# It’s DNSSEC Not DNSSUX

**Lori MacVittie, 2009-18-11**

*Whenever keys, certificates, and PKI enter into a security solution's architecture the solution almost always becomes overly complex. DNSSEC is no exception, but it doesn't have to be.*

DNS plays a role in every application on the Internet. It is the 411 of the Internet, essentially, without which the millions of users that don't memorize the IP addresses associated with domain names would be utterly lost. But DNS is vulnerable to exploitation and has, in fact, been exploited in the past. Like any core infrastructure upon which we depend to conduct business, communicate, and generally entertain ourselves, it needs to be protected. DNSSEC (DNS Security Extensions) is a protocol and management extension to DNS designed to guarantee the authenticity of responses. Its basic theory is sound, but putting into practice can quickly turn DNSSEC into DNSSUX, at least for DNS administrators.

It should be no surprise, then, that the difficulties inherent in such an effort are causing delays in implementation. VeriSign, for example, mentions the "size" of the zone as a reason the top level domains (TLD) are taking so long to adopt DNSSEC:

> "VeriSign is moving forward with the implementation of DNSSEC across all of the Top Level Domains that we operate," VeriSign said in a statement to Network World. ".com will most likely be the last TLD to adopt DNSSEC due to the size of the zone. We anticipate full implementation of DNSSEC to be complete across all TLDs in approximately 24 months."

Given the complexity and requirements involved in a DNSSEC deployment, that's actually no surprise.

## WHAT'S SO DIFFICULT ABOUT DNSSEC?

The premise behind DNSSEC is that responses to DNS queries need to be trustable. Following the example of web-based applications, DNSSEC applies the principle of signatures via public/private key encryption as a means to achieve that trust. Essentially DNSSEC is the wrapping of the DNS infrastructure within a trusted, PKI-based superstructure that validates through certificates managed records (zones).

Deploying DNSSEC involves signing zones with public/private key encryption and returning DNS responses with signatures (new RRSIG resource record). A client's trust of those signatures is based on a chain of trust established across administrative boundaries, from parent to child zone, using new DNSKEY and DS resource records. DNSSEC also calls for "authenticated denial of existence" via NSEC (and/or NSEC3) records. And complicating the deployment is the requirement that any DNSSEC deployment must manage cryptographic keys: multiple key generation, zone signing, key swapping, key rollover and timing, and recovering from compromised keys.

**DNS CACHE POISONING**

Cache poisoning occurs when a resolver or recursive DNS server queries another server in an effort to answer a query, and an attacker spoofs the query response to the resolver or recursive server. This can occur when the attacker impersonates the queried server by using an appropriate DNS message. In the case of the recursive server receiving such an answer, it not only supplies the resolver with the falsified information, it caches the information such that future queries, at least during the valid time interval of the answer, are answered with the same falsified information.

Currently BIND, the most common implementation of DNS, supports DNSSEC. There are solutions that combine BIND clones with DNSSEC signing devices that sit beside the DNS infrastructure. So far there doesn't seem to be an implementation that simultaneously achieves ease of deployment, scalability, and high performance.

What's needed is a way to reduce the complexity and the costs associated with layering this PKI infrastructure atop - or alongside - the existing DNS infrastructure while ensuring that DNSSEC is properly implemented and supported.

## WE BORROWED FROM WEB APPLICATIONS FOR THE FIRST SOLUTION, WHY NOT THE SECOND?

These very same problems have been, and continue to be, felt by administrators of web applications that need to implement secure communications via HTTPS. The best answer to managing SSL implementations is to centralize them; essentially implementing a proxy-based approach to securing all web applications simultaneously using a common SSL implementation. Load balancers and later application delivery controllers have been providing this capability for years and it is practically commoditized at this point. We call it "table stakes" in product management because it's one of the features you must support in application delivery to even get a seat at a potential customer's table.

So why aren't we applying that same logic to the problem of deploying and managing DNS, especially large scale DNS implementations?

Turns out that we are. But I'm guessing you knew *that* was coming, didn't you?

A DNSSEC-enabled global server load balancer (GSLB) can support both a centralized, proxy-style DNSSEC implementation or it can be deployed as a stand-alone, DNSSEC-enabled DNS server a la BIND. The difference between a stand-alone deployment of a DNSSEC-enabled GSLB and a BIND + DNSSEC signing solution deployment is that the former integrates DNS and DNSSEC capabilities and does not require separate solutions to provide a workable solution. Even if you aren't using GSLB to provide load balancing across multiple datacenters or cloud computing environments (a la cloud balancing), you can still take advantage of a DNSSEC-enabled GSLB to basically "proxy" DNS queries and centralize signing of responses through a single, centralized solution.

DNSSEC doesn't have to be DNSSUX.

F5 Networks, Inc. | 401 Elliot Avenue West, Seattle, WA 98119 | 888-882-4447 | f5.com

F5 Networks, Inc.
Corporate Headquarters
info@f5.com

F5 Networks
Asia-Pacific
apacinfo@f5.com

F5 Networks Ltd.
Europe/Middle-East/Africa
emeainfo@f5.com

F5 Networks
Japan K.K.
f5j-info@f5.com