DNS Cache Poisoning: 정의와 방어

By Tom Olzak
편역: vangelis(securityproof@gmail.com)

Domain Name System(DNS)이 없다면 인터넷은 멈추게 되거나 가능하지 않을 수 있다. 이 글에서 보게 되듯 전 세계 수많은 노드(node)들에 대한 주소를 유지하고 분배해주는 것이 DNS를 적절하게 운영하는 것의 기본이다.

DNS 보안에는 몇 가지 측면이 있다. 이 글은 가장 위험한 공격 형태들 중의 하나인 DNS cache poisoning에 초점을 맞출 것이다. 이 위협에 대해 완벽하게 이해하기 위해 우리는 DNS가 작동하는 방법, 크래커가 cache poisoning을 쉽게 하기 위한 두 가지 방법, 이런 형태의 공격이 여러분의 조직에 어떤 영향을 미칠 수 있는지, 정보 자산을 보호하기 위해 취할 몇 가지 단계들 등을 살펴볼 것이다.

DNS란 무엇인가?

인터넷과 TCP/IP[1]의 세계에서 IP 주소는 출발지(source)에서 목적지(destination)으로 패킷[2]을 라우팅하기 위해 사용된다. 하나의 IP 주소, 예를 들어 203.192.135.234는 기억하기 어렵지는 않다. 하지만 IP 주소가 많을 경우 어떤 서버/노드가 각 주소와 관련되어 있는지를 포함해 이 주소들을 기억하고 흔적을 추적하려고 노력하는 것은 끔찍한 일이 될 수 있다. 그래서 우리는 IP 주소 대신 통신 하고자 원하는 시스템들을 가리키는 도메인 이름[3]을 사용한다.

실제 인터넷 도메인 이름 Google.com을 예로 들어보자. 브라우저의 주소 바에 Google 도메인 이름을 입력하고 엔터를 치면 Google 페이지가 나타난다. 이것은 PC가 Google.com을 IP 주소로 resolve[4]하기 위한 프로세스를 실행했기 때문이다. 그 IP 주소를 가짐으로써 인터넷 상의 또 다른 하나의 시스템과 세션을 시작할 수 있는 시스템이 가능해진다. IP 주소 확인(resolution)이 일어날 수 있는 두 가지 방법을 알아보자.

---

[1] http://en.wikipedia.org/wiki/TCP/IP

[2] http://en.wikipedia.org/wiki/Packet

[3] http://www.answers.com/topic/domain-name?method=22

[4] (역자 주) resolve라는 단어를 DNS를 다룰 때 자주 등장하게 되는데, 이 단어에는 '결정하다', '분해하다', '변형하다', '해결하다' 등의 의미가 있다. DNS를 다룰 때 resolve는 도메인 주소를 분석하여 IP 주소로 변경하는 것을 의미한다.

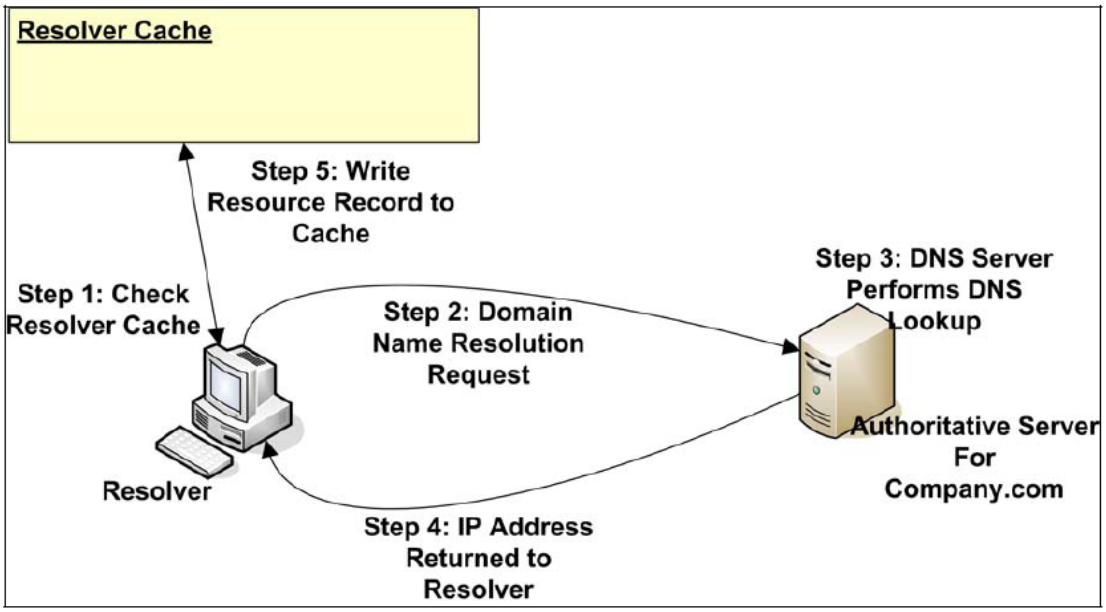

Figure 1: Internal DNS Server Lookup

Figure 1은 타깃 시스템과 DNS 서버가 내부에 존재할 때 domain name/IP address resolution 과정을 보여준다. 이 예에서 워크스테이션[5]은 *Farpoint.company.com*라는 이름을 가진 도메인의 서버와 세션을 구성해야 한다. 워크스테이션이 DNS를 구현하기 위해 DNS 클라이언트 또는 *resolver*[6]를 실행하고 있어야 한다. Resolver는 다음 아래의 과정을 시작하고, 그 결과 도메인 이름은 IP 주소로 변환(conversion)된다(Microsoft TechNet, 2005).

Step 1: resolver는 워크스테이션의 메모리에 있는 resolver cache가 Farpoint.company.com에 대한 엔트리를 가지고 있는지 확인한다. 만약 워크스테이션이 그 도메인 이름을 IP 주소로 이전에 변경(resolve)한 적이 있다면 그 엔트리는 존재할 것이며, 그 엔트리의 TTL(Time to Live)[7]을 초과하지 않았을 것이다. **Figure 1**에서는 Farpoint.company.com에 대한 엔트리는 없는 상태이다.

Step 2: resolver cache에서 어떤 엔트리도 발견하지 못했기 때문에, 그 resolver는 내부 DNS 서버에 resolution query를 보낸다.

Step 3: DNS 서버가 그 질의를 받으면 먼저 그것이 company.com 도메인에 대해 신뢰할 수

---

[5] (역자 주) 이 글에서 워크스테이션은 Step 1에서 보이는 reolver가 설치된 시스템을 가리킨다.

[6] (역자 주) BIND DNS의 클라이언트 프로그램을 resolver라고 한다.

[7] http://en.wikipedia.org/wiki/Time_to_live#Time_to_live_of_DNS_records

(역자 첨가, 출처: terms.co.kr) TTL은 IP 패킷 내에 있는 값으로서, 그 패킷이 네트웍 내에 너무 오래 있어서 버려져야 하는지의 여부를 라우터에게 알려준다. 패킷들은, 여러 가지 이유로 적당한 시간 내에 지정한 장소에 배달되지 못하는 수가 있다. 예를 들어, 부정확한 라우팅 테이블의 결합은 패킷을 끝없이 순환하게 만들 수도 있다. 일정한 시간이 지나면 그 패킷을 버리고, 재전송할 것인지를 결정하도록 그 사실을 발신인에게 알리기 릴 수 있게 하기 위한 해결책으로 TTL이 사용된다.

있는(authoritative) 것인지 확인한다. 즉, 그것은 company.com이 상주하는 zone[8]을 관리할 책임이 있는 것인가? 만약 그렇다면 DNS 서버는 내부 zone table에서 검색(lookup)을 수행한다. 이 경우, Farpoint.company.com에 대한 IP 주소를 가지고 있는 호스트 RR(Resource Record)[9]를 찾아낸다.

Step 4: Farpoint.company.com의 IP 주소는 resolver로 리턴된다.

Step 5: resolve된 도메인 이름과 IP 주소는 resolver cache에 저장된다. **Figure 2**는 워크스테이션의 resolver cache 내용의 실제 목록이다. 그 IP 주소는 Farpoint와 접촉하는데 사용된다.

Figure 2: Actual Resolver Cache Listing

[8] http://en.wikipedia.org/wiki/DNS_zone

[9] http://en.wikipedia.org/wiki/Domain_Name_System#Types_of_DNS_records

Figure 1에서 타깃 서버는 요청자의 내부 네트워크에 위치했다. 하지만 이와는 달리 그 타깃 디바이스가 인터넷의 어딘가에 위치하는 경우가 많다. 이들 경우 과정이 약간 다르다. 이 두 번째 DNS resolution 과정은 **Figure 3**을 통해 알아볼 것이다.

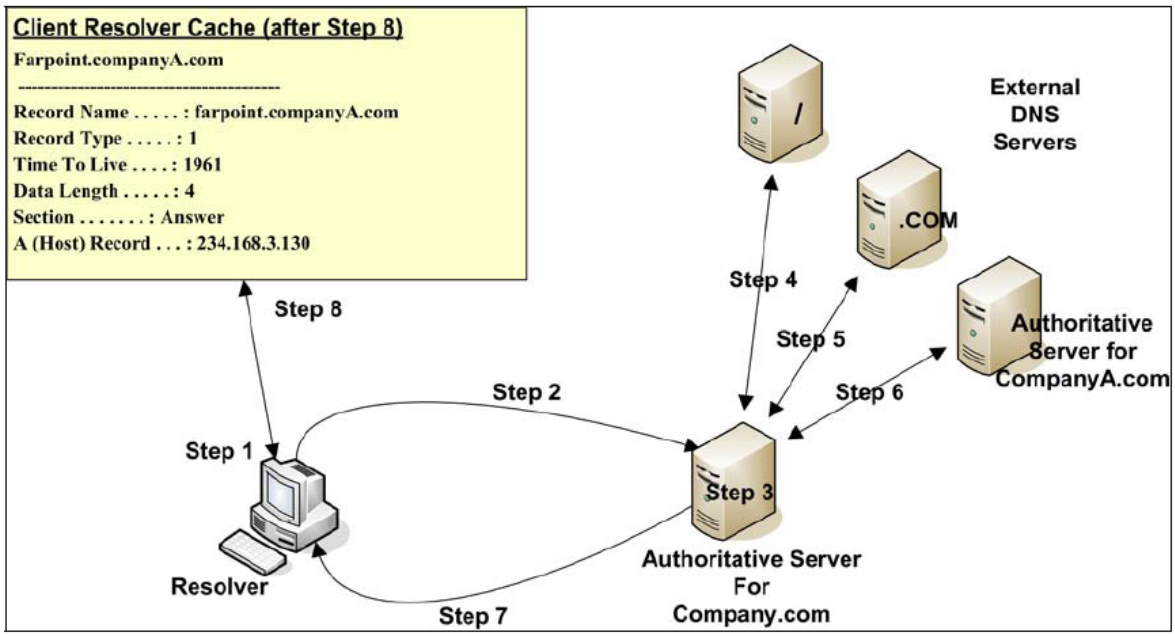

Figure 3: Recursive DNS Query

Step 1: resolver는 Farpoint.companyA.com에 대한 엔트리를 가지고 있는지 워크스테이션의 메모리에서 resolver cache를 확인한다.

Step 2: resolver cache에서 어떤 엔트리도 발견하지 못했기 때문에, 그 resolver는 내부 DNS 서버에 resolution request를 보낸다.

Step 3: DNS 서버가 그 요청을 받을 때 먼저 그것이 신뢰할 수 있는 것인지 확인한다. 이 경우 companyA.com에 대해서는 신뢰할 수 없다. DNS 서버가 취하는 다음 행동은 Farpoint.companyA.com에 대한 엔트리가 존재하는지 여부를 확인하기 위해 로컬 cache를 확인하는 것인데, 여기서는 하지 않는다. 그래서 Step 4에서 도메인 이름을 resolve하거나 또는 도메인 이름 엔트리가 존재하지 않는다는 것이 명백해지는 지점에 도달할 때까지 내부 DNS 서버는 반복적으로 외부 DNS 서버들에 질의를 하는 과정을 시작한다.

Step 4: DNS 질의가 인터넷의 "root" 서버들[10] 중의 하나로 보내진다. 요청을 받은 root 서버는 그 .COM 인터넷 공간에 대해 신뢰할 수 있는 서버의 주소를 알려준다.

Step 5: DNS 질의가 .COM에 대해 신뢰할 수 있는 서버로 보내진다. companyA.com 도메인에 대해 신뢰할 수 있는 DNS 서버의 주소가 리턴된다.

---

[10] http://www.answers.com/topic/root-server?method=22

Step 6: DNS 질의가 companyA.com에 대해 신뢰할 수 있는 서버로 보내진다. Farpoint.companyA.com의 IP 주소가 리턴된다.

Step 7: Farpoint에 대한 IP 주소는 클라이언트 resolver로 리턴된다.

Step 8: 하나의 엔트리가 resolver cache에 만들어지고, Farpoint.companyA.com와의 세션이 함께 시작된다. 클라이언트 resolver의 관점에서 보면 이 과정은 순환 질의(recursive query)로 알려져 있다. 적절한 소프트웨어와 함께 클라이언트 resolver는 Step 4,5,6에서와 같이 반복적인 질의를 수행할 수 있다.

DNS가 어떻게 작동하는지 이제 이해했으므로 어떻게 이 과정이 하나 또는 그 이상의 DNS cache들과 함께 작동하는데 사용될 수 있는지 알아보자.

DNS Cache Poisoning란 무엇인가?

DNS cache poisoning은 클라이언트 또는 서버 중의 하나에 있는 resolver caches의 기록들을 변경하거나 추가하고, 어떤 도메인에 대한 DNS 질의는 의도한 도메인 대신 공격자의 도메인에 대한 IP 주소를 리턴한다. 이것이 어떻게 작동하는지 **Figure 4**의 각 단계를 알아보자.

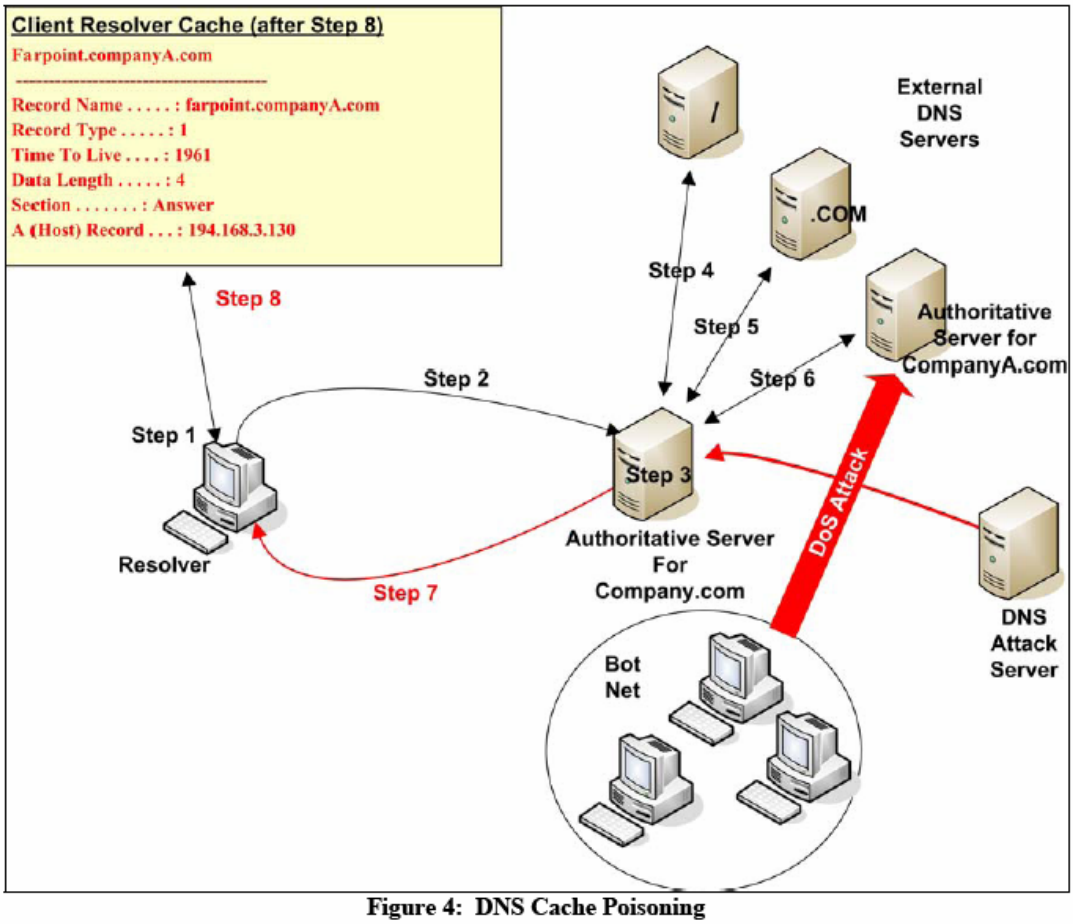

Figure 4: DNS Cache Poisoning

Step 1: resolver는 Farpoint.companyA.com에 대한 엔트리가 있는지 워크스테이션의 메모리에 있는 resolver cache를 확인한다.

Step 2: resolver cache에서 어떤 엔트리도 발견하지 못했기 때문에, 그 resolver는 내부 DNS 서버에 resolution request를 보낸다.

Step 3: DNS 서버가 그 요청을 받을 때 그것의 신뢰 여부를 먼저 확인한다. **Figure 4**에서는 companyA.com에 대해 신뢰성이 없다. 그것이 취하는 다음 행동은 Farpoint.companyA.com에 대한 엔트리가 존재하는지 확인하기 위해 로컬 cache를 체크하는 것이며, 여기서는 하지 않는다. 그래서 Step 4에서 도메인 이름을 resolve하거나 또는 도메인 이름 엔트리가 존재하지 않는다는 것이 명백해지는 지점에 도달할 때까지 내부 DNS 서버는 반복적으로 외부 DNS 서버들에 질의를 하는 과정을 시작한다.

Step 4: DNS 질의가 인터넷의 "root" 서버들 중의 하나로 보내진다. 그 root 서버는 그 .COM 인터넷 공간에 대해 신뢰할 수 있는 서버의 주소를 알려준다.

Step 5: DNS 질의가 .COM에 대해 신뢰할 수 있는 서버로 보내진다. companyA.com 도메인에 대해 신뢰할 수 있는 DNS 서버의 주소가 리턴된다.

Step 6: DNS 질의가 companyA.com에 대해 신뢰할 수 있는 서버로 보내진다. 이것은 한가지를 제외하고는 순환 질의를 위한 표준 과정과 동일하다. 크래커는 내부 DNS 서버의 cache에 포이즈닝하는 것을 결정한다. 질의를 가로채고 악의적인 정보를 리턴하기 위해 크래커는 transaction ID를 알아야 한다. 일단 그 transaction ID가 알려지면 공격자의 DNS 서버는 companyA.com에 대한 신뢰할 수 있는 서버로서 응답할 수 있다.

비록 이것이 오래된 DNS 소프트웨어(예를 들어, BIND 4 및 이전 버전)에서는 간단한 문제일 수 있으나 새로운 DNS 시스템들에는 보호수단들이 내장되어 있다. **Figure 4**에서는 각 질의의 예를 확인하기 위해 사용된 transaction ID는 랜덤화되어 있다. 하지만 그 transaction ID를 이해하는 것은 불가능하지는 않다. 필요한 것은 시간이다. 실제 신뢰할 수 있는 서버의 응답을 늦추기 위해 크래커는 DoS(Denial of Service) 공격을 시작할 botnet[11]을 이용한다. 신뢰할 수 있는 그 서버가 이 공격을 처리하려고 노력하는 동안 공격자의 DNS 서버는 transaction ID를 확인할 시간을 가진다.

일단 그 ID가 확인되면 질의에 대한 응답은 내부 DNS 서버로 보내진다. 하지만 그 응답 속에 있는 Farpoint.companyA.com에 대한 IP 주소는 실제로 공격자 사이트의 IP 주소이다. 이 응답은 내부 DNS 서버의 cache 안에 저당된다.

---

[11] http://en.wikipedia.org/wiki/Botnet

Step 7: Farpoint의 '가짜 IP'[12]는 클라이언트의 resolver로 리턴된다.

Step 8: 하나의 엔트리가 resolver cache에 만들어지고, 공격자 사이트와의 세션이 시작된다. 이 시점에서, 워크스테이션의 cache와 내부 DNS 서버의 cache는 poison된다. Farpoint.companyA.com의 내부 네트워크 requesting resolution 상에 있는 어떤 워크스테이션도 내부 DNS 서버의 cache에 등록되어 있는 공격자의 주소를 받게 된다. 이것은 그 엔트리가 삭제될 때까지 지속된다.

DNS cache를 poison하는데 사용되는 또 다른 한가지 방법은 공격자가 보낸 순환 질의(recursive query)를 사용하는 것이다. 그 질의는 질의 중에 있는 도메인의 신뢰할 수 있는 출발지로 타깃 서버가 연결하도록 강제할 수 있다. 일단 연결되면 하나 또는 그 이상의 도메인들에 대한 가짜 정보가 질의하고 있는 서버로 보내지고 그 서버의 cache로 포스팅될 수도 있다.

DNS cache를 poison하기 위해 공격자들이 사용하는 다른 방법들이 있지만 그 목적은 같다. 이제 DNS cache poisoning의 결과들을 알아보자.

Cache Poisoning의 잠재적인 결과들

Pharming[13]는 cache poisoning과 관계 있는 주요 위험이다. 크래커들은 다음 4가지 주된 이유들 때문에 pharming을 사용한다(Hyatt, 2006): 개인정보 절도, malware 유포, 거짓 정보 보급, 그리고 man-in-the-middle 공격

개인정보 훔침 및 사칭

공격자가 당신을 공격자의 사이트로 일단 방문하게 하여 당신을 사칭하기 위해 사용할 수 있는 정보를 남기도록 당신을 속이려고 할 것이다. 우리의 첫 번째 예에서 개인 정보를 훔치기 위한 한 가지 방법은 실제 Farpoint.companyA.com와 동일한 사이트를 만드는 것이다. 사용자가 poison된 cache 정보를 사용해 연결되면 이름, 주민번호, 주소 등을 입력하도록 할 것이다.

Malware 유포

cache poisoning을 이용하는 공격자의 또 다른 하나의 목적은 malware를 자동으로 유포하는 것이다. 악성코드를 인터넷에 배포하거나 예측할 수 없는 결과를 가져오는 것 대신 공격자의

---

[12] (역자 주) 공격자가 각종 공격을 위해 사용할 공격자 서버의 IP 주소

[13] (역자 주) pharming에 대해 간단히 말하면, DNS 서버 등을 조작하여 가짜 사이트로 접속을 유도하여 각종 정보를 훔쳐가는데 주로 사용되며, 원래 의도한 사이트의 주소를 그대로 입력하더라도 DNS가 조작되어 있기 때문에 공격자의 사이트로 연결된다. DNS cache poisoning을 공부하는 분이라면 DNS 서버를 직접 테스트 서버에 설치하여 각종 설정을 해보길 권한다.

사이트에 아무런 의심도 하지 않는 사용자들을 리다이렉트시키기 위해 공격자의 공격용 IP 주소를 사용하는 것은 좀더 집중적인 공격 벡터(vector)가 될 수 있다. 일단 워크스테이션이 악의적인 사이트와 세션을 시작하면 malware는 사용자가 개입하지 않거나 사용자가 모르게 그 워크스테이션으로 업로드 된다.

거짓 정보 보급

자기의 이익만을 도모하는 정보를 유포시키고자 하는 공격자들에게 유용한다. 이것은 큰 이익을 거두기 위해 주식 가격을 조작하는 등의 경우에도 사용될 수 있다.

Man-in-the-middle 공격

이 공격 형태에서 워크스테이션은 공격자의 서버와 세션을 시작한다. 공격자의 서버는 실제 타깃 사이트와 세션을 시작한다. 워크스테이션과 진짜 사이트 사이를 지나가는 모든 정보는 공격자의 서버를 지나가고 그래서 공격자의 서버는 이 정보를 가로챌 수 있다.

DNS 서버를 설정하고 배치하는 동안 보안을 염두에 두지 않을 경우 심각한 결과를 초래할 수 있다. 다음 섹션은 cache poisoning을 막는데 도움이 될 수 있는 가이드라인을 제공한다.

자산 보호하기

cache poisoning에 대한 첫 번째 방어층(layer of defense)[14]은 최신 버전의 DNS를 사용하는 것이다. BIND 9.3.x 또는 Microsoft Windows Server 2003에 기반을 둔 DNS는 초기 버전으로 구현된 DNS보다 훨씬 더 안전하다.[15] 앞에서 예를 든 poisoning을 성공적으로 완수하는 것이 더 어려워졌는데, 이는 이 시스템들이 transaction ID에 덧붙여 DNS 질의에 사용되는 포트[16]를 랜덤화시켰기 때문이다.

순환 질의는 내부 DNS 서버에 국한되어야 한다. 만약 순환 질의 기능을 가진 인터넷이 필요하면 내부 주소들로부터의 질의만을 받아들여야 한다. 이를 통해 외부 시스템들이 악의적인 의도를 가지고 질의를 보내는 것을 막을 수 있다. 다음 목록은 추가 가이드라인(Hyatt, 2005)이다.

▲ 외부 및 내부 DNS 서버를 물리적으로 분리시킨다.

▲ 인증 받은(2차 서버들) 디바이스들에 zone transfer[17]를 제한한다.

---

[14] (역자 주) 여기서 layer라는 단어를 사용한 것은 하나의 대책만으로는 부족하기 때문일 것이다.

[15] (역자 주) 이 글이 발표된 시점을 고려한다면 이에 대한 진의 여부를 확인하는 것이 필요하다.

[16] http://www.answers.com/topic/port-computing?method=22

[17] http://www.answers.com/topic/dns-zone-transfer?method=22

▲ zone transfer와 zone update에 디지털 서명을 하기 위해 TSIG [18] 를 사용한다(poisoning을 막는 가장 좋은 방법들 중의 하나는 'source'의 신뢰여부를 강제로 확인하는 것이다).

▲ 가능하면 동적 DNS update[19]를 제한한다.

▲ DNS 서버에 사용되는 BIND의 버전을 숨긴다.

▲ DNS 서버에 실행되는 불필요한 서비스들을 제거한다.

▲ 가능한 곳이라면 다목적 서버 대신 전용 설비를 사용해라.

결론

DNS cache poisoning은 초기 버전의 DNS 솔루션을 운영하고 있는 곳에는 큰 위협이 될 수 있다. 사용되고 있는 DNS 소프트웨어의 버전과 상관없이 부주의하게 DNS를 배치한 회사들에게도 위협은 확대될 수 있다. DNS 디바이스들의 버전 관리와 안전한 설정에 대해 주의 깊은 관심을 보인다면 cache poisoning의 위협은 어느 정도 무난한 수준까지 줄여들 것이다.

참고문헌

Hyatt, M. (2005, June). *Five steps to minimize DNS cache poisoning.* Retrieved March 14, 2006 from http://searchwindowssecurity.techtarget.com/tip/1,289483,sid45_gci1101998,00.html

Hyatt, R. (2006, January). Keeping DNS trustworthy. *The ISSA Journal*. January 2006, p. 37-38.

Microsoft TechNet (2005, January). *How DNS query works*. Retrieved March 14, 2006 from http://technet2.microsoft.com/WindowsServer/en/Library/1fd5b3be-ca29-43d0-b5e2-8a65192d5b781033.mspx

[18] http://romana.now.ie/james/tsig.html

[19] http://www.answers.com/topic/dynamic-dns-update?method=22