

that roam from one network segment to another would benefit from keeping the same domain name even if their IP addresses change as they move.

Another interesting application of dynamic updates is updating certificates stored in the domain name system [11]. A CA could dynamically add new certificates when they are created. If the certificates could be updated easily and automatically, their lifetimes could be shorter, and the risks related to revocation could be reduced.

We also see the dynamic update mechanism as a potential replacement for the traditional (and error-prone) approach of manually editing the zone files. With dynamic updates, the changes could be done remotely and the client could perform sanity checks on the data. This, however, is a usability issue with very little relevance to security.

The operations that can be performed with dynamic updates are adding and deleting resource records (RRs).

3.1 Existing solutions

An early proposal for securing dynamic updates suggested that the KEY RR would include a few bits (the signatory field) that indicate if the key may be used for dynamic updates [7]. The KEY record's name, class, and the few bits would be used to encode the update policy. However, since there are only four bits available for the purpose, this approach to authorization is hardly flexible enough for most situations. Attempts to use more bits to gain more flexibility have not been very successful either [21]. Indeed, the use of KEY signatory bits for encoding policy has been considered obsolete for a long time.

The current proposal for securing dynamic updates abandons the concept of storing the policy inside DNS, and simply states that the policy is fully implemented in the primary server's configuration [26]. The proposal suggests a couple of things which should be supported by policy im-

| Field | Description | Example |
|-------------------------------|---|--------------------|
| <i>What is being modified</i> | | |
| zone | zone name | example.com |
| name | fully-qualified name | saturn.example.com |
| type | RR set type as text, or ANY | A |
| ttl | time-to-live in seconds | 3600 |
| rdata | record data, in zone file format (may be empty) | 192.168.1.2 |
| | add or delete | |

friendliness cannot be guaranteed, the transactions need to be authenticated with one of the other DNSSEC transaction

binding between the key and the name obviously needs to be secure.

However, when using decentralized trust management for making an access control decision, the server really

the 1996 IEEE Symposium on Security and Privacy, pages 164-173, Oakland, California, May 1996.

- [6] Matt Blaze, John Ioannidis, and Angelos D. Keromytis. Trust management for IPsec. To appear in *Proceedings of the Network and Distributed System Security Symposium (NDSS 2001)*, San Diego, Cali-