6.033 Critique 1: DNS

Jisoo Min

1 Introduction

The domain name system (DNS) is a critical infrastructure of the Internet. It resolves, or translates, human-friendly domain names to Internet Protocol (IP) addresses. DNS completes look-ups of a requested name on name server modules to find a path to an appropriate IP address. The purpose is to allow clients easily use the Internet without knowing the IP addresses.

The system designers seek to eliminate extra work from the user and to enhance name discovery performance by enforcing modularity. The system, however, has some trade-offs between its design goals of performance and security.

2 Background

Telephone book model or central directory model used in the past resulted in an excessive network traffic and a work overload for the centralized administration. In the late 1980s, as a response to the high demand of Internet resources, programmers sought a more efficient system. As a result, newly designed system DNS now serves the purpose with multiple server modules each in charge of its submodules. [4.4.1]

Without clever search methods and supplementary functions, however, hierarchical structure design by itself faces difficulties in achieving the desired performance. We shall focus this critique on modularity and on additional mechanisms used to accommodate users’ needs and any potential failures. Then security limitation shall be briefly discussed.

3 System Design

3.1 Modularity

The system relies heavily on modularity to achieve a simple, resilient name resolution process. With hierarchical namespace and distributed domain names, DNS decreases the load on any particular server. The benefit that modularity provides is twofold.

First, Internet resources are easy to manage under the hierarchical structure. Different from a model of complex net of records, tree structure of domains provides the precise information about the depth of a node (or a name) and about a path to it from the root. The benefit extends to name updates and modifications. [4.4]

Second, modularity allows division of labor in name resolution processes. When given a path name to resolve, going from the most to the least significant component of the name, DNS looks for a referral record or a name record only within one name server at a time. Each name server manages a set of records that are in turn mapped to domain submodules. [4.4.2] For the case in which the client, the name server, and the target domain are in the same domain, one request is sufficient. [4.4.1] Because each naming authority can function independently under modularity, the system does not suffer from traffic.

3.2 Performance

The DNS emphasizes performance in its design to ensure satisfiable Internet experience for the clients. Here, performance can be defined as a
reduction in process time without the loss of accuracy.

3.2.1 Cache

DNS cache is the most important contributing factor to performance enhancement. [4.4.1] During a name resolution process, discovered paths are cached so that name servers, which originally do not have the record for the requested path name, can later directly ask for information from the authoritative name server. Because redundant searches are common, cache look-up, as opposed to path search, successfully saves time.

3.2.2 Hierarchy

Performance improvement is partially a natural consequence of independent naming authorities. A few identical lower-level domain names of two different path names do not affect the performance. DNS will have already searched through a different branch based on the higher-level domain referrals. [4.4.2]

Moreover, the hierarchical structure allows for a recursive search mechanism. DNS does not always query the root name server on the first round. Instead, the search can be directed to any nearby name server, and then go up the hierarchy if needed. This initial name server can act as the root server in that it fetches the answer and returns it to the user. Under this method, root server is no longer called on the initial query for all searches. [4.4.1]

3.3 Simplicity

By nature of query service, DNS is simple for the user. In other words, clients can type easy, symbolic names to access Internet resources.

In addition to the basic easiness on user interface, synonyms successfully enhance simplicity. The user may want to bind a new name to an existing name, update a machine, or assign separate life times to servers. DNS supports all these needs with the use of synonyms. Synonyms are aliases; multiple path names can point to a single IP address or a single path name can point to multiple IP addresses. [4.4.3]

3.4 Fault-Tolerance

The last major design goal of the DNS is fault-tolerance. Biggest common failure is power outage and local network disconnection. Because ubiquity and accuracy is the key to Internet services, DNS responds to these problems by introducing replicas. If there is a failure in some component of the hierarchy, DNS cannot give up on all servers under that faulty branch. [4.4.3] With replicas, the system can successfully recover in no time and even prevent the fault by dividing up the traffic in advance.

One limiting factor to synonyms is long binding hours of up to 36 hours. Life times of servers are typically long enough that immediate need of a new synonym is rare. DNS mitigates this limitation with a large set of replicas.

3.5 Limitations

As discussed above, synonyms and caches are simple and powerful for the user. DNS cache poisoning attack, however, exists as a major vulnerability among beneficial design goals. [Bellovin 1995] Due to structural simplicity of DNS, malicious hackers can inject harmful IP address to the cache. Even worse, under the hierarchical structure, the problem can propagate quickly. It is recommended that DNS implements cryptography techniques as its defense mechanism. [Arends 2005]

4 Conclusion

The DNS attempts to provide a simple but high-performance service for users of the Internet. It achieves its design goals by maintaining various types of data in hierarchy. Centered around this model, DNS focuses on system performance, simplicity, and fault-tolerance as primary design goals. With over millions of authoritative servers now, DNS has become an imperative system.
References

