Steffen Ullrich
GeNUA mbH

GeNUA

HTTPS

The Days After

about:me

- Steffen Ullrich
- seit 15 Jahren Perl
- seit 10 Jahren bei GeNUA mbH
  Arbeit an Hochsicherheitsfirewalls mit Perl
- seit 2006 Maintainer von IO::Socket::SSL
- Autor von Mail::SPF::Iterator,
  Net::SSLGlue, Net::SIP, Net::INET6Glue,
  Net::PcapWriter...

Überblick über Vortrag

GeNUA

- was ist HTTPS
- existierende Implementationen
- was ist HTTPS nicht
- aktuelle Probleme
- und mögliche Lösungen

Was ist HTTPS

HTTPS - Grundidee

GeNUA

- Alice will bei Bob einkaufen
  - sicherstellen, das Bob wirklich Bob ist
    - Bob zeigt Alice Ausweis
    - Alice checkt, ob der Ausweis für Bob ist
    - Alice checkt, ob der Ausweis aktuell ist
    - Alice vertraut Herausgeber von Ausweis CArl
    - Alice checkt, ob CArl den Ausweis entwertet hat
  - Verbindung verschlüsseln, damit keiner Kontendaten abhört

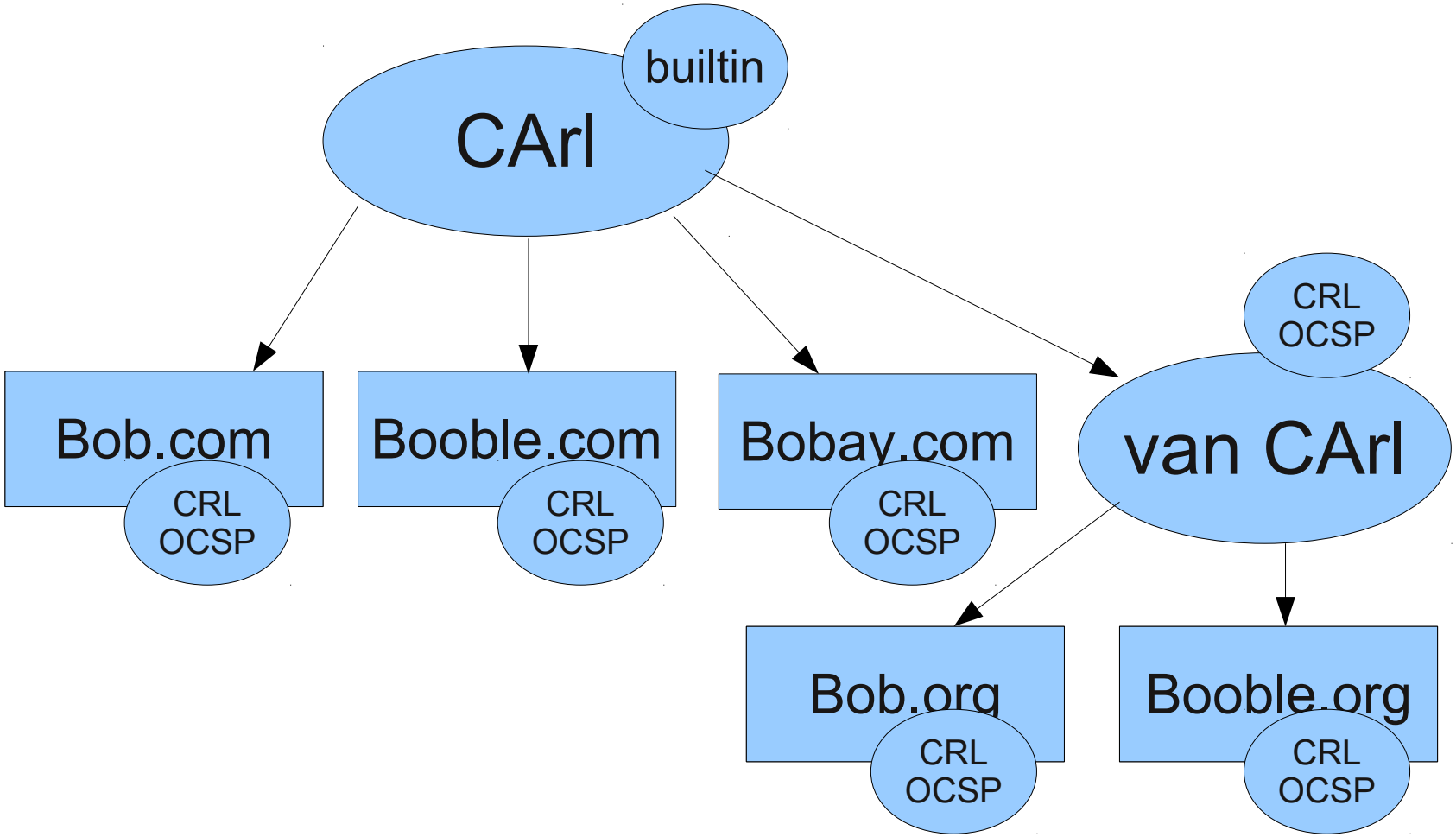
- Bestandteile von HTTPS
  - HTTP
  - ”Secure” - TLS (ehemals SSL)
    - Identifizierung
    - Verschlüsselung

- cryptografische Grundlagen TLS
  - Public-Key Cryptographie: RSA...
  - symetrische Verschlüsselung: DES, RC4, AES...
  - Hash: MD2, MD5, SHA-1...
  - Signierung: DSA...
- Mathematik stabil, sicher nur für bestimmte Zeit

- administrative Grundlagen TLS: PKI
  - Eigentümer
  - Herausgeber: Certificate Agency (CA)
  - Gültigkeitsbeschränkungen
  - Revocation

  ⇒ Zertifikat: X.509
    - Namen: Subject, CN, subjectAltName...
    - Gültigkeitszeitraum
    - Basic Constraints, Key Usage
    - CRL, OCSP
    - signiert von CA

PKI - Theorie


builtin
CArl
CRL
OCSP
Bob.com
CRL
OCSP
Booble.com
CRL
OCSP
Bobay.com
CRL
OCSP
van CArl
Bob.org
CRL
OCSP
Booble.org
CRL
OCSP

- administrative Grundlagen HTTPS
  - Namensprüfung
  - in CA und Client nötig
  - in Theorie
    - Hostnamen lt. DNS
    - RFC2818: CN, subjectAltName, Wildcards
  - und Praxis
    - nationale Gepflogenheiten (*.co.uk, *.de..)
    - IDN Homographs, Typosquatter...

GeNUA

- Alice will via Browser beim Server von Bob einkaufen: https://www.bob.com
  - Server zeigt Browser Zertifikat
  - Check, ob Zertifikat für Server ist - Namensüberprüfung
  - Checkt, ob Zertifikat aktuell ist - Gültigkeitszeitraum
  - Browser vertraut CA von Zertifikat, da builtin
  - Browser checkt, ob CA den Ausweis entwertet hat - Revocation
  - Verbindung verschlüsseln

Implementierungen

TLS/HTTPS - Implementierungen

- OpenSSL (nur TLS, nicht HTTPS)
- Mozilla NSS
- Microsoft SSPI
- Chrome, Opera, Safari, Adobe, JSSE...
- GnuTLS
- diverse Implementationen für embedded Systeme
- u.v.m

HTTPS - Browser

- viele Zertifikate für Root-CA fest im Browser bzw. zentral im OS eingebaut
- CRLs werden nur wenn explizit gewünscht benutzt
- OCSP Anfragen üblich, aber fail soft (außer bei EV-Zertifikaten)

GeNUA

- Net::SSLeay (OpenSSL), IO::Socket::SSL
  - TLS + u.a. RFC2818
  - CRL Check gegen lokale Files, kein Holen basierend auf CRLDistributionPoints
  - kein OCSP
  - LWP>6.0 nutzt IO::Socket::SSL mit Mozilla::CA für strikte Überprüfungen per Default
- div. Schnittstellen zu anderen TLS Implementierungen (NSS...), aber eher unbedeutend

GeNUA

- httplib2
  - OpenSSL
  - Namensprüfung RFC2818 ähnlich (prüft CN auch wenn subjectAltName)
  - CRL/OCSP genauso schlecht wie in Perl
  - eigener CA-Store mit ausgewählten Root-CA
- twisted
  - benutzt pyopenssl (OpenSSL)
  - keine Namensprüfung?
  - CRL/OCSP auch schlecht..

GeNUA

- net::http
  - OpenSSL
  - RFC2818 konforme Namensprüfung
  - CRL/OCSP so schlecht wie bei den anderen
  - auf irgendwo vorhandene CAs angewiesen

HTTPS - Mono(C#)

- kein OpenSSL, eigene Implementation
  - Namensprüfung RFC2818 konform
  - eigener leerer CA-Store
    (kann Mozillas Zertifikate importieren)
  - weder CRL noch OCSP Unterstützung?
    - man mono: ”not implemented yet”

- zuviele Java-Implementationen
- JSSE eigene TLS-Implementation
- javax.net.ssl
  - HttpsURLConnection: lt. Dokumentation RFC2818 konform
  - OCSP/CRL wohl irgendwie mit Security.setProperty möglich (Dokumentation hübsch aber unbrauchbar)

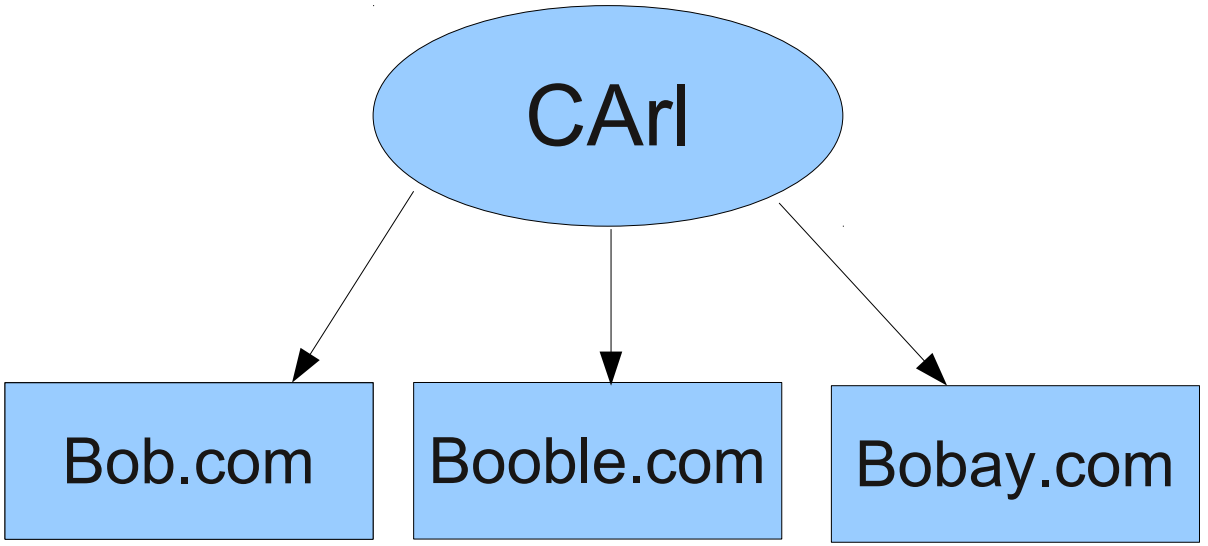
Was ist HTTPS nicht

Was ist HTTPS nicht

- stellt nicht sicher, dass
  - Bob die empfangenen Daten sicher verwahrt
  - Bob keine bösen Absichten hegt
  - Alice Rechner nicht kompromittiert wurde
  - CArl keinen Ausweis für den falschen Bob ausstellt
- d.h. nimmt an, dass
  - die Guten 100% gut und perfekt sind
  - und die Bösen nur wenig Mittel haben

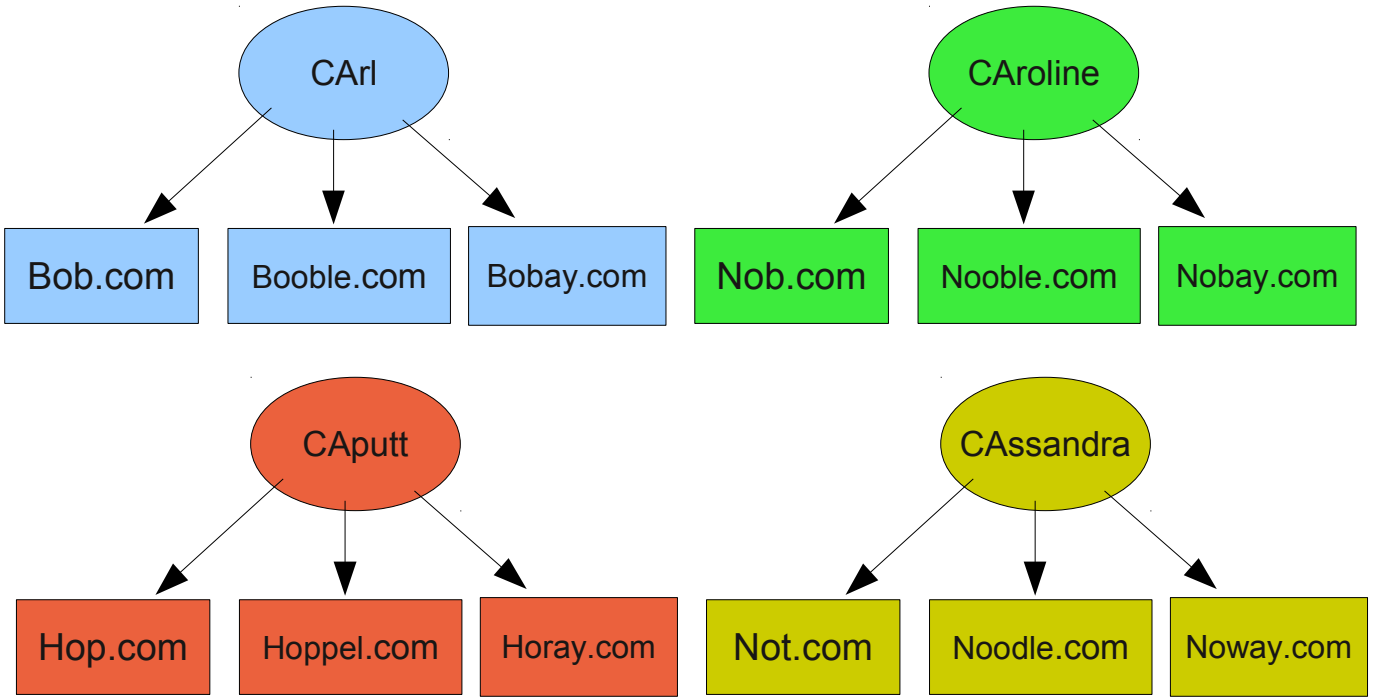
Der GAU

Am Anfang gab es nur CArl



- CArl hatte das Monopol
  - volle Kontrolle
    - wer ein Zertifikat bekommt
    - welche Zertifikate im Umlauf
  - freie Preisgestaltung
  - intransparent
  - wenn man die Sicherheit auf das Notwendigste beschränkt ist der Profit höher
    - merkt ja keiner

Dann kamen noch CAroline, CAssandra, CAputt,...

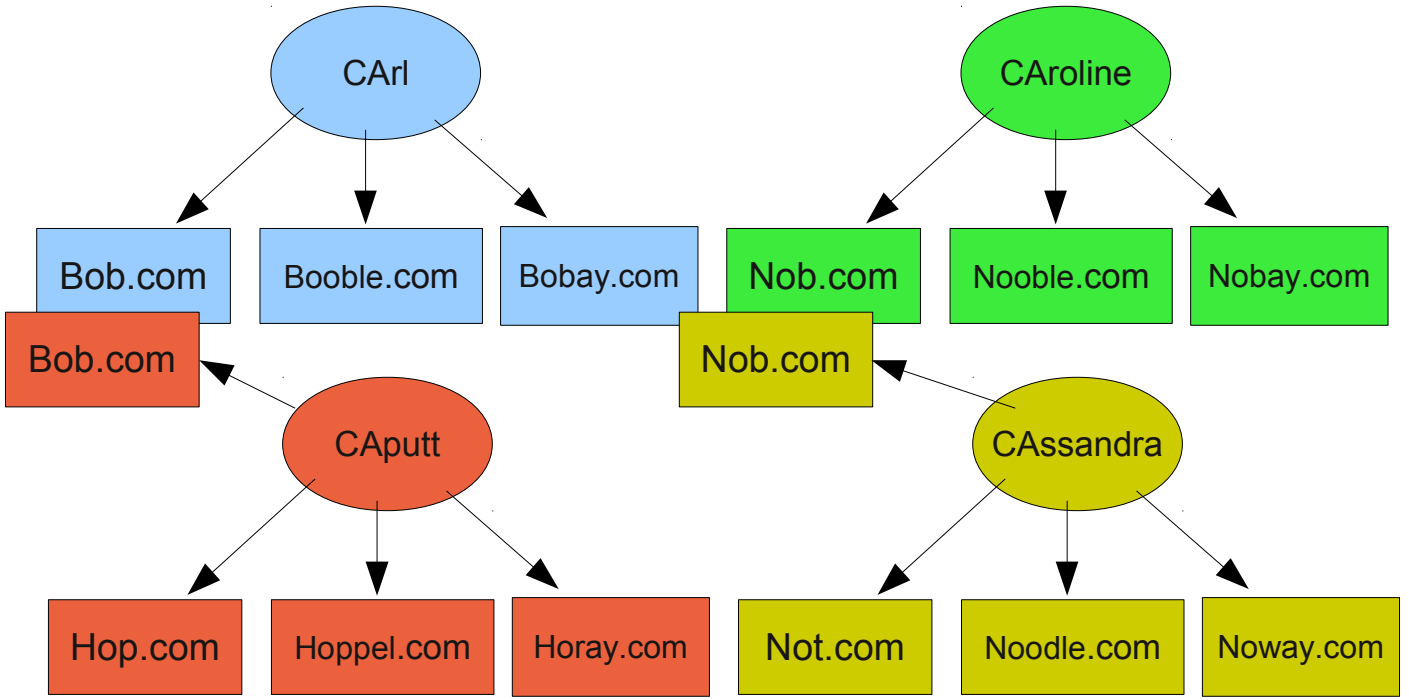


- ein Wettbewerb zwischen den CAs entstand
- die Preise gingen runter
  - Sicherheit aus Kostengründen minimiert
  - und damit auch der Aufwand um Bob von Mallory zu unterscheiden
- fehlende Transparenz kaschiert Probleme
  - nicht klar, welche Zertifikate ausgestellt sind
  - nicht klar, welche zurückgezogen (nur SN)

Willkommen in der Realität

- die Guten waren leider nur 99% gut
- und es gab viel zu viele davon
- die Angriffsfläche war gross genug
- und man konnte sich das einfachste Ziel vornehmen

- 03/2011 Comodo
- 07/2011 DigiNotar
- live.com, google.com, yahoo.com, skype.com, addons.mozilla.com... - aktiv in Iran verwendet



Schadensbegrenzung

- in der Theorie einfach CRL erstellen
- in der Praxis Updates an vielen Stellen:
  - kompletter Browser: Firefox, Chrome, Safari, MSIE, Opera, ???
  - Betriebssystem
  - Java, Acrobat Reader, Mozilla::CA
  - manuelles Löschen aus Monos CA-Store
  - Upgrade SSL-aware Firewalls
  - … bestimmt noch was vergessen...

Überraschung?

proaktive vs. reaktive Sicherheit

- proaktive Sicherheit ist teuer
- aber man sieht nichts für sein Geld (Y2K)
  (und mal ehrlich, wer versteht das schon alles?)
- Risiko muss fühlbar sein, damit man reagiert
- starke Sicherheit politisch umstritten
  - eigene Wirtschaft soll geschützt werden
    aber fremde ausspioniert
  - kriminelle Aktivitäten müssen verfolgbar sein
    für die jeweilige Definition von ”kriminell”

Risiko fühlbar machen...
(unvollständig)

- 2002: MSIE nutzt jedes Zertifikat ohne Basic Constraints CA als Zwischenzertifikat, Verisign liefert die Zertifikate dazu,
  2011: Bug taucht in iOS wieder auf
- 2008: Zertifikat mit MD5 Kollisionen (Problem seit 90er Jahre bekannt) – CAs verlassen das sinkende MD5 Schiff
- CVE-2008-2809: Spoofing via user-trusted subjectAltName (seit 2004 bekannt)
- 2008: Thawte erstellt Zertifikat für ”internen Server” www.live.com, Comodo für mozilla.com

Risiko fühlbar machen...
(unvollständig)

- 2009: null-Prefix ”*\00doxpara.com”, sslsniff
- 2009: OCSP Attacken – soft fail
- 2009: sslstrip – Reaktion HSTS
- 2009: Renegotiation Attacke
- 2010: EFF SSL Observatory zeigt Chaos der CA
- 03/2011: Comodo Hack – Reaktion Certificate-Pinning in Chrome
- 07/2011: DigiNotar Hack – hatten selber keine Ahnung, was erstellt wurde
- 2011: BEAST (Problem seit 2006 oder sogar 2002 bekannt)

GeNUA

The Days After

- Was waren die Probleme nochmal?
  - jeder darf alles
    auch andere mit allen Rechten ausstatten
  - schwächstes Glied bestimmt Stärke
  - unzureichender Revokationsmechanismus
  - fehlende Transparenz
  - keine effektive Bestrafung möglich, wenn groß genug (lock in)
  - reaktive statt proaktive Sicherheit

Fixit! - Jeder darf alles

- Einschränkung, welche CA welche (Toplevel)Domains zertifizieren darf
  - analog zu DNS (evtl. sogar gleiche Betreiber)
  - verständlicher, wem man vertraut
  - Unterstützung in HTTPS-Implementationen erforderlich (evtl. Zusatzinfo in Root-Zertifikaten)
  - führt evtl. zu neuer Monopolstellung
    - erhöht Preis, nicht zwingend Sicherheit
    - evtl. durch regelmäßige Neuausschreibungen verhinderbar

- Verteilung Zertifikate per DNSSec
  - theoretisch man kann nur Zertifikate für eigene Domains verteilen
    - DNS Betreiber bisher nicht für Sicherheit berühmt
    - Länder-Root-Keys in staatlicher Hand
  - CA optional
  - Support für DNSSec in OS und Routern muss verbessert werden
  - Unterstützung in HTTPS-Implementationen nötig

Fixit! -
Jeder darf alles

- CA darf nur eingeschränkte Rechte weitergeben
  - Sub-CA darf nicht die gleichen oder mehr Rechte als CA haben, d.h. darf nur konfigurierte Subdomains signieren
  - muss in HTTPS Implementationen forciert werden

Fixit! -
schwächstes Glied bestimmt Stärke

- mehrere CAs signieren Zertifikat
  - Güte abhängig davon, wer und wie viele signiert haben
  - reicht wenn von ausreichend beteiligten CA revoked

Fixit! - schwächstes Glied bestimmt Stärke

- unabhängige Notare
  - CA optional
  - Perspectives/Convergence
    - schon geringe Durchdringung erhöht Hackschwelle
    - einige glaubwürdige Notare reichen initial
  - certs.googlednstest.com
    - kein DNSSec, Antwort fälschbar

Fixit! - schwächstes Glied bestimmt Stärke

- hartcodiertes Pinning Domain an CA
  - in Chrome als Antwort auf Comodo Hack eingeführt
  - hat DigiNotar Hack entdeckt
- automatisches Pinning Zertifikat bei erster Nutzung
  - wenn geändert, dann meckern
  - FF Erweiterterung Certificate Patrol

Fixit! - Revocation

- aktuelle Situation:
  - riesige CRLs
  - mangelhafter OCSP Support
  - CRL/OCSP URL steht in Zertifikat selber
  - Revocation für Seriennummer, intransparent
  - keine Positivauskunft, dass Zertifikat bekannt, nur Negativauskunft, dass nicht revoked
- OCSP überall implementieren, Möglichkeit für Positivauskunft bei CA bereitstellen

Fixit! - fehlende Transparenz

- CAs müssen alle Zertifikate veröffentlichen
  - suchbar per Name
  - suchbar per SN um CRL zuordnen zu können
- Prozess muß garantieren, das keine Zertifikate heimlich bleiben können

Fixit! -
nur die Kleinen werden bestraft

- wenn Zertifikat nicht mehr fix an eine CA gebunden, können auch die Großen bestraft werden
  - mangelnde Sicherheit wird Wettbewerbsnachteil

Fixit! -
nur die Kleinen werden bestraft

- Möglichkeit einführen, CA-Signaturen nachträglich zu befristen, d.h. nur Zertifikate signiert vor.. werden akzeptiert
  - ausgestellte Zertifikate vor GAU bleiben erhalten, aber für neue muß auch große CA erstmal Hausaufgaben machen
  - CA könnte versuchen zu tricksen (wird aber bemerkt, sofern mehr als Einzelfälle)

Fixit! -
reaktive vs. proaktive Sicherheit

GeNUA

- brauchen mehr Leute wie Moxie Marlinspike
- regelmäßige, unabhängige Audits der CAs
- keine Soft-Fails (OCSP..)
- koordiniertes Vorgehen Browserhersteller, da ohne Browsersupport CA fast wertlos
- vorgeschlagene Ansätze miteinander kombinieren erhöht Robustheit und senkt Angriffsfläche

GeNUA

- wäre wirklich Zeit endlich mal OCSP einzubauen
  - dummerweise OpenSSL Doku nicht vorhanden oder schrecklich
- Verifikation gegen certs.googlednstest.com sollte einfach machbar sein
- Implementation Convergence Concept
  - Protokoll wohl nur implementiert, nicht dokumentiert

GeNUA

- TLS überall gewünscht
- mehr Power nötig: Load Balancing, CDN,... viele viele Server weltweit verteilt mit gleichem Zertifikat
  - wie halte ich den private Key sicher
  - wie stelle ich fest, wenn der Key kompromittiert ist
  - und wie gelangt die Info zum Anwender

In eigener Sache

wir suchen Perl-, OpenBSD-,..
Entwickler und Interessierte an
Forschung.

Fragen?