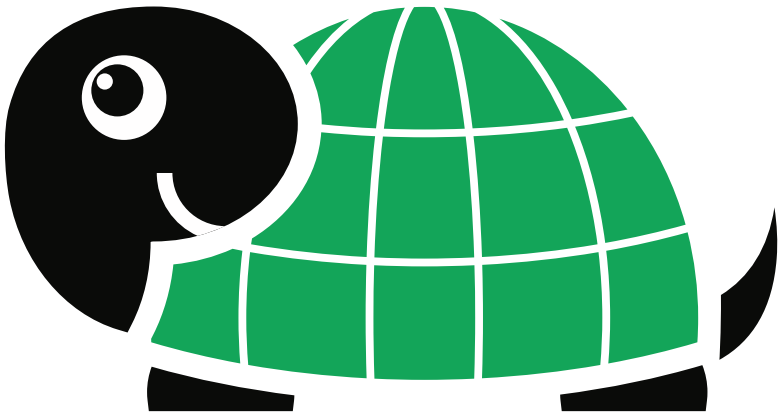
DNSNMC + okTurtles

Easy to use, state of the art security, for existing online communications

Author: Greg Slepak



Version 1.0 http://okturtles.com December 6, 2013

1 Abstract

In this paper, we'll briefly describe some of the shortcomings of today's Internet security (SSL/TLS, HTTPS, and Certificate Authorities), and why these systems do not provide the security that they claim to provide. We'll focus on the importance that authentication plays in securing communication systems, why a system that fails to provide strong authentication does not provide *meaningful security*, and how these problems can be addressed. We then introduce two projects that address these issues. The first project, DNSNMC, deprecates today's insecure and fraudulent[1] public key infrastructure (PKI) by gracefully transitioning DNS from its hierarchical design, to one that is based on a globally distributed, peer-to-peer network that successfully "squares Zooko's triangle".[1] We then use the strong authentication provided by DNSNMC to introduce okTurtles: a protocol and browser extension that protects the content of almost all online text-based communication from a variety of threats (such as MITM), and even private key compromise.

KEYWORDS: Namecoin, DNS, SSL/TLS, security, authentication, X.509, HTTPS, P2P, MITM, DNSNMC, okTurtles, OTR, Certificate Authorities, PKI, Zooko's Triangle, plausible deniability

2 Motivation

Thanks to the whistleblowers, it may very well be common knowledge that many governments around the world—with the United States in particular—record virtually all electronic communication without obtaining a warrant, or permission from the individuals and groups they are recording. In the United States, this surveillance was found to violate the nation's highest law (the Constitution).[2] For reasons unknown to this author, this practice continues to this day, unabated.

---

[1] Companies that sell SSL certificates usually claim that their certificates provide customers with "security." Customers are led to believe that these certificates protect browser-server communication from eavesdropping and tampering. As elaborated in this paper, this simply isn't true today.

3 Technologies enabling unconstitutional surveillance

Today's surveillance is made possible because most protocols that facilitate online communication do not provide all of the following properties:

1. End-to-end encryption
2. Secure authentication
3. Perfect Forward Secrecy (PFS)
4. Plausible deniability (only necessary for communication between individuals)

We use the term "meaningful security" to refer to the security provided by protocols that employ all of these features for communication between individuals, and the first three features for protocols that facilitate communication between individuals and non-sentient entities (such as static content served by HTTP servers, etc.). The security that such protocols provide is *meaningful* because it protects users from all of the known life-threatening and *life-damaging* threats they may encounter online. Any entity that sits between (or next to) the user and the endpoint they're communicating with represents a potential threat. The list of threatening actors includes institutions of all sorts (businesses, universities, etc.), governments, internet service providers, malevolent network administrators, and random hackers.

Governments pose the most significant threat to users because of their fantastic surveillance capabilities, virtually unlimited resources, and their ability to operate above the law or modify it to suit their needs. As an illustrative example, it was revealed that the NSA appears to be storing all of the information that they are technically capable of getting their hands on.[3][4][5][6] The NSA gives encrypted traffic special attention, storing it for a longer period of time.[7] The existence of such pervasive data retention means that it is no longer sufficient for online communications protocols to simply encrypt data. Communication protocols that do no provide *Perfect Forward Secrecy* (PFS) can no longer be said to provide any meaningful security to their users. PFS employs the use of ephemeral keys to encrypt data that is transmitted between endpoints. This renders the stored encrypted communication worthless in the face of a compromised long-term private key.[8]

Plausible deniability, in the context of computer security, represents feature(s) that give users a "legitimate out", or a way to deny responsibility or ownership of a piece of encrypted data.[9] The existence of repressive governments that employ physical force to threaten the livelihoods of their citizens means that communication protocols must address the threat of coercion. Plausible deniability feature(s) protect users from such threats. Therefore, any protocol that facilitates communication between individuals must provide plausible deniability. Protocols that do not provide such protections open their users to the potential threat of death or imprisonment, and therefore cannot be said to provide meaningful security.

3.1 HTTPS, X.509 & Certificate Authorities

Today, HTTPS is the primary means by which the connection between a user and a website is secured. HTTPS does not provide meaningful security (as has been defined in this paper). HTTPS relies on several underlying technologies for its meaningless security: SSL/TLS and X.509.

According to T. Dierks et. al, "Transport Layer Security (TLS) and its predecessor, Secure Sockets Layer (SSL), are cryptographic protocols which are designed to provide communication security over the Internet."[10][11] The latest version of TLS provides encryption and PFS, but it does not provide secure authentication or plausible deniability, and therefore does not provide meaningful security. For authentication, TLS relies on *certificates* to authenticate end-points as described by X.509 PKI.[10][12] Certificates are electronic documents that can contain various information (a public key and information about some entity), and employ cryptographic signatures that prove their authenticity.

When a user connects to a website over HTTPS, they are given the website's certificate. The user's web browser has no way to tell whether the certificate is legitimate. It is possible that the certificate was replaced with a fraudulent one enroute to the browser via a Man-in-the-Middle (MITM) attack.[13]

To verify the legitimacy of the certificate, web browsers rely on "root certificates" that belong to *Certificate Authorities* (CAs). These certificates are shipped with the browser

itself, and therefore can (supposedly) be trusted.[2] These "root certs" are used to verify the certificates provided by HTTPS servers. Server operators must purchase a certificate that is signed by one of these CAs. If *any one* of the 600+ root certs the browsers comes with declares the website's certificate legitimate, the web browser will actually declare to the user (via graphical elements such as those shown in Figure 1) that the connection to the website is secure, and that no MITM is taking place.

Unfortunately, this method of authentication is broken.[14][15] Because browsers will trust any one of the 600+ CAs, it is only necessary to compromise a single one (whether it be via hacking, a court order, or physical intimidation) to compromise the security of HTTPS worldwide, for all websites.

This is not a theoretical problem. It appears to happen routinely, and there are many documented cases where fraudulent certificates have been used to MITM internet traffic, often to large companies like Google.[14][20][21]

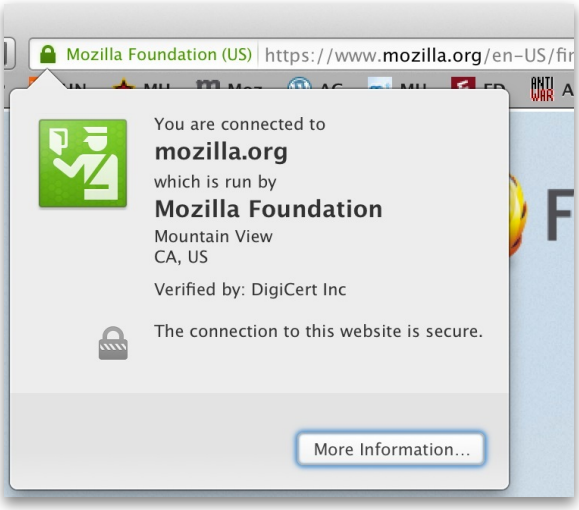


Figure 1. Firefox believes that "The connection to this website is secure."[3]
Unfortunately, so do most users.

[2] Or so the theory goes. Few consider the possibility that the browser itself was compromised enroute to the user. Even if the browser remained untouched, glaring security holes still exist because of CAs.

[3] The sad part is, Mozilla (and others) pay hundreds of dollars for their certificates. Every year! :-O

4 Existing attempts to fix this problem

This document would not be complete if it failed to mention some of the existing proposals to address these problems. The table below mentions a few, along with their respective shortcomings:

Proposal Summary	Problem(s) with the proposal
Google's "Certificate Transparency"[22] wants CAs to note all of the certs that they issue into a log somewhere protected in someway and verified via some mechanism. The original proposal is almost this vague. Website owners are then asked to monitor these logs to see if their clients were hacked, in addition to continuing to pay the CAs money for the worthless certificates they provide.	The proposal summary seems to do an adequate job of summarizing the problems with it. This is not surprising. Companies that require access to your information for their survival will never provide meaningful leadership on security.
DNSSEC[23] suggests a complicated mechanism to essentially re-create many of the same problems with X509 and CAs in DNS itself, by providing a chain of trust to untrustworthy entities. Its intended goal is to secure DNS and thereby assure clients that when they ask for "**apple.com**", they are actually visiting **apple.com**.	This is another proposal that seems like a great argument against itself. It does not appear to protect against MITM.[25] Unnecessary complexity breeds security problems. DNSSEC appears to be a terrible idea for various other reasons.[26]
In their words, **Convergence** "is a secure replacement for the Certificate Authority System. Rather than employing a traditionally hard-coded list of immutable CAs, Convergence allows you to configure a dynamic set of Notaries which use network perspective to validate your communication."[24] In our words: Convergence is similar to having a known_hosts ssh key file for your browser, and comparing it against your friend's file. Not a terrible idea.	The website claims it's simple to use, but we have to disagree because users are asked to manage a list of notaries. It depends on group consensus, but the group might not be very bright. What happens then? It does not provide MITM protection on first-visit. All of the notary info appears to be stored locally to the computer, or even the browser. Rather inconvenient for just about everyone.

4 DNSNMC: Your connection to the Namecoin blockchain

DNSNMC fixes the authentication problems previously described, and it addresses all of the problems that with the previously mentioned proposals. It does this first by combining DNS with Namecoin (NMC), and then by encouraging a "trust only those you know" policy.[4]

"Namecoin is an open source decentralized key/value registration and transfer system based on Bitcoin technology".[16] Namecoin "squares Zooko's Triangle", meaning, it makes it possible to have domain names (and other types of identifiers) that are:

- **Authenticated:** users can be certain that they are not speaking to an impostor
- **Decentralized:** there is no central authority controlling all the names
- **Human-readable:** names look just like today's domain names

However, by itself, Namecoin does not provide the means by which ordinary users can take advantage of the features it provides. Using Namecoin is far too cumbersome for the vast majority of internet users, even those with years of computer expertise. For one, it cannot be used on mobile devices (like iPhones) in its current state because of its network requirements.

DNSNMC provides the missing "glue" to the Namecoin blockchain that makes it immediately accessible to clients of all types with *zero configuration*. A network administrator need only enter the IP address of a DNSNMC-compliant DNS server to instantly make the information within the blockchain accessible to all of the users that she (or he) provides internet access to.

To be assured of the authenticity of answers provided by a DNSNMC server, clients must have its public key fingerprint. With these two pieces (the server's IP address and the server's fingerprint), users are given strong authentication for all of the information that resides within the blockchain. Of course, we do not claim that this system provides *perfect authentication*[5], but rather it provides authentication that is *meaningful*. Once this relationship has been established between the DNSNMC server and its clients, the

[4] In contrast to the "trust a bunch of untrustworthy strangers" policy that today's browsers implement.

[5] Such a system is most likely impossible. Even if the person is standing right in front of you, they might not be themselves. Watch *The Exorcist* if you don't believe me. ;-)

clients are *guaranteed* to receive accurate values from the blockchain, *so long as the software involved (both server & client) and their respective keys (public and private) are not compromised.*

Ensuring the integrity of the software and keys involved is an orthogonal problem that affects all authentication systems, and thus is outside the scope of this paper.

The magic doesn't stop there. DNSNMC isn't just DNS + NMC, it's also an HTTP server. DNSNMC provides its clients with secure access to the Namecoin blockchain itself through a RESTful API. 😊

It accomplishes this by exposing a *meta-TLD*: **.nmc**

When a request is made to **dns.nmc** (or any of its subdomains), the DNSNMC server responds with its own public-facing IP address. This is what makes okTurtles possible.

If a client wants to know Joe Biden's public key fingerprint, they need only to load the following URL:

dns.nmc/id/jbiden

If the client uses a DNSNMC DNS server, some JSON will be returned containing all of the relevant info, along with a header proving the authenticity of the message. To avoid getting a cached IP address, simply prepend a random subdomain:

ajpow8jwojfsdjkl.dns.nmc/id/jbiden

5 okTurtles + DNSNMC

okTurtles takes the authentication provided by DNSNMC, and uses it to provide secure communication through virtually any website. It will initially be implemented as a web browser extension. okTurtles uses DNSNMC to authenticate individuals (instead of websites). This makes it possible to communicate with individuals around the world through almost any website, and without any action or intervention on the part of the site operator. Specific use cases for okTurtles include establishing secure communication

with individuals on social networks like Facebook, and the sending of secure email via web email clients like Google's Gmail.

okTurtles employs DNSNMC for secure authentication, and uses asynchronous OTR to provide plausible deniability and PFS. Thus, okTurtles fulfills the criteria outlined in this document for meaningful security.

When a user installs the okTurtles browser extension, they will be asked to claim an identity and to choose a password (known only to them) that will be the foundation upon which their identity is verified. A public key is generated on their behalf, and the fingerprint to this key is sent to the Namecoin blockchain (along with any supplied information about their identity, such as their full name, their online handles, etc.).

Whenever an okTurtles user clicks on an HTML `<textarea>` field, okTurtles will scan the entire page to try and use site-specific plugins to figure out who they are trying to talk to. If it is able to find some form of identity (for example, on Facebook this would be their friend's name), it will then check the blockchain to see if it can find a matching identity, and if not, the user can manually enter their friend's unique Namecoin id (id), or invite their friend to install okTurtles and create a global identity.

Similarly to how web browsers come with root certs, okTurtles will have a list of DNSNMC server IP addresses and their corresponding public key fingerprints. To avoid falling into the same trap that web browsers enjoy today with Certificate Authorities, okTurtles will encourage the user to use their own DNSNMC server, whether it's one that they have setup themselves, or one that belongs to someone they trust. Defaults are provided to encourage adoption and to make the software function immediately upon install.

5.1 Note on JavaScript Cryptography

okTurtles makes extensive use of JavaScript to enhance the functionality of HTML `<textarea>` tags, and to transparently encrypt and decrypt messages. Ideally, we would like to do as much of this in JavaScript as possible because JavaScript is easier to maintain, and actually has several security advantages over low-level languages (no NULL pointer exceptions, for example). We are well aware that some concerns have been raised about JavaScript cryptography.[17] Most of these concerns have to do with

the integrity of the code during to phases: (1) distribution (updates to the extension), and (2) when JavaScript is injected into a potentially malicious webpage.

Verifying code updates to the extension is easily accomplished thanks to DNSNMC, so that is not a concern. Verifying the integrity of JavaScript that has been injected into an arbitrary webpage is a more challenging problem and must be adequately addressed. Our plan is simple and consists of two primary approaches:

1. Investigate the capabilities of DefensiveJS [18] and JavaScript sandboxing [19].
2. For any crypto that cannot be safeguarded on the page, we will implement it in the browser extension in a manner that shields it from the page.

We'd like to remind the reader that this document merely provides a brief description of the design of DNSNMC and okTurtles, not a complete one.

6 Conclusion

In this paper we described the problems that plague today's online security, and introduced two free and open source projects that address them.

We hope you enjoyed it!

For more information, please visit our website (links below). The .bit link is for those already using Namecoin for DNS.

okturtles.com or okturtles.bit

We look forward to the day that all domains (not just .bit), are looked up via the blockchain. That will be the day that you actually own your domains[6], pay almost nothing for them, and your connection will be MITM-free! :-)

[6] Unlike traditional domains, domains (and other identifiers) in the blockchain cannot be "seized".

7 References

[1] A. Swartz, "Squaring the Triangle: Secure, Decentralized, Human-Readable Names", <http://www.aaronsw.com/weblog/squarezooko> (6 December 2013).

[2] M. Rumold, "EFF Victory Results in Release of Secret Court Opinion Finding NSA Surveillance Unconstitutional", <https://www.eff.org/deeplinks/2013/08/eff-victory-results-expected-release-secret-court-opinion-finding-nsa-surveillance> (6 December 2013).

[3] J. Bamford, "The NSA Is Building the Country's Biggest Spy Center (Watch What You Say)", <http://www.wired.com/threatlevel/2012/03/ff_nsadatacenter/all/> (6 December 2013).

[4] J. Stray, "FAQ: What You Need to Know About the NSA's Surveillance Programs", ProPublica, <http://www.propublica.org/article/nsa-data-collection-faq> (6 December 2013).

[5] B. Schneier, "Evidence that the NSA Is Storing Voice Content, Not Just Metadata", Schneier on Security, <https://www.schneier.com/blog/archives/2013/06/evidence_that_t.html> (6 December 2013).

[6] J. Howerton, "NSA Can Spy on 75 Percent of ALL U.S. Internet Traffic, Says Stunning New Report", TheBlaze.com, <http://www.theblaze.com/stories/2013/08/20/report-nsas-broad-reach-covers-75-percent-of-all-u-s-internet-traffic/> (6 December 2013).

[7] M. Kassner, "Does using encryption make you a bigger target for the NSA?", TechRepublic, <http://www.techrepublic.com/blog/it-security/does-using-encryption-make-you-a-bigger-target-for-the-nsa/> (6 December 2013).

[8] "Forward secrecy", Wikipedia, <https://en.wikipedia.org/wiki/Perfect_forward_secrecy> (6 December 2013).

[9] "Plausible deniability", Wikipedia <https://en.wikipedia.org/wiki/Plausible_deniability#Use_in_cryptography> (6 December 2013).

[10] "Transport Layer Security", Wikipedia, <https://en.wikipedia.org/wiki/Transport_Layer_Security> (6 December 2013).

[11] T. Dierks, "The transport layer security (TLS) protocol version 1.2" (2008).

[12] "X.509", Wikipedia, <https://en.wikipedia.org/wiki/X.509> (6 December 2013).

[13] "Man-in-the-middle attack", Wikipedia, <https://en.wikipedia.org/wiki/Man-in-the-middle_attack> (6 December 2013).

[14] P. Eckersley, "How secure is HTTPS today? How often is it attacked?", Electronic Frontier Foundation, <https://www.eff.org/deeplinks/2011/10/how-secure-https-today> (6 December 2013).

[15] "Spooks break most Internet crypto, but how?", Ars Technica, <http://arstechnica.com/security/2013/09/spooks-break-most-internet-crypto-but-how/> (6 December 2013).

[16] "Namecoin", Namecoin, <http://namecoin.info/> (6 December 2013).

[17] "Javascript Cryptography Considered Harmful", Matasano Web Security Assessments for Enterprises, <http://www.matasano.com/articles/javascript-cryptography/> (6 December 2013).

[18] "Defensive JavaScript home", Defensive JavaScript home, <http://www.defensivejs.com/> (6 December 2013).

[19] "Welcome to Oasis", Welcome to Oasis, <http://oasisjs.com/> (6 December 2013).

[20] "Iranian Man-in-the-Middle Attack Against Google Demonstrates Dangerous Weakness of Certificate Authorities", Electronic Frontier Foundation, <https://www.eff.org/deeplinks/2011/08/iranian-man-middle-attack-against-google> (6 December 2013).

[21] "New NSA Leak Shows MITM Attacks Against Major Internet Services", Schneier on Security, <https://www.schneier.com/blog/archives/2013/09/new_nsa_leak_sh.html> (6 December 2013).

[22] Laurie, "Certificate Authority Transparency and Auditability", <http://www.certificate-transparency.org/original-proposal> (6 December 2013).

[23] "DNSSEC – What Is It and Why Is It Important?", ICANN, <http://www.icann.org/en/about/learning/factsheets/dnssec-qaa-09oct08-en.htm> (6 December 2013).

[24] M. Marlinspike, "Convergence | Beta", <http://convergence.io/details.html> (7 December 2013).

[25] Various, "[Cryptography] DNSSEC = completely unnecessary?", <http://comments.gmane.org/gmane.comp.encryption.general/16353> (7 December 2013).

[26] "DNSSEC Downtime: Outages & Validation Failures", <http://ianix.com/pub/dnssec-outages.html> (7 December 2013).