Security

TLS II

UC San Diego

Housekeeping

General course things to know

- Midpoint check-in document is due **2/14 at 11:59pm PT**
 - Introduction (frame the problem)
 - Related work section (should include ~5 – 10 relevant papers)
 - Research plan, current status, what's left to do

Today's lecture

Learning Objectives

- Learn the “trust” mechanism that underlies TLS and how it works in practice
- Discuss the certificate validation paper
- Discuss the certificate misissuance paper

The Most Dangerous Code in the World: Validating SSL Certificates in Non-Browser Software

TLS Fundamentals

TLS provides three fundamental security properties:

TLS Fundamentals

TLS provides three fundamental security properties:

- Confidentiality. What is confidentiality, and how does TLS provide it?

TLS Fundamentals

TLS provides three fundamental security properties:

- Confidentiality. What is confidentiality, and how does TLS provide it?
- Integrity. What is integrity, and how does TLS provide it?

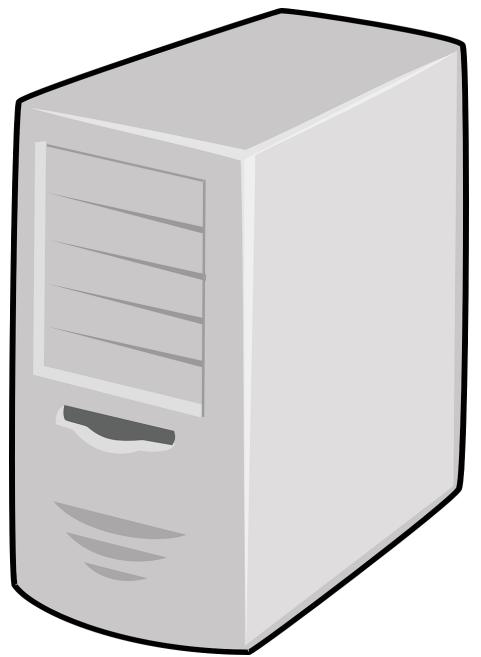
TLS Fundamentals

TLS provides three fundamental security properties:

- Confidentiality. What is confidentiality, and how does TLS provide it?
- Integrity. What is integrity, and how does TLS provide it?
- **Authenticity**
 - Validating the source or origin of data: i.e., *knowing who you are talking to*

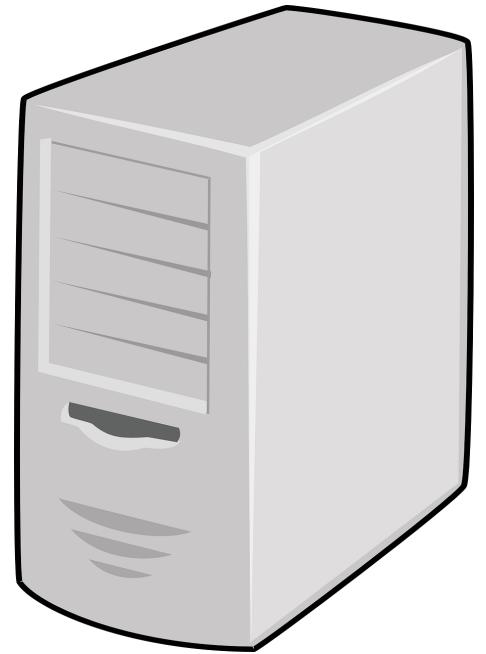
How do you get HTTPS on your website?

kumarde.com



How do you get HTTPS on your website?

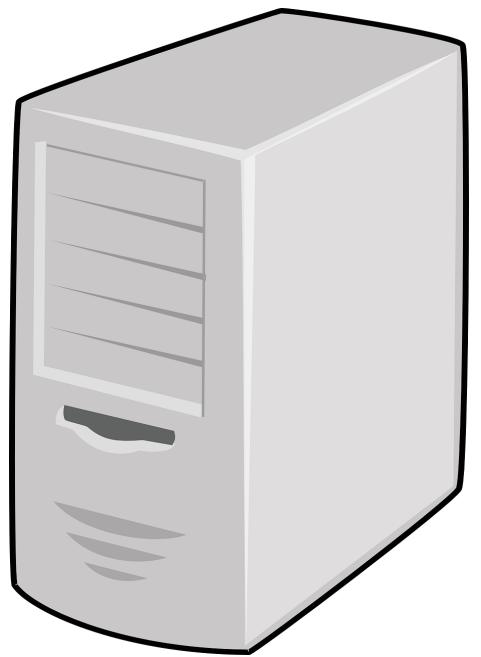
kumarde.com



What is a Certificate Authority (CA)?

How do you get HTTPS on your website?

kumarde.com

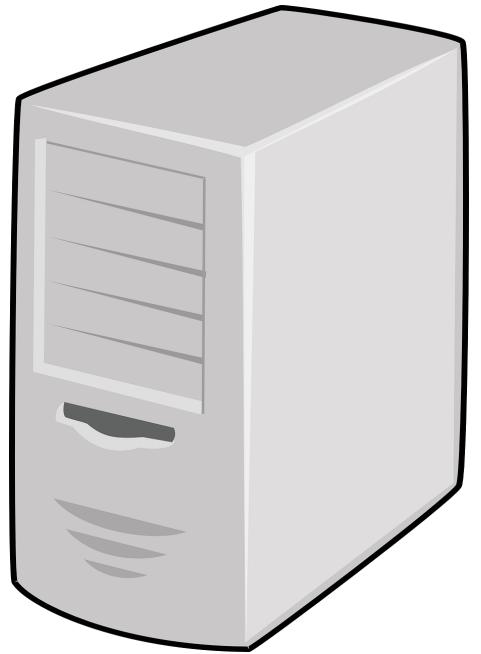


What is a Certificate Authority (CA)?

"A trusted entity that issues digital certificates to verify the identity of individuals, companies, email addresses, and websites."

How do you get HTTPS on your website?

kumarde.com

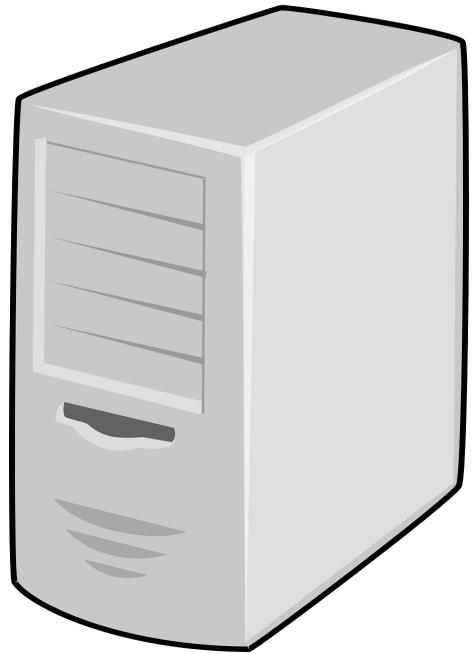


certificate for kumarde.com pls



How do you get HTTPS on your website?

kumarde.com



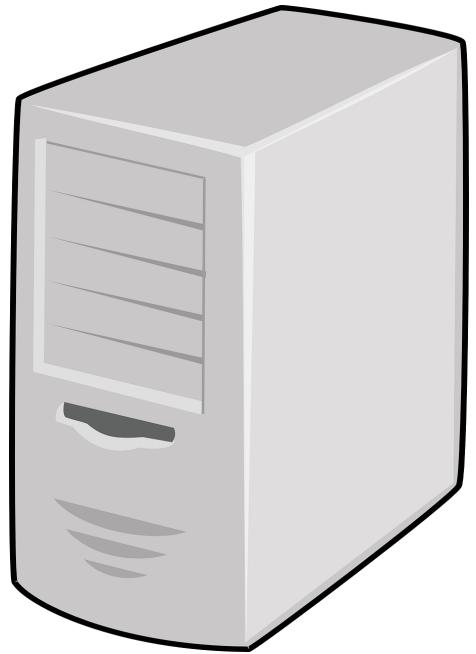
certificate for kumarde.com pls

verify who you are!



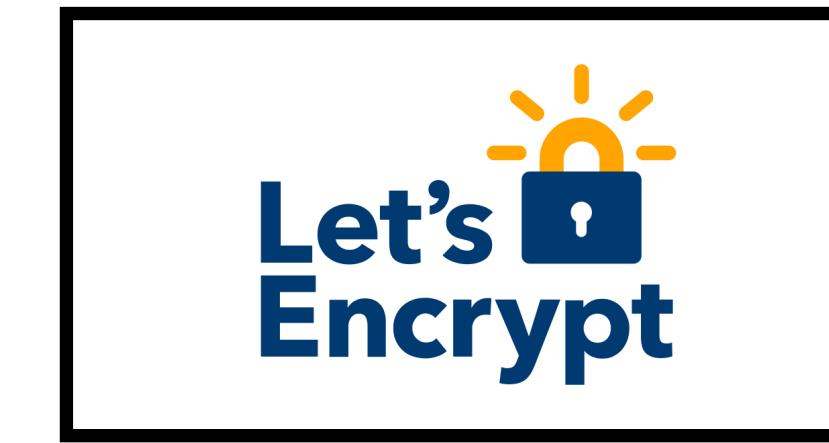
How do you get HTTPS on your website?

kumarde.com



certificate for kumarde.com pls

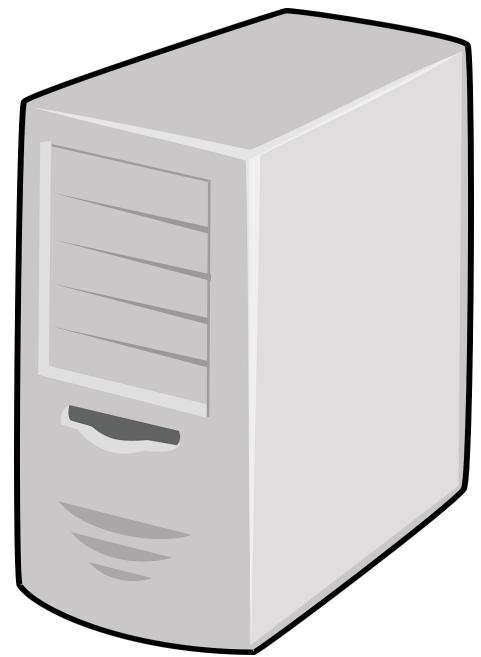
verify who you are!



What are some ways that CAs verify your identity?

How do you get HTTPS on your website?

kumarde.com



certificate for kumarde.com pls

verify who you are!

here is a cert signed by us



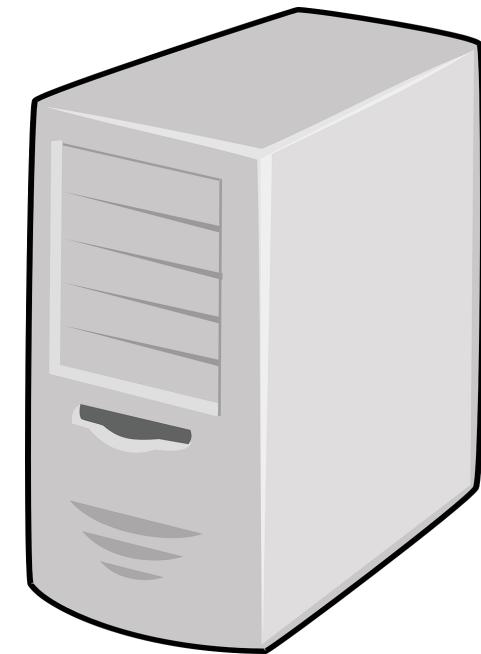
Certs have lots of details, but most importantly they have your **public key**, thereby linking your verified identity to your cryptographic identity

How do you get HTTPS on your website?

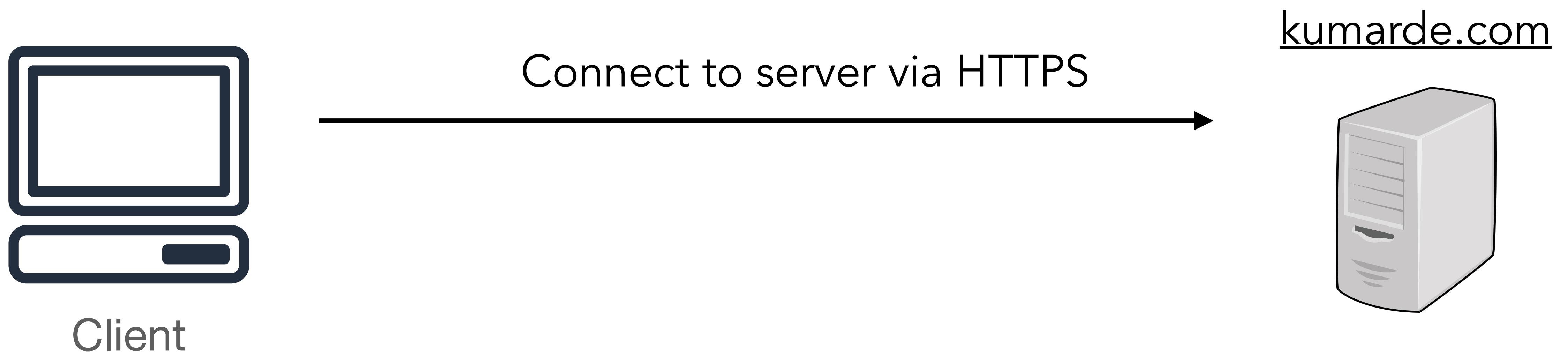


Client

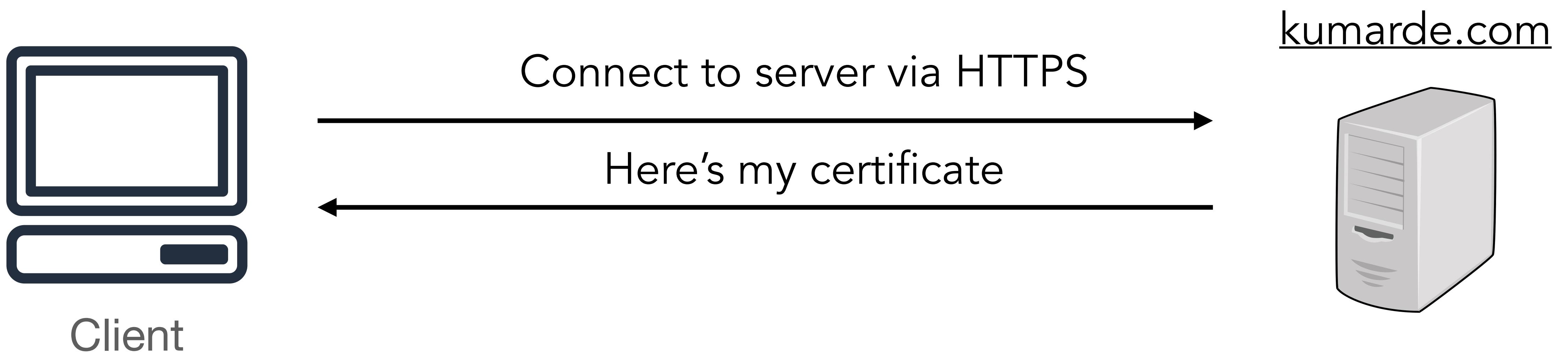
kumarde.com



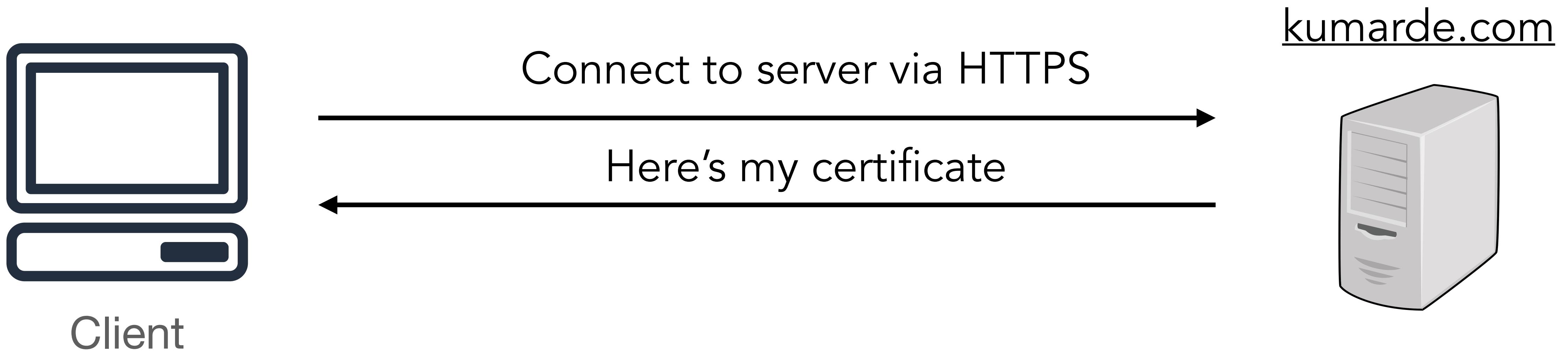
How do you get HTTPS on your website?



How do you get HTTPS on your website?



How do you get HTTPS on your website?



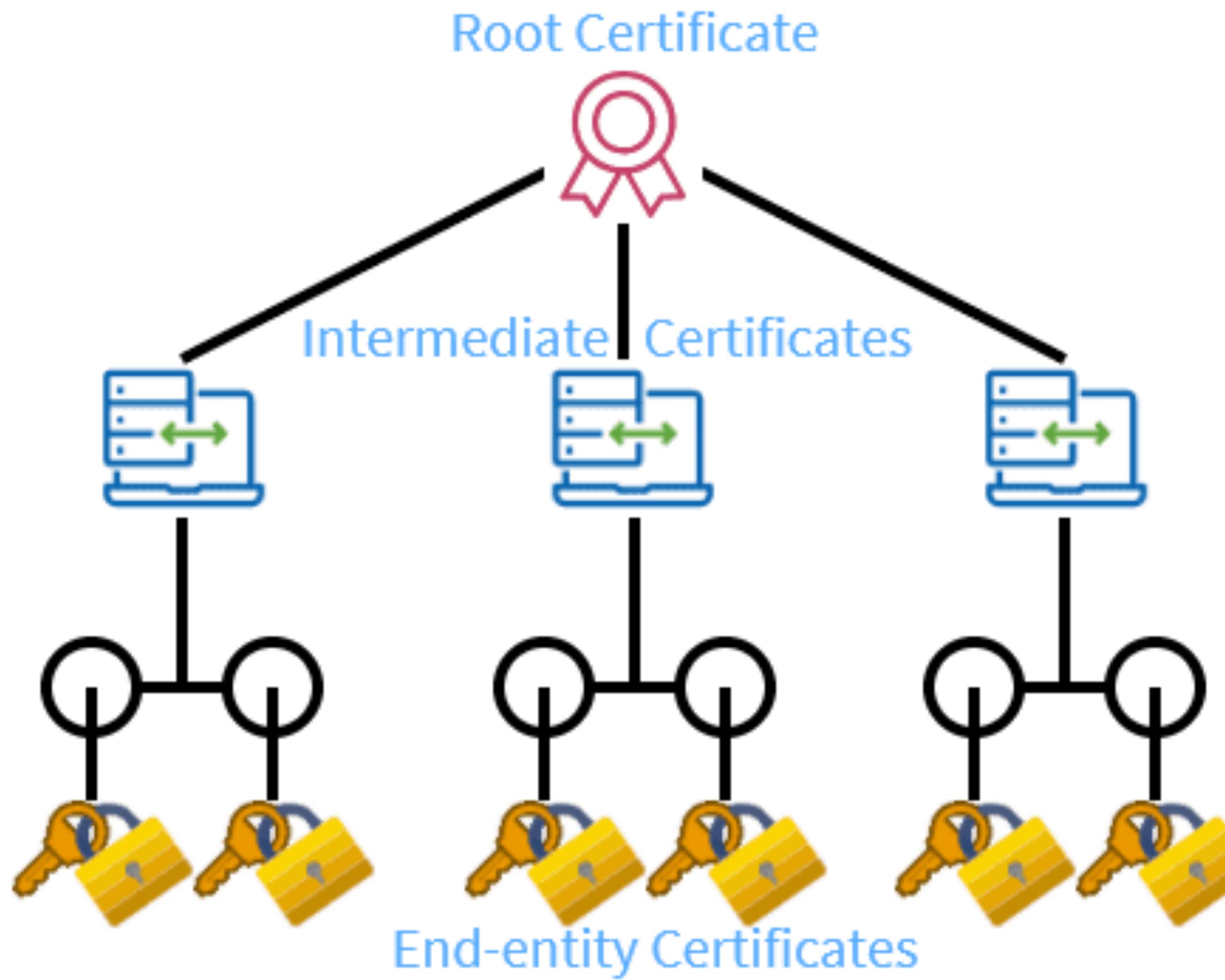
What does the client need to do with the certificate in order to trust the server on the other end?

Certificate validation

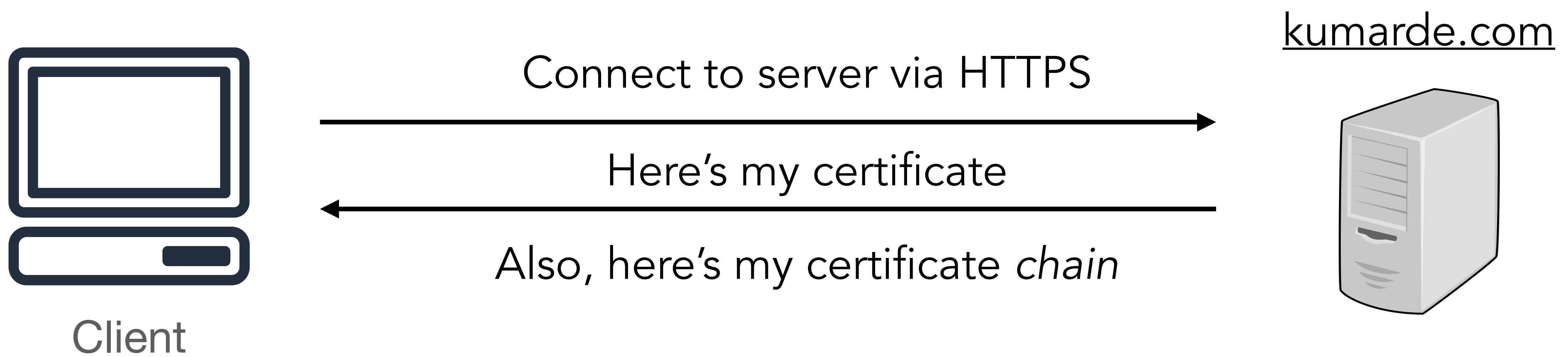
- Chain-of-trust verification
- Hostname verification
- Revocation checks

What is a chain of trust?

What is a chain of trust?



How do you get HTTPS on your website?



Who decides who the roots of trust are?

Who decides who the roots of trust are?



CA/BROWSER FORUM



Public-Key Infrastructure (HTTPS)

Otherwise known as the Web PKI

- The web PKI is the hardware, software, policies, processes, and procedures that exist to manage, distribute, and revoke web certificates and public-keys
- Every machine or software that communicates with HTTPS **comes with roots installed** – these are explicitly trusted by browsers, OSes, etc.
- Chock-full of arbitrary and complicated rules that have evolved over time due to “problems”
 - E.g., Roots can’t sign leaf certificates, roots need to have intermediates that can sign leaf certificates

Hostname verification

- What is hostname verification?
- Why might hostname verification be challenging to implement?

Hostname verification

- What is hostname verification?
- Why might hostname verification be challenging to implement?
 - Wildcards, multiple potentially conflicting names in a certificate, x509 extensions, etc.

Certificate Revocation

- What is revocation?
- How is certificate revocation implemented in practice?

Certificate Revocation

- What is revocation?
- How is certificate revocation implemented in practice?
 - CRL – Certificate Revocation List
 - OCSP – Online Certificate Status Protocol

Certificate Revocation

- What is revocation?
- How is certificate revocation implemented in practice?
 - CRL – Certificate Revocation List
 - OCSP – Online Certificate Status Protocol
- **Both are broken, revocation isn't really used today.**
- What happens when the revocation system doesn't work?

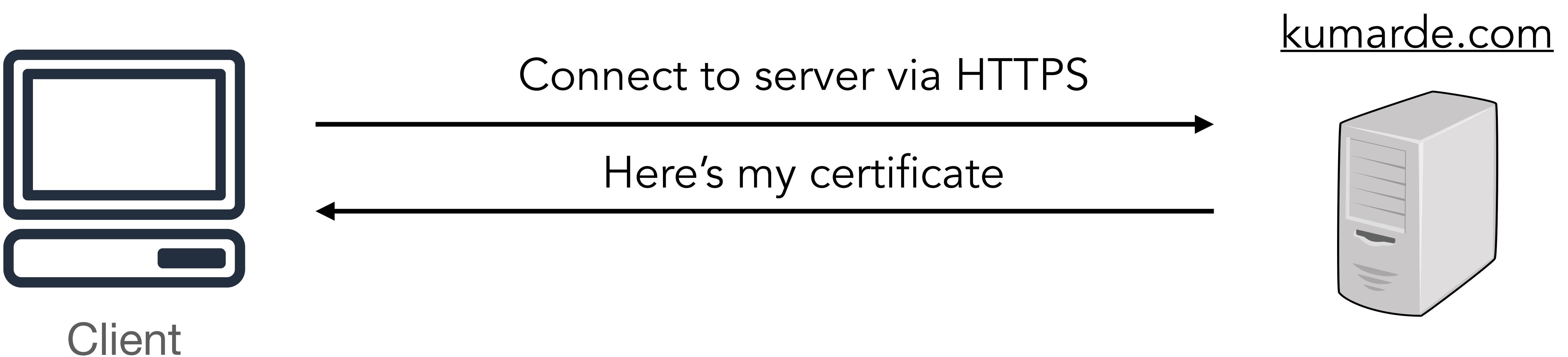
<https://scotthelme.co.uk/revocation-is-broken/>

Threat model

- What is the attack the authors are trying to pull off?

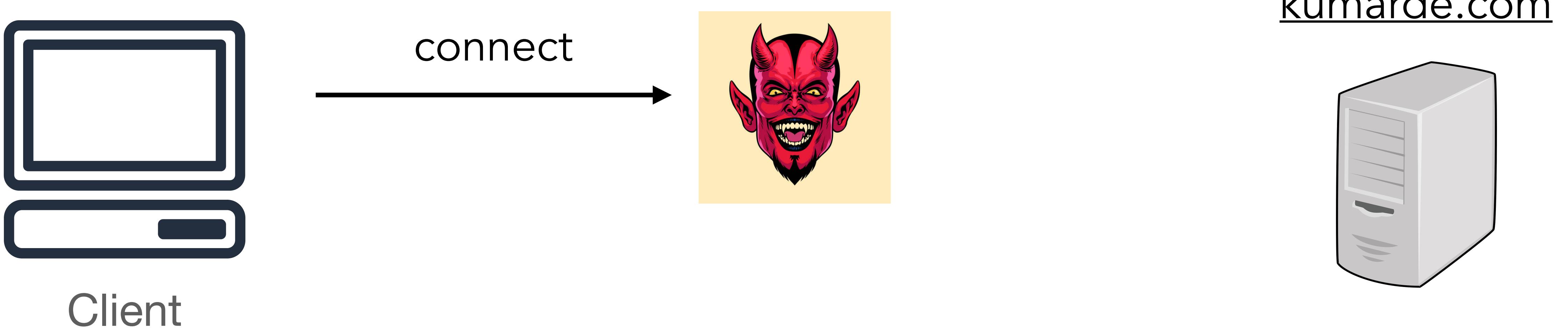
Threat model

- What is the attack the authors are trying to pull off? **Man in the middle attack.**



Threat model

- What is the attack the authors are trying to pull off? **Man in the middle attack.**



Threat model

- What are the capabilities of the attacker?

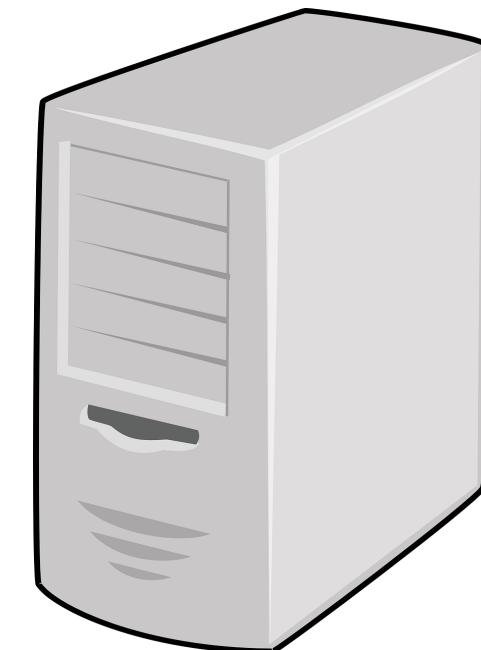


Client

connect

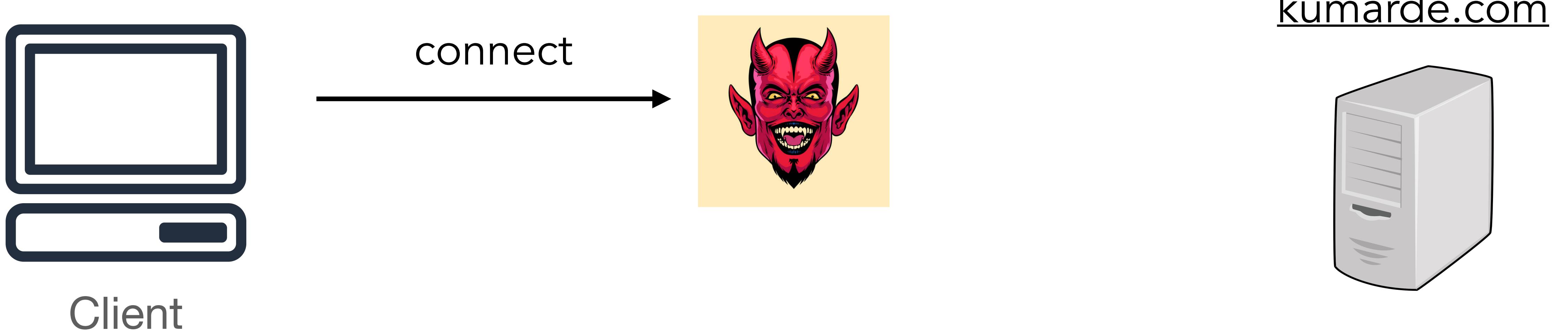


kumarde.com



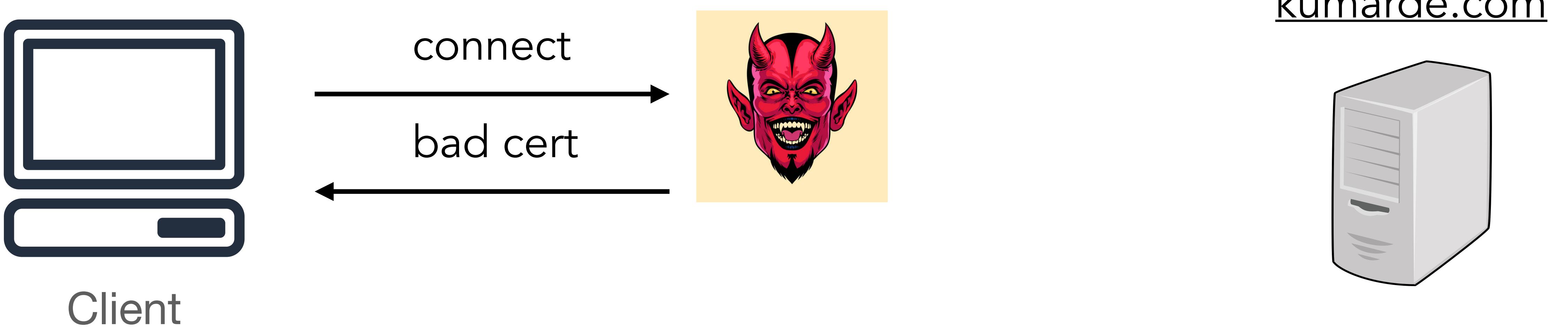
Threat model

- What are the capabilities of the attacker? **No private keys, no CAs, cannot forge certs**



Threat model

- What are the capabilities of the attacker? **No private keys, no CAs, cannot forge certs**



Evaluating Certificate Verification

- The authors used *black-box fuzzing* techniques to evaluate the certificate verification processes of dozens of software.
- What is black-box fuzzing?

Evaluating Certificate Verification

- The authors used *black-box fuzzing* techniques to evaluate the certificate verification processes of dozens of software.
- What is black-box fuzzing?
- What types of bad certs did the authors try?

Evaluating Certificate Verification

- The authors used *black-box fuzzing* techniques to evaluate the certificate verification processes of dozens of software.
 - What is black-box fuzzing?
 - What types of bad certs did the authors try?
 - Self-signed certificate with the same common name as host (e.g., kumarde.com)
 - Self-signed certificate with an incorrect common name
 - Valid certificate with an incorrect common name

Results

- Certificate validation is totally and wholly broken in many, many, many applications
 - Mostly due to *middleware* – the interface between two or more pieces of software – not communicating with each other properly
- My favorites:
 - Checking if a function returns the wrong value (e.g., expected -1 but sometimes it's null)
 - Having nonsensical defaults (i.e., SSLSocketFactory **fails-open** when there's a NULL value in a string)
 - Turning off verification altogether!

Meta-thoughts on the paper

- Many of these bugs are fixed today. So why are we reading this paper?
- What about this paper surprised you? What didn't surprise you?
- What does this paper teach us about trust?

Break Time + Attendance



Codeword:
Certainty

<https://tinyurl.com/cse227-attend>

Tracking Certificate Misissuance in the Wild

Certificate Authorities, Redux

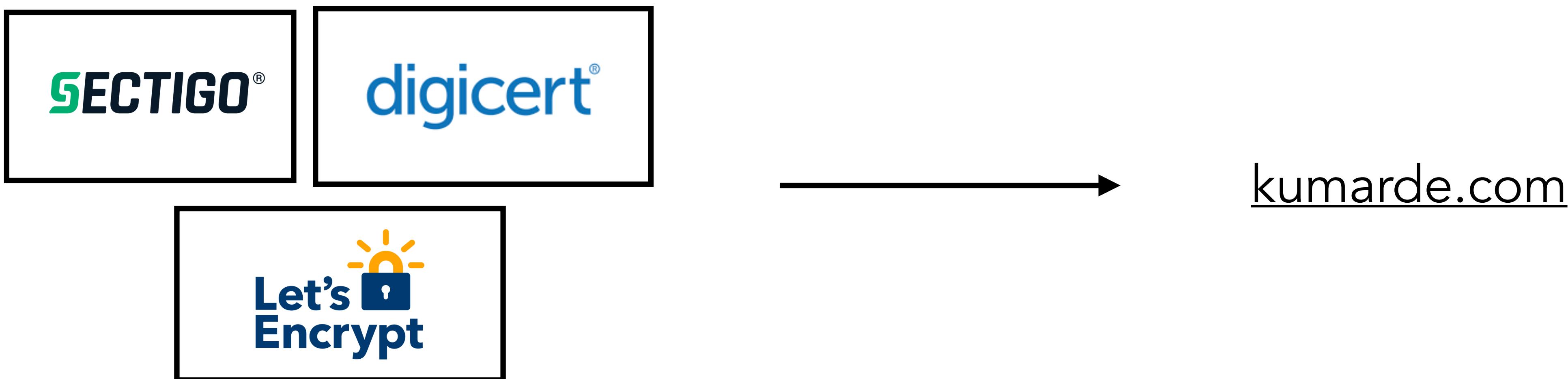
- There is a fundamental “vulnerability” with the way the certificate authority ecosystem is set up. What is it?

Certificate Authorities, Redux

- There is a fundamental “vulnerability” with the way the certificate authority ecosystem is set up. What is it?
- **Anyone trusted can sign for anyone on the Internet!**

Certificate Authorities, Redux

- There is a fundamental “vulnerability” with the way the certificate authority ecosystem is set up. What is it?
- **Anyone trusted can sign for anyone on the Internet!**



Iranian Man-in-the-Middle Attack Against Google Demonstrates Dangerous Weakness of Certificate Authorities

The TURKTRUST SSL certificate fiasco – what really happened, and what happens next?

Google Blocks Fraudulent Certificates Used by French Government

Revoking Trust in one CNNIC Intermediate Certificate

The rules that govern the PKI

- Two major sets of policies that CAs must follow:
 - CA/B Baseline Requirements
 - RFC 5280 – X509 Certificate Standard



CA/BROWSER FORUM



ZLint: An X.509 Certificate Linter

- This paper a certificate **linter**, called ZLint, that measures if a certificate is *misissued*
 - How do the authors identify if a certificate is *misissued*?



ZLint: An X.509 Certificate Linter

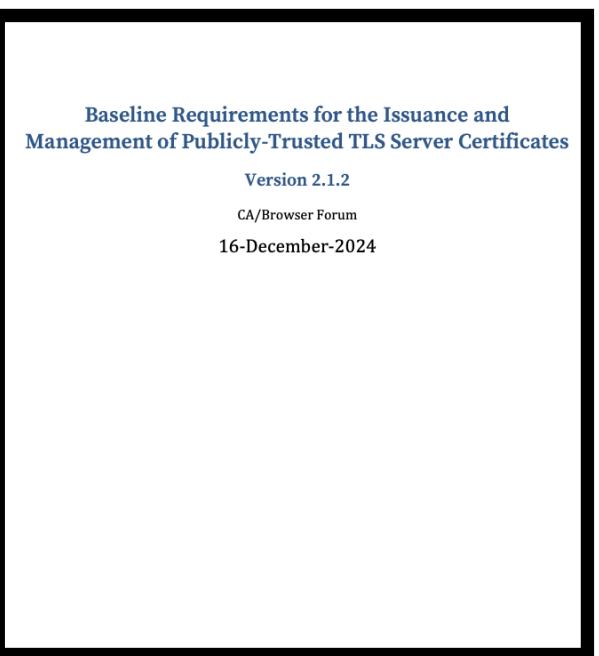
- This paper a certificate **linter**, called ZLint, that measures if a certificate is *misissued*
 - How do the authors identify if a certificate is *misissued*?

"It's 2017 - it's both time to stop making excuses and time to recognize that the ability of CAs to adhere to the rules is core to their trustworthiness. Technical rules are but a proxy for procedure rules." - Ryan Sleevi



How it really happened...

I lost my mind the summer of 2017



Rules

PhD Me

```
import (
    "strings"

    "github.com/zmap/zcrypto/x509"
    "github.com/zmap/zlint/v3/lint"
    "github.com/zmap/zlint/v3/util"
)

type DNSNameUnderscoreInTRD struct{}

func init() {
    lint.RegisterCertificateLint(&lint.CertificateLint{
        LintMetadata: lint.LintMetadata{
            Name:          "w_rfc_dnsname_underscore_in_trd",
            Description:  "DNSName MUST NOT contain underscore characters",
            Citation:     "RFC5280: 4.1.2.6",
            Source:       "lint.RFC5280",
            EffectiveDate: util.RFC5280Date,
        },
        Lint: NewDNSNameUnderscoreInTRD,
    })
}

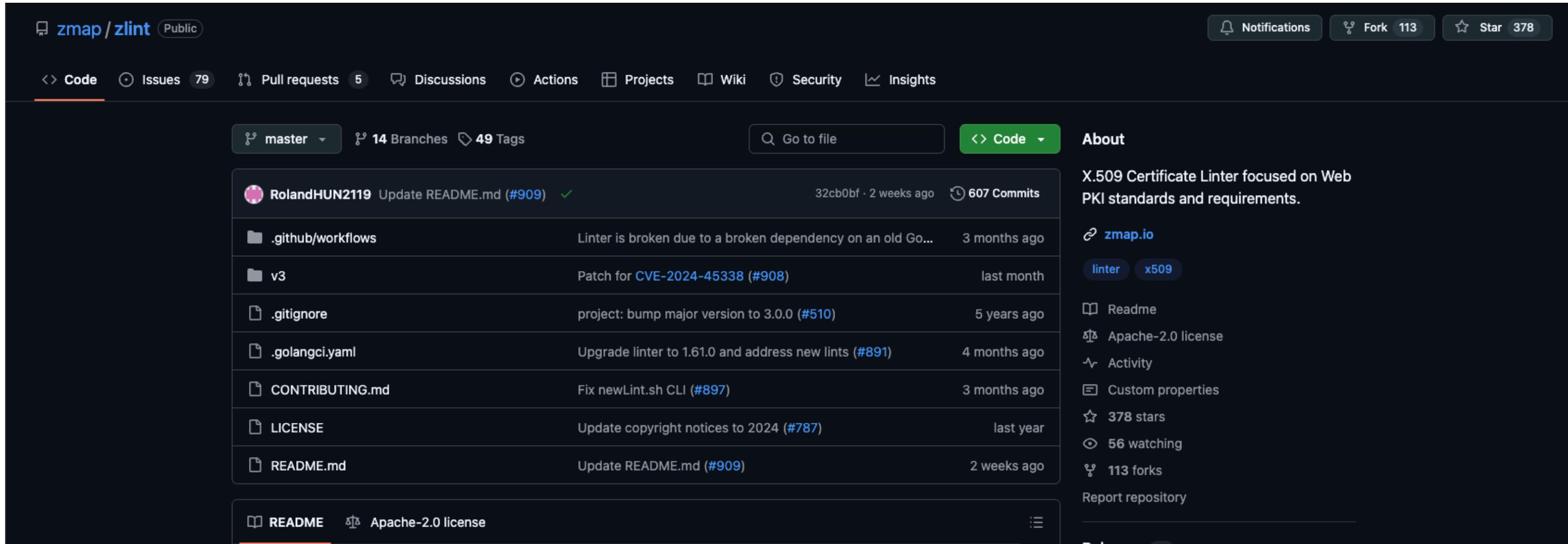
func NewDNSNameUnderscoreInTRD() lint.LintInterface {
    return &DNSNameUnderscoreInTRD{}
}

func (l *DNSNameUnderscoreInTRD) CheckApplies(c *x509.Certificate) bool {
    return util.IsSubscriberCert(c) && util.DNSNamesExist(c)
}

func (l *DNSNameUnderscoreInTRD) Execute(c *x509.Certificate) *lint.LintResult {
    parsedSANDNSNames := c.GetParsedDNSNames(false)
    for i := range c.GetParsedDNSNames(false) {
        if parsedSANDNSNames[i].ParseError != nil {
            return &lint.LintResult{Status: lint.NA}
        }
        if strings.Contains(parsedSANDNSNames[i].ParsedDomain.TRD, "_") {
            return &lint.LintResult{Status: lint.Warn}
        }
    }
    return &lint.LintResult{Status: lint.Pass}
}
```

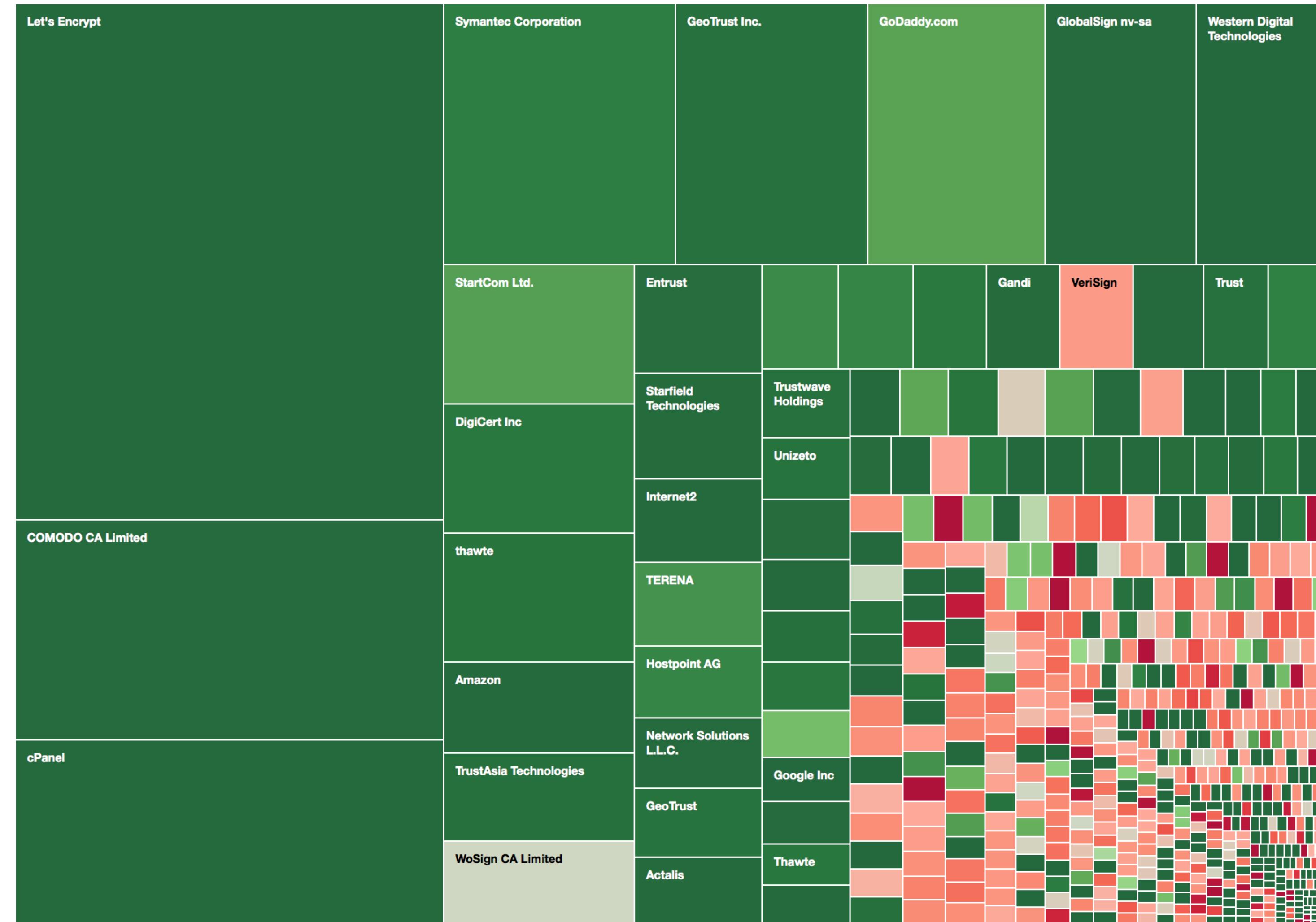
Code

Side note: ZLint remains an active OSS project!



Used by almost every major CA on the planet: Let's Encrypt, Google, DigiCert, etc.

TL;DR Big CAs are pretty good, small CAs are terrible



Some of my favorite takeaways: Nestle is a CA?

Organization	Misissued	Organization	Misissued	Organization	Misissued
Nestle (1)	968 100%	Consorci Catalunya (2)	1,117 58.8%	GoDaddy.com (3)	38,215 2.4%
PSCProcert (1)	39 100%	RHRK (2)	1,171 35.6%	Symantec Corp. [†] (22)	23,053 0.8%
Giesecke and Devrient (1)	18 100%	KPN Corporate BV (2)	1,933 34.5%	StartCom Ltd. [‡] (17)	11,617 2.1%
Unizeto Sp. z o.o. (1)	18 100%	DFN-Verein (5)	1,689 29.8%	WoSign CA Lmtd. [‡] (39)	9,849 5.0%
CertiPath LLC (1)	9 100%	Universitaet Stuttgart (1)	1,830 29.2%	VeriSign [†] (10)	9,835 23.1%
Helsana Gruppe (1)	8 100%	AC Camerfirma S.A. (1)	2,725 25.9%	GeoTrust Inc. [†] (22)	5,694 0.3%
Chunghwa Telecom Co. (1)	7 100%	VeriSign (10)	42,622 23.1%	Comodo Ltd. (30)	3,219 0.1%
TSCP Inc. (1)	5 100%	Trend Micro Inc (1)	6,374 19.8%	DigiCert (43)	2,597 0.1%
Dell Inc. (1)	4 100%	AlphaSSL (1)	3,848 17.2%	Thawte [†] (4)	1,751 0.4%
DigitPA (1)	2 100%	Uni Erlangen Nuernberg (1)	1,115 14.4%	TERENA (9)	1,405 1.7%

(a) Highest Misissuance Rate (b) Highest Misissuance Rate (>1K certificates) (c) Most Misissued Certificates

Sometimes, people are nice

Thank you so much (from Let's Encrypt) Inbox ×

 **Jenessa Petersen** <jpetersen@letsencrypt.org>
to me ▾

Tue, Sep 1, 2020, 3:32PM ⭐ ← ⋮

Hi Deepak,

Thank you so much for your **zlint** help today! We so appreciate it - we truly can't do this work without you and our community!

If it's alright with you, we would like to send you something to say thanks from us. If you are interested, can you let me know a good mailing address to send it to?

Thanks again for your help.

Best,
Jenessa Petersen
Fundraising Specialist at Let's Encrypt

 **Deepak Kumar** <kumarde@cs.stanford.edu>
to Jenessa ▾

Wed, Sep 2, 2020, 10:03AM ⭐ ← ⋮

Hi Jenessa!

Not a problem at all – I for one am very happy that **ZLint** has made it all the way into the issuance pipeline for Let's Encrypt and is proving to be useful :)

I'd be happy to accept! You can send it to:

Sometimes, people are mean

Meta-thoughts on the paper

- Certificate Misissuance is no longer really a problem.... so what did we learn from this paper?
- What about this paper surprised you? What didn't surprise you?
- Why do we think there are so many small CAs? Is there anything to do about them? How do we make our system more resilient against these types of threats?

Next time...

- Moving back down the stack a little to focus on network attacks – DDoS and botnets
- Midpoint check-in due **next Friday!**