Network Working Group
Internet-Draft
Intended status: Informational
Expires: April 24, 2014

H. Tschofenig
E. Lear
IAB Security Program
October 21, 2013

Evolving the Web Public Key Infrastructure
draft-tschofenig-iab-webpki-evolution-00.txt

Abstract

The problems with the WebPKI have received the attention by the Internet security community when DigiNotar, a Dutch certificate authority, had a security breach in 2011 and in the same year a Comodo affiliate was compromised. Both cases lead to fraudulent issue of certificates and raise questions regarding the strength of the WebPKI used by so many applications.

Almost 2 years have passed since these incidents and various standardization activities have happened in the meanwhile offering new technical solutions to make the public key infrastructure more resilient.

The important question, however, is which of the technical solutions will get widespread deployment? In this document we compare the different technical solutions in an attempt to engage the impacted stakeholders to trigger deployment actions to improve the status quo. This document does not include any recommendations what techniques to use.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 24, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction

High-profile data breaches and security incidents on the Web are gaining increasing attention from the public, the press, and governments. A few examples may illustrate the problems: DigiNotar, a Dutch certificate authority, had a security breach [DigiNotar] and in the same year a Comodo affiliate was compromised [Comodo]. Both cases lead to fraudulent issue of certificates.

Public Key Infrastructure (PKI) makes use of a trusted third party, the certificate authority (CA), to bind the subject name to a public key. A CA may, however, get compromised despite the best security practices and operational procedures. The main problem, however, is that any CA can issue a certificate for any domain name. One compromised CA is therefore able to impact the security of the entire public key infrastructure. In the case of DigiNotar the attacker was able to issue certificates for Google services even though Google never made use of services from DigiNotar and might not have ever heard of that CA before.

Furthermore, over time browsers and applications increased the number of trust anchors that are shipped pre-installed. Depending on software the number of trust anchors may exceed 600, as reported by the Electronic Frontier Foundation (EFF) in their SSL Observatory study [SSL-Observatory]. While the larger number provides choice for relying parties regarding the CA they can select for obtaining a certificate there is also a downside: with today's WebPKI set-up it is sufficient to compromise a single CA to impact the security for all relying parties. Many users and researchers were surprised about the large number of trust anchors installed in normal operating systems and browsers without having an easy way to adjust that list to their preferences.

To re-state the problem statement: Every CA can issue certificates for any relying party even though that relying party may have never been in a relationship with the issuing CA. (Note that the trust anchor of that CA needs to be provisioned into the trust anchor store.)

These developments have led to a number of protocol design activities for improving the public key infrastructure. In this document we briefly summarize the available technical solutions and include an assessment about who needs to make changes, what type of benefits are provided, and what dependencies exist. The investigated solutions include DANE [RFC6698], Certificate Transparency [RFC6962], Public Key Pinning [I-D.ietf-websec-key-pinning], TACK [I-D.perrin-tls-tack], Perspectives [Perspectives], Sovereign Keys [SovereignKeys], MECAI [MECAI], Convergence [Convergence], and DetecTor [DetecTor].

While there are other challenges with security on the Web, such as user interface problems with certificate warnings, insecure use of cookies, cross-site scripting attacks, injection attacks, etc., this document focuses on improving the public key infrastructure only. It is also worth reminding ourselves that the Web public key infrastructure is not only used for Web applications but also for a range of other applications, including smart phone apps. Furthermore, other public key infrastructures that operate under a different regime with different policies may suffer from similar problems. Consequently, the solution techniques discussed in this document are also useful for these other PKI deployments.

The main purpose of this document is to provide an overview of the technical solutions. This description will help us to develop a roadmap for the deployment of the best solutions to improve the overall security of the public key infrastructure.

Final note: There are also process solutions, such as stricter audits of CAs with the aim to improve operational practices, and these are not described in this document. These measures will be useful in addition to technical solutions but alone they will, however, not address the underlying problem.

2. Terminology

This document uses the following terms from from RFC 3280 [RFC3280]:

end entity: user of PKI certificates and/or end user system that is the subject of a certificate.

CA: certification authority

This document also re-uses the term "Leap of faith" from RFC 5386 [RFC5386]:

> "Leap of faith is the term generally used when a user accepts the assertion that a given key identifies a peer on the first

communication (despite a lack of strong evidence for that assertion), and then remembers this association for future communications."

This security property has become fairly popular with use of Secure Shell [RFC4251], which made use of the leap of faith property.

RFC 6973 [RFC6973] provides a definition of the term 'relying party':

"The relying party is an entity that relies on assertions of individuals' identities from identity providers in order to provide services to individuals. In effect, the relying party delegates aspects of identity management to the identity provider(s). Such delegation requires protocol exchanges, trust, and a common understanding of semantics of information exchanged between the relying party and the identity provider."

In the context of this document the relying party is a TLS server, for example, used to protect the communication of a Web server. Although a lot of focus is on the Web there are other non-HTTP-based services that are included in the definition and may benefits from improvements discussed in this document.

The terms 'trust anchor' and 'trust anchor store' are defined in [RFC6024]:

"A trust anchor represents an authoritative entity via a public key and associated data. The public key is used to verify digital signatures, and the associated data is used to constrain the types of information for which the trust anchor is authoritative."

"A trust anchor store is a set of one or more trust anchors stored in a device. A device may have more than one trust anchor store, each of which may be used by one or more applications."

3. Technical Solutions

3.1. Public Key Pinning for HTTP

[I-D.ietf-websec-key-pinning] describes a solution for instructing user agents (UAs) to remember ("pin") certificates (end entity certificates or CA certs) for a given period of time. During that time, UAs will require that the TLS server presents a certificate chain including at least one Subject Public Key Info structure whose fingerprint matches one of the pinned fingerprints for that host.

While the specification provides a number of instructions for the Website operator to indicate towards the UA the basic operation is rather simple and assumes a leap-of-faith policy. To deal with the change of certificates or other failure scenarios the concept of a backup pin is utilized. A Backup Pin is a fingerprint for the public key of a secondary, not-yet-deployed key pair. The operator keeps the backup key pair offline, and sets a pin for it in the Public-Key-Pins header. Then, in case the operator loses control of their primary private key, they can deploy the backup key pair. An interesting feature of the specification is to report pin validation failure.

When a pin validation failure occurs the expectation is that the user is notified about the inconsistency (with optionally reporting taking place in the background).

This document is the product of the IETF Web Security working group.

3.2. Trust Assertions for Certificate Keys (TACK)

Similarly to the key pinning solution described in Section 3.1 TACK [I-D.perrin-tls-tack] also aims to enables a TLS server to support "pinning" to a self-chosen signing key. There are, however, a number of substantial differences in the design despite the similarity of the name.

TLS server operators create a so-called "TACK signing key" (TSK) and sign their own keys used by TLS servers. A TACK pin then associates a hostname, a TSK, and various parameters (including pin creating time, and lifetime of the pin). A TLS server operator may change a key for a server at any point in time since the TSK will be unchanged. The existing public key infrastructure is replaced by a form of self-signed certificates. Clients store the TACK pins in their pin stores, which they may have obtained from different sources. Although the focus of the specification is to obtain the TACK pins via a TLS extension from the server directly a mechanism to obtain these TACK pins from a third party infrastructure is envisioned, although outside the scope of the specification. When TACK pins are obtained from the TLS server directly they follow a leap-of-faith approach; a third party distribution mechanism may have additional security properties.

For incremental deployment the TLS client uses the extension mechanism of TLS to indicate support for the TACK extension by including a new TLS extension type in the ClientHello message. A TLS server that does not support TACK will reply with an ordinary certificate. In case the TLS server supports the extension it replies with the newly defined tack structure, which contains the TACK pin for that server.

This specification is an individual submission to the IETF.

[Editor's Note: The document claims that the proposal also works with certificates. However, details are missing to describe how the TLS server key is signed with the TSK and then used by a regular TLS library.]

3.3. Perspectives

Perspectives [Perspectives] aims to utilize notaries (i.e., public servers) that monitor and record the history of public keys used by sites. While the description focuses on the use of raw public keys (in the style of SSH) the same concept also works with certificates.

The basic approach is simple: When a TLS client starts to interact with a TLS server it is presented with a key/certificate that it had not seen before. To verify that the key/certificate is the same as observed by other vantage points in the network it contacts one or multiple notary servers. These notary servers provide key/certificate information they have obtained about the specific website earlier.

To improve the leap of faith security by clients the notary services adds security value since they may have obtained the key/certificate from the website in the past already and from a different vantage points in terms of the path used to talk to the server. This helps when attacks are either temporary and or when the man-in-the-middle attacker is located somewhere along the path between the client and the server but closer to the client. The use of multiple notaries also helps to detect malicious notaries.

With clients caching information about the keys/certificates of sites visited earlier and the information obtained from notaries there is no additional protocol overhead. In this respect the solution works similar to key pinning. The additional communication overhead for the client only occurs at the time when the client talks to a server for the first time or when the cached information expires.

Similar to other notary services there is the question about how they obtain information about the available TLS servers. For popular

services obtaining the keys/certificates a list of sites is assumed to pre-configured and queried periodically but for the long tail of small websites the suggested approach query these sites the first time the client wants to connect to them.

The proposal is documented in form of an academic research paper, see [Perspectives], but no technical specification is available.

3.4. Convergence

Convergence [Convergence] is a proposal by Moxie Marlinspike that makes two improvements to Perspectives, namely

Reducing Notary Lag: Perspective required TLS clients to interact with notaries to check whether the certificate obtained through TLS matches the information seen by one or many notaries. Notaries then had to initiate an interaction with the TLS servers to obtain information about what certificates they see. Convergence reduces this interaction by utilizing caching of certificates at the notaries. By doing this, however, they also introduce a delay between the time a new certificate is put in operation at a TLS server and when the notaries get to learn about it.

Increased Privacy Protection: First, clients cache certificates so that they do not need to contact notaries every time they contact a Web site. Second, clients use a concept called 'notary bouncing' whereby they pick a notary randomly out of their pool of trusted notaries and use it as a proxy to talk to other notaries. Thereby, the notary that receives the query will only see the IP address of another notary who forwarded the query rather than the IP address of the client.

The client can decide how many notaries are consulted to obtain certificate from a given TLS servers. As a main advantage the author claims that there is no impact on TLS servers deployments, except in rare situations where multiple certificates are used by a single site in combination with a load balancer.

Notaries are designed to be extensible by supporting different mechanisms how they obtain certificates. Currently, Convergence uses the technique proposed by Perspectives to probe a TLS server.

The documentation of Convergence only exists in form of a presentation by Moxie Marlinspike given at the BlackHat USA 2011 conference [ConvergenceTalk].

3.5. Sovereign Keys

Sovereign Keys [SovereignKeys] is a proposal by the Electronic Frontier Foundation (EFF) suggesting to introduce two new concepts to deal with attacks against the public key infrastructure.

Sovereign key: Domain owners create a new key pair, the sovereign key, and use it to sign their operational certificates / the public keys.

Timeline servers: Append-only timeline servers, as new entities, are introduced that stores mappings between domain names to sovereign keys. To claim a key for a domain name requires evidence of control in the DNS either via a CA-signed certificate or via a key published in the DNS (as provided by DANE).

Each timeline server possesses a unique private/public key pair and these keys are assumed to be shipped with client software or TLS libraries to ensure that clients can verify the authenticity of timeline entries. The timeline servers record the history of claims to sovereign keys.

TLS clients query timeline servers for entries that belong to a certain domain and verify that the end-entity certificate has been cross-signed by the sovereign key. If the verification fails then the connection attempt is refused.

A high-level description can be found at [SovereignKeys] and a more detailed technical specification is available at [SovereignKeys-Spec].

3.6. Mutually Endorsing CA Infrastructure (MECAI)

MECAI [MECAI] builds conceptually on top of the Perspective proposal. Perspectives introduces notaries, as new entities in the public key infrastructure, and MECAI takes the position that this function can be taken by existing CAs. With this new role they would turn into Voucher Authorities (VAs), who issue vouchers that confirm what they observe. A voucher is a signature computed over a number of fields including the hash of the server certificate, the certificate chain, the IP address of the server, revocation status information, etc. Of course, a voucher would be created by a CA other than the one that created the original certificate.

A client would therefore perform the following steps: it connects to a server via TLS and the server provides the certificate. Then, the client needs to obtain one or multiple vouchers for the server certificate. This can happen either inband within the TLS handshake

when talking to the server, similarly to how OCSP stapling works, or via a separate protocol exchange. The former approach is less expensive in terms of communication costs for the client. In any case, a voucher request protocol is needed to let entities (like TLS servers) talk to VAs to obtain a voucher.

A client or a server can detect misissuance by matching the information in the vouchers with the certificate.

Only a high-level description is available via [MECAI] but no detailed technical specification.

3.7. DNS-Based Authentication of Named Entities (DANE)

DANE [RFC6698] offers the option to use the DNS infrastructure to store certificates. DANE is envisioned as a preferable basis for binding public keys to DNS names, because the entities that vouch for the binding of public key data to DNS names are the same entities responsible for managing the DNS names in question.

Distributing certificates via the DNS does, however, require DNSSEC. With the help of DNSSEC [RFC4033][RFC4034][RFC4035] this offers an opportunity to eliminate off-line processes for validation of the subject name, which today often requires sending a mail to the administrator of that domain. This relationship can be easily demonstrated by having the zone administrator for the subject domain post the public key in the DNS and digitally sign the resulting zone.

A high-level description about the different options offered by DANE can be found in [IETF-Journal-DANE] and the authoritative version can be found in RFC 6698 [RFC6698].

3.8. Certificate Transparency

RFC 6962 [RFC6962] specifies Certificate Transparency, a protocol for publicly logging the existence of certificates as they are issued or observed, in a manner that allows anyone to audit certificate authority (CA) activity and notice the issuance of suspect certificates as well as to audit the certificate logs themselves. The intent is that eventually clients would refuse to honor certificates that do not appear in a log, effectively forcing CAs to add all issued certificates to the logs.

The publicly auditable, append-only logs of all issued certificates does not prevent misissue but allows interested parties to detect misissuance.

While various projects, including the EFF with their SSL Observatory [SSL-Observatory] and Crossbear [Crossbear], have scanned the Internet to collect all certificates of publically accessible TLS servers the cooperation from all CAs or from certificate owners is required to make Certificate Transparency proposal successful. The reasons are two-fold: IPv6 makes scanning the address range of the entire Internet much more difficult and the increasing deployment of the TLS Server Name Indication [RFC6066] prevents it from obtaining all available difficult.

The expected operation is as follows: CAs or certificate owners contact logs and upload certificates, as they issue them. In response, they receive a Signed Certificate Timestamp (SCT). The SCT is the log's promise to incorporate the certificate in the Merkle Tree, which is the data structure used by the log, within a fixed amount of time. Everyone can check the log for consistency. Particularly website operators will have an interest to regularly check the logs for misissuance of certificates. TLS clients on the other hand are not expected to directly communicate with logs to avoid the communication overhead. Instead, the TLS servers provides the SCT along with the certificate within the TLS handshake. TLS clients reject certificates that do not have a valid SCT for the end entity certificate. Since there is ideally more than one log TLS servers need to provide SCTs from multiple logs to the client.

This document has gone through a public review process, and has been approved by the Internet Engineering Steering Group and published an experimental RFC.

3.9. DetecTor

DetecTor [DetecTor] extends the idea of MECAI and Perspectives by utilizing the Tor onion routing infrastructure [Tor] in order to see connect to sites via different paths through the network. The Tor infrastructure thereby replaces the need to have dedicated notary servers, who connect to sites to obtain certificates from a different vantage point. The server certificate obtained via one or multiple Tor connections is then compared with the certificate that was obtained via the direct TLS connection between the client and the site (i.e., without using Tor). This offers capabilities for the client to detect whether there was an adversary along the path but close to the client.

Unlike other proposals, the suggestion is made to provide no information to the user once a failure has been detected. Instead, the connection attempt will be rejected and no recourse is possible. Like other proposals information about the observed certificates may be cached by the client to lower the initial set-up delay.

A high-level description can be found at [DetecTor] but no detailed technical specification is available.

4. Analysis

This version of the document re-uses the analysis criteria proposed by Eric Rescorla [Rescorla].

4.1. Public Key Pinning for HTTP

Changes Needed: Browsers, Servers

Benefits: Prevention and Detection (when reporting is used)

Dependencies: None

Incremental Deployment: Newly added server can make use of the technology when browsers have been updated. Works with existing PKI infrastructure.

Risks: Provides a leap of faith concept. Self-DoS if pins are configured incorrectly.

4.2. Trust Assertions for Certificate Keys (TACK)

Changes Needed: Browsers, Servers

Benefits: Prevention

Dependencies: Requires server operators to create and manage new public / private key pair (TSK)

Incremental Deployment: Newly added server can make use of the technology when browsers have been updated. Does not seem to work with existing PKI infrastructure.

Risks: Provides a leap of faith concept.

4.3. Perspectives

Changes Needed: Third party infrastructure (notaries), Clients

Benefits: Prevention

Dependencies: Requires notaries to be deployed.

Incremental Deployment: Once notaries are available and client software the solution works with every server.

Risks: Increased communication overhead for contacting notaries and for letting notaries check servers. Potential problems when load balancers are deployed at the server infrastructure.

4.4. Convergence

Changes Needed: Third party infrastructure (notaries), Clients

Benefits: Prevention

Dependencies: Requires notaries to be deployed.

Incremental Deployment: Once notaries are available and client software the solution works with every server.

Risks: Increased communication overhead for contacting notaries and for letting notaries check servers (although more extensive caching is utilized than with Perspectives). The notary bounding concept may introduce additional latency. Potential problems when load balancers are deployed at the server infrastructure. With caching a certain lag may be introduced between the time when a new server certificate is configured and the time when the notaries notice about its existence.

4.5. Sovereign Keys

Changes Needed: Third party infrastructure (timeline), Clients, Servers

Benefits: Prevention

Dependencies: Requires server operators to create and manage new public / private key pair (sovereign key). Requires third party infrastructure (timeline servers).

Incremental Deployment: New server operators will receive benefits once timeline servers are deployed, and updates to the client software has been made.

Risks: Increased communication overhead for contacting timeline servers.

4.6. Mutually Endorsing CA Infrastructure (MECAI)

Changes Needed: CAs (who operate the Voucher Authorities (VAs)), Servers, Clients

Benefits: Prevention

Dependencies: Requires CAs to operate VAs

Incremental Deployment: Requires at least two CAs to support MECAI before TLS servers can start to offer vouchers in the TLS handshake to clients for verification.

Risks: A VA has to obtain a certificate and verify it before it can issue a voucher. A client may request vouchers from a number of VAs to have enough confidence.

4.7. DNS-Based Authentication of Named Entities (DANE)

Changes Needed: Clients, Server's DNS

Benefits: Prevention

Dependencies: DNSSEC deployment at clients, and intermediaries.

Incremental Deployment: A new server can add support for DANE only it the DNS allows TLSA records to be added and secured via DNSSEC. Then, clients need to have software support in the browsers for verifying the DNSSEC protected TLSA record.

Risks: Self-DoS with incorrect TLSA records, false positives with broken intermediaries, lack of DNSSEC deployment or failure with DNSSEC validation.

4.8. Certificate Transparency

Changes Needed: Third party infrastructure (notaries), CA, Clients, Servers

Benefits: Detection

Dependencies: Requires notaries and all CAs to participate

Incremental Deployment: CAs and servers who want to deploy the infrastructure can start deployment (after notaries have become available).

Risks: Non-participating CAs are not monitored and attacks against those cannot be detected.

4.9. DetecTor

Changes Needed: Clients

Benefits: Prevention

Dependencies: Depends on Tor infrastructure

Incremental Deployment: With client-side only changes all servers can be verified.

Risks: Setup delay and sites that utilize load balancers. Relies on leap-of-faith security. Certificate changes on the server-side might cause mismatches with cached information.

4.10. Limitations

A common problem of all proposals that aim to prevent attacks lies in the user interface design when a failure occurs and end users are informed about the problem. In many cases, the failure may not necessarily be caused by real attacks but rather by the use of captive portals or server-side configuration problems (like warnings caused by expired certificates today). User interface studies, such as [SE09], [SR07], and [BO09], have shown that end users are typically not in the best position to make judgements about these security warning dialogs. Furthermore, proposals that make use of out-of-band communication interactions may face problems with firewalled networks and the additional incurred delay. Claims have been made that this is a problem with the use of OCSP today [OCSP-Performance], which has been the motivation for developing and standardizing OCSP stapling and multiple OCSP stapling.

5. Security Considerations

This entire document is about security.

6. Privacy Considerations

The main privacy threat is correlation. Correlation is the combination of various pieces of information related to an individual or that obtain that characteristic when combined. In this specific case there is the risk that newly introduced entities obtain information about the history of service usage. For example, a notary that is contacted each time a user visits a new website can easily be seen as problematic from a privacy point of view.

7. IANA Considerations

This document does not require actions by IANA.

8. Acknowledgements

We would like to thank all participants of the NIST workshop on "Improving Trust in the Online Marketplace", April 10-11 2013, for sharing their views with the community. We would also like to thank the authors of various solution proposals for their work.

9. References

9.1. Normative References

[ConvergenceTalk]
Marlinspike, M., "BlackHat USA 2011: SSL And The Future Of Authenticity", URL: http://www.youtube.com/watch?v=Z7Wl2FW2TcA, 2013.

[Convergence]
Marlinspike, M., "Convergence", URL: http://convergence.io, 2013.

[DetecTor]
Engert, K., "DetecTor", URL: http://detector.io, Sep 2013.

[I-D.ietf-websec-key-pinning]
Evans, C., Palmer, C., and R. Sleevi, "Public Key Pinning Extension for HTTP", draft-ietf-websec-key-pinning-08 (work in progress), July 2013.

[I-D.perrin-tls-tack]
Marlinspike, M., "Trust Assertions for Certificate Keys", draft-perrin-tls-tack-02 (work in progress), January 2013.

[MECAI] Engert, K., "MECAI - Mutually Endorsing CA Infrastructure", URL: https://kuix.de/mecai/, Feb 2012.

[Perspectives]
Wendlandt, D., Andersen, D., and A. Perrig, "Perspectives: Improving SSH-style Host Authentication with Multi-Path Probing", URL: http://www.cs.cmu.edu/~dga/papers/perspectives-usenix2008/, 2008.

[RFC3280] Housley, R., Polk, W., Ford, W., and D. Solo, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002.

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, March 2005.

[RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, March 2005.

[RFC4035] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Protocol Modifications for the DNS Security Extensions", RFC 4035, March 2005.

[RFC6066] Eastlake, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, January 2011.

[RFC6698] Hoffman, P. and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA", RFC 6698, August 2012.

[RFC6962] Laurie, B., Langley, A., and E. Kasper, "Certificate Transparency", RFC 6962, June 2013.

[SovereignKeys-Spec]
Eckersley, P., "Sovereign Key Cryptography for Internet Domains", URL: https://git.eff.org/?p=sovereign-keys.git;a=blob;f=sovereign-key-design.txt;hb=master, Oct 2013.

[SovereignKeys]
EFF, "The Sovereign Keys Project", URL: https://www.eff.org/sovereign-keys, Oct 2013.

9.2. Informative References

[BO09] Biddle, R., van Oorschot, P., Patrick, A., Sobey, J., and T. Whalen, "Browser Interfaces and Extended Validation SSL Certificates: An Empirical Study, Proceedings of the 2009 ACM workshop on Cloud computing security", URL: http://www.andrewpatrick.ca/cv/Biddle-CCSW-2009.pdf, 2009.

[Comodo] Hallam-Baker, P., "The Recent RA Compromise", URL: http://blogs.comodo.com/it-security/data-security/the-recent-ra-compromise/, Mar 2011.

[Crossbear]
Technical University Munich, "Crossbear", URL: https://pki.net.in.tum.de/, Oct 2013.

[DigiNotar]
Arthur, C., "DigiNotar SSL certificate hack amounts to cyberwar, says expert", URL: http://www.guardian.co.uk/technology/2011/sep/05/diginotar-certificate-hack-cyberwar, Sep 2011.

[IETF-Journal-DANE]
Barnes, R., "DANE: Taking TLS Authentication to the Next Level Using DNSSEC, IETF Journal", URL: http://www.internetsociety.org/articles/dane-taking-tls-authentication-next-level-using-dnssec, Oct 2011.

[OCSP-Performance]
Netcraft, "Certificate revocation and the performance of OCSP", URL: http://news.netcraft.com/archives/2013/04/16/certificate-revocation-and-the-performance-of-ocsp.html, Apr 2013.

[RFC4251] Ylonen, T. and C. Lonvick, "The Secure Shell (SSH) Protocol Architecture", RFC 4251, January 2006.

[RFC5386] Williams, N. and M. Richardson, "Better-Than-Nothing Security: An Unauthenticated Mode of IPsec", RFC 5386, November 2008.

[RFC6024] Reddy, R. and C. Wallace, "Trust Anchor Management Requirements", RFC 6024, October 2010.

[RFC6973] Cooper, A., Tschofenig, H., Aboba, B., Peterson, J., Morris, J., Hansen, M., and R. Smith, "Privacy Considerations for Internet Protocols", RFC 6973, July 2013.

[Rescorla]
Rescorla, E., "Deployment Models for Backup Certificate Systems, NIST Workshop on Improving Trust in the Online Marketplace", URL: http://csrc.nist.gov/groups/ST/ca-workshop-2013/presentations/Rescorla_ca-workshop2013.pdf, Apr 2013.

[SE09] Sunshine, J., Egelman, S., Almuhimedi, H., Atri, N., and L. Cranor, "Crying Wolf: An Empirical Study of SSL Warning Effectiveness, 18th USENIX Security Symposium", URL: http://lorrie.cranor.org/pubs/sslwarnings.pdf, Aug 2009.

[SR07] Schechter, S., Dhamija, R., Ozment, A., and I. Fischer, "The Emperor's New Security Indicators: An evaluation of website authentication and the effect of role playing on

usability studies, The 2007 IEEE Symposium on Security and Privacy", URL: http://www.usablesecurity.org/emperor/, May 2007.

[SSL-Observatory]
EFF, "The EFF SSL Observatory", URL: https://www.eff.org/observatory, Oct 2013.

[Tor] The Tor Project, "Tor - Anonymity Online", URL: https://www.torproject.org/, Oct 2013.

Authors' Addresses

Hannes Tschofenig
IAB Security Program

EMail: Hannes.Tschofenig@gmx.net

Eliot Lear
IAB Security Program

EMail: elear@cisco.com