DNS SECURITY - II

13.1.2 Security Types

We classify each threat type below. This classification simply allows us select appropriate remedies and strategies for avoiding or securing our system. The numbering used below relates to diagram 1-3.

(1) The primary source of Zone data is normally the Zone Files (and don't forget the named.conf file and even logs which can contain lots of interesting data as well). This data should be secure and securely backed up. This threat is classified as **Local** and is typically handled by good system administration practices (minimal permissions, appropriate access control).

(2) If you run slave servers you will do zone transfers. **Note:** You do not have to run with slave servers, you can run with multiple masters and eliminate the transfer threat entirely. This is classified as a **Server-Server (Transaction)** threat and there are multiple configuration techniques available from simple IP based ACLs to TSIG based cryptographic solutions.

(3) The BIND default is to **deny** Dynamic Zone Updates. If this service is enabled or is required, it poses a serious threat to the integrity of your Zone files and should be protected. There is a significant difference in complexity when moving from allowing on-server (localhost only) Dynamic Updates to provisioning of remote access. This is classified as a **Server-Server (Transaction)** threat and there are multiple configuration techniques available from simple IP based ACLs to TSIG or SIG (0) based cryptographic solutions.

(4) The possibility of Remote Cache Poisoning due to IP spoofing, data interception and other hacks is a judgement call. If running a simple, non-controversial, non-revenue earning web site the threat is probably verging on the non-existent and the effect, in the unlikely event it does happen, is merely annoying. If the site is high profile, open to competitive threat or is a high revenue earner the threat is both significant and the effects potentially ranging from embarassment to loss of earnings or repution. This is classified as a **Server-Client** threat and there are complex configuration techniques available using DNSSEC based cryptographic solutions.

(5) The current standards do allow for end-to-end DNSSEC implementation. As of 2013 this was at best in the embryonic phase and would involve major upgrades to all PC/Server based stub resolvers. This is classified as a **Server-Client** and **Client-Client** threat.

13.1.3 Security - Local

Normal system administration practices such as ensuring that files (configuration and zone files) are securely backed-up, proper read and write permissions applied and sensible physical access control to servers may be sufficient.

Implementing a Stealth (or Split) DNS server provides a more serious solution depending on available resources.

Finally you can run BIND (named) in a **chroot jail**.

13.1.4 Server-Server (TSIG Transactions)

Zone transfers. If you have slave servers you will do zone transfers. BIND provides **Access Control Lists (ACLs)** which allow simple IP address protection. While IP based **ACLs** are relatively easy to subvert they are a **significantly** better than doing nothing and require very little work. It is possible to run with multiple masters (no slaves) and eliminate the threat entirely. Zone files will require manual synchronization but this may be a simpler solution if changes are not frequent.

Dynamic Updates. If you must run with this service it should be secured. BIND provides **Access Control Lists (ACLs)** which allow simple IP address protection but this is probably not adequate unless you can secure the IP addresses, for example, all systems are behind a firewall/DMZ/NAT or the updating host is using a private IP address.

TSIG/TKEY If all other solutions fail DNS specifications (RFCs 2845 - TSIG and RFC 2930 - TKEY) provide authentication protocol enhancements to secure these Server-Server transactions.

TSIG and **TKEY** implementations are messy but not too complicated - simply because of the scope of the problem. With **Server-Server** transactions there is a finite and normally small number of hosts involved. The protocols depend on a **shared secret** between the master and the slave(s) or updater(s). It is further assumed that you can get the **shared secret** securely to the peer server by some means not covered in the protocol itself. This process, known as **key exchange**, may not be

trivial (typically long random strings of base64 characters are involved) but you can use the telephone(!), mail, fax or PGP email amongst other methods.

The **shared-secret** is open to **brute-force** attacks so frequent changing of **shared secrets** may become a fact of life. Sample configurations for securing Dynamic Updates when using DHCP to update either or both of the forward and reverse zone files are provided.

13.1.5 Server-Client (DNSSEC)

The classic Remote Poisoned cache problem is not trivial to solve simply because there may an infinitely large number of Remote Caches involved. It is not reasonable to assume that it can use a **shared secret**. Instead the mechanism relies on public/private key (asymmetric) cryptography. The DNSSEC specifications (RFC 4033, RFC 4034 and RFC 4035) attempt to answer three questions:

1. Authentication - the DNS responding really is the DNS that the request was sent to.
2. Integrity - the response is complete and nothing is missing.
3. Integrity - If a DNS record is not available it can be verified (Proof-of-non-existence - PNE).

Source : http://www.zytrax.com/books/dns/ch13/