DNS缓存投毒攻击原理与防御策略

靳冲[1,2]，郝志宇[1]，吴志刚[1]
[1]中国科学院计算技术研究所，北京100190
[2]中国科学院研究生院，北京100190

摘　要：DNS是Internet最重要的基础设施之一，若遭受攻击将影响Internet的正常运转，因而其安全性备受关注。本文分析了传统以及新型（Kaminsky）DNS缓存投毒（Cache Poisoning）攻击的原理，给出攻击实例，详细描述了攻击流程，验证了DNS缓存投毒攻击潜在的危害性，并提出若干防御策略。

关键词：DNS；缓存投毒；Kaminsky攻击

Principles and Defense Strategies of DNS Cache Poisoning

Jin Chong[1,2], Hao Zhiyu[1], Wu Zhigang[1]
[1]Institute of Computing Technology of Chinese Academy of Sciences, Beijing 100190
[2]Graduate University of Chinese Academy of Sciences, Beijing 100190

Abstract: DNS is one of the most important basic infrastructures of the Internet, attacks on which will prevent the Internet from working properly. Therefore, its security is receiving great concern. This paper analyzes the principles of both traditional and novel (Kaminsky) DNS Cache Poisoning, presents attack samples and describes the whole attack process in detail. After verifying the potential harm of DNS Cache Poisoning, the paper gives out several defense strategies.
Key Words: DNS; cache poisoning; Kaminsky attack

I. 概述

DNS（Domain Name System）是一个多层次的分布式数据库系统，其基本功能是完成域名解析，即提供域名和IP地址之间的映射关系，为互联网（Internet）用户提供便利。DNS是Internet的基础，也是目前互联网上最成功的应用之一，其安全性备受关注。DNS缓存投毒（DNS Cache Poisoning）是DNS攻击中危害较大的一种，也是当前DNS攻击领域的研究热点。它通过使用虚假IP地址信息替换名字服务器缓存中主机记录的真实IP地址信息来制造破坏[6,7]。本文介绍了传统DNS缓存投毒攻击以及新型的Kaminsky攻击的基本原理，并提出了可行的防御策略。

II. 传统DNS缓存投毒攻击

一台 DNS 服务器只记录本地资源的所有授权主机，若想查询非本地的主机信息，则要向信息持有者（权威 DNS 服务器）发送查询请求。为了避免每次查询都发送请求，DNS 服务器会把权威 DNS 服务器返回的查询结果保存在缓存中，并保持一定时间，这就构成了 DNS 缓存（DNS Cache）。DNS 缓存投毒攻击就是通过污染 DNS Cache，用虚假的 IP 地址信息替换 Cache 中主机记录的真实 IP 地址信息来制造破坏。

2.1 DNS解析过程

在分析 DNS 缓存投毒攻击原理之前，先阐述一下 DNS 解析过程。假设要查询的域名为 www.ict.ac.cn，并假设客户端和首选的缓存名字服务器（即缓存 DNS 服务器）满足以下条件：

（1）首选缓存名字服务器和客户端首次启动，没有本地缓存信息。

（2）首选缓存名字服务器不是目标域名的权威名字服务器。

查询过程如图 1 所示，步骤如下：

（1）客户端向首选缓存名字服务器发起递归查询 www.ict.ac.cn 的请求。

（2）首选缓存名字服务器检查本地资源记录，若存在则作授权回答；若不存在，则检查本地缓存，如存在则直接给客户端返回结果。若本地资源记录和缓存中都不存在，则向根名字服务器发起迭代查询。根据假设条件，本例查询中将向根名字服务器继续查询。

（3）根名字服务器返回 cn 域的权威名字服务器（即顶级域名服务器）的地址，首选缓存名字服务器继续向顶级域名服务器发起迭代查询。

（4）顶级域名服务器返回 ac.cn 域的权威名字服务器地址，首选缓存名字服务器继续向该地址请求迭代查询。如此继续，直到得到 www.ict.ac.cn 的授权回答，保存在本地缓存中，并返回给客户端，完成此次查询。

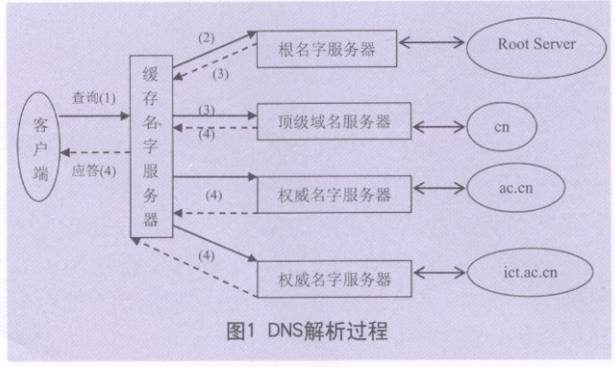

图1 DNS解析过程

2.2 DNS 报文格式

DNS 定义了用于查询和应答的报文格式，通常采用 UDP 协议传输查询请求和应答数据包。表 1（a）显示了 DNS 查询报文格式，表 1（b）显示了 DNS 应答报文格式。

表1(a) DNS查询报文

标识	标志
问题数	资源记录数
授权资源记录数	附加资源记录数
查询问题	

表1(b) DNS应答报文

标识	标志
问题数	资源记录数
授权资源记录数	附加资源记录数
查询问题	
回答资源记录（可变）	
授权资源记录（可变）	
附加资源记录（可变）	

2.3 传统DNS缓存投毒攻击原理

目前 DNS 采用 UDP 协议传输查询和应答数据包，采用简单信任机制，对首先收到的应答数据包仅进行原查询包发送 IP 地址、端口和随机查询 ID 的确认，而不会对数据包的合法性做任何分析，若匹配，则接受其作为正确应答数据包，继续 DNS 解析过程，并丢弃后续到达的所有应答数据包。这就使得攻击者可以仿冒权威名字服务器向缓存 DNS 服务器发送伪造应答包，力争抢先完成应答以污染 DNS 缓存。若攻击者发送的伪造应答包在权威名字服务器发送的正确应答包之前到达缓存 DNS 服务器，并与原查询包 IP 地址、端口和随机查询 ID 相匹配，就能够成功污染 DNS 缓存。图 2 显示了传统 DNS 缓存投毒攻击的过程。

攻击者伪造的应答数据包（红色所示）若能先于正确应答数据包(黑色所示)到达缓存 DNS 服务器，并与原查询包发送 IP 地址、端口和随机查询 ID 相匹

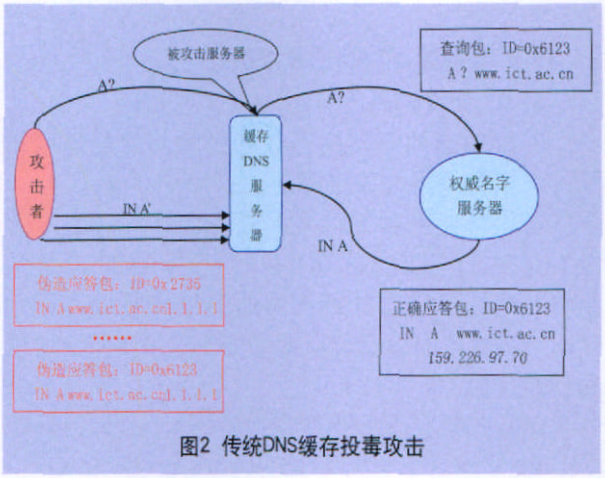

图2 传统DNS缓存投毒攻击

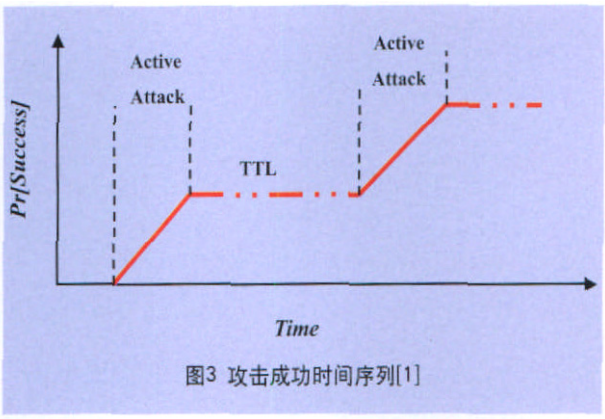

图3 攻击成功时间序列[1]

配，则将成功污染缓存 DNS 服务器的 Cache[10]，使得在该 Cache 有效期内，所有使用该缓存 DNS 服务器的用户对域名 www.ict.ac.cn 的查询请求都返回攻击者指定的 IP 地址 1.1.1.1。

2.4 传统DNS缓存投毒攻击的缺陷

图 3 显示的是传统 DNS 缓存投毒攻击成功的时间序列示意图 [1]。

其中，Active Attack 阶段表示：缓存 DNS 服务器的 Cache 中没有要查询域名主机的记录，因而向外发送查询数据包，从向该域名主机的权威名字服务器发送查询请求开始到收到应答数据包的时间(即如图 1 中过程（4))。若缓存 DNS 服务器的 Cache 中存在要查询域名主机的记录，则缓存 DNS 服务器将直接由 Cache 中读取相应信息，TTL 则表示 DNS Cache 中对已有记录信息的保持时间。

（1）如果 DNS Cache 中没有攻击目标域名的记录，则由传统 DNS 缓存投毒攻击原理可知，攻击者在 Active Attack 阶段内发动攻击，且伪造的应答数据包在 Active Attack 阶段内到达缓存 DNS 服务器，才有可能污染 Cache，使得攻击成功。

（2）如果 DNS Cache 中存在攻击目标域名的记录，即处于 TTL 时间段内，这时缓存 DNS 服务器将直接从 Cache 中读取信息，而不再进行向根名字服务器等的迭代查询，攻击者将不可能成功；这种情况下，攻击者要等该记录在 Cache 中失效，即 TTL 时间后，重新处于（1）中所述情况时，再发动攻击，才有可能成功。

当对同一域名持续攻击时，假设攻击开始前缓存 DNS 服务器的 Cache 中没有该域名记录，若在一个 Active Attack 阶段中攻击不成功，再组织有效攻击则需要有 TTL 的时间间隔。但是，Active Attack 阶段时间一般较短，而当前大多数缓存 DNS 服务器的 Cache TTL 设置的时间又较长，这就导致一定时间内可利用的攻击时间很少，大大降低了攻击成功的概率。这也就是传统 DNS 缓存投毒攻击最早在 1990 年即已出现，并且一旦攻击成功，危害极为严重，却一直没有引起广泛关注的原因。然而，新型的 Kaminsky 攻击克服了这一缺陷，大大提高了攻击成功率，因此一经公布就受到广泛重视。

III.KAMINSKY缓存投毒攻击

2008 年夏天，Dan Kaminsky 发现了一种新型 DNS 缓存投毒攻击，引起了网络安全界的广泛关注。该攻击方法克服了传统 DNS 缓存投毒攻击存在的攻击所需时间长、成功率很低的缺陷。

3.1 Kaminsky攻击原理

传统的 DNS 缓存投毒攻击，污染的目标是应答数据包中带有查询结果 IP 地址的回答资源记录部分（参见表 1 (b)），而 Kaminsky 攻击上升了一个层次，污染的目标是应答数据包中 Authority Records 部分（授权资源记录，参见表 1 (b)）。图 4 显示了 Kaminsky 攻击流程。

(1) 攻击者向被攻击的目标服务器发送一个 DNS 查询请求，该查询请求中的域名主机使用随机序列和目标域名的组合，如图 4 中的 www276930.ict.ac.cn,其中 ict.ac.cn 为目标域名,276930 是随机生成的序列。显然，这个查询的域名主机是不存在的，正常返回的应答数据包中回答资源记录部分应为 NXDOMAIN（表示该域名主机不存在）。

（2）被攻击目标服务器会按 2.1 节中所述 DNS

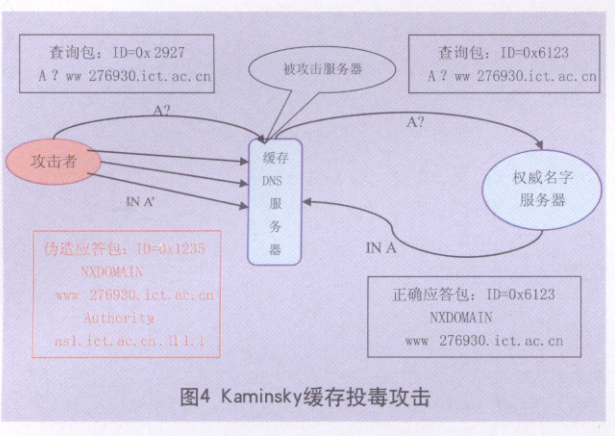

图4 Kaminsky缓存投毒攻击

解析过程进行查询，此时攻击者伪造应答数据包（红色所示）并发送给目标服务器，若该数据包能在正确应答包（黑色所示）之前到达目标服务器，并能成功匹配原查询包发送 IP 地址、端口和随机查询 ID，则攻击成功。

（3）在攻击者伪造的应答数据包中，回答资源记录部分与正确应答包一样（NXDOMAIN，表示该域名主机不存在），但是授权资源记录部分是 ns1.ict.ac.cn 的伪造 IP 地址 1.1.1.1，一旦攻击成功，该资源记录信息将被写入目标服务器的 Cache 中。在 Cache 保持时间内，对名字服务器 ns1.ict.ac.cn 管辖的所有域名的查询都将被发送到攻击者自己控制的 IP（1.1.1.1）中。

3.2 Kaminsky攻击的优势

（1）如 2.4 节所述，传统 DNS 缓存投毒攻击中，存在攻击所需时间长、成功率低的问题，而 Kaminsky 攻击克服了这一缺陷。Kaminsky 攻击中，对同一域名进行持续攻击，每次查询都会在目标域名上添加随机序列，这使得在目标 DNS Cache 中一般不存在各个构造域名主机的记录，因此若攻击不成功，则可以更换随机序列连续不断地进行攻击，不存在有效攻击时间的问题，也不存在攻击时间间隔，这将极大地节省攻击所需时间，有效提高攻击成功率。

（2）传统 DNS 缓存投毒攻击成功后，只是污染了目标 DNS Cache 中的一条主机记录，之后对这一个域名主机的查询将被发送到攻击者控制的 IP 地址。而 Kaminsky 攻击成功后，污染的是目标 DNS Cache 中一个域名主机的权威名字服务器的记录，之后对该名字服务器管辖的所有域名主机的查询都将被发送到攻击者控制的 IP 地址中，破坏力度远高于传统 DNS 缓存攻击。

3.3 缓存投毒攻击所需信息

3.3.1 所需信息

一个伪造的 DNS 应答数据包需要的信息包括 TCP/IP 层和 DNS 协议层两方面。

TCP/IP 层包括 srcip（权威名字服务器的 IP 地址）、dstip（被攻击的缓存 DNS 服务器 IP 地址）、sport（权威名字服务器使用的端口，通常为 53）、dport（被攻击的缓存 DNS 服务器发送请求报文时使用的端口）。

DNS 协议层包括查询域名、TXID（查询随机 ID，16bit）。

3.3.2 获取方式

srcip 通过查询要攻击域名的 NS 即可获得，可以用 dig、nslookup 等命令查询得到。sport 通常为 53。在公网上搜索可以得到要攻击 DNS 服务器的 IP 地址信息，从而获得 dstip。缓存投毒攻击为攻击者主动发送查询数据包，因此查询域名为攻击者自己构造，相当于已知，例如在 3.1 节中，查询域名为 276930.ict.ac.cn。

最难获得的是 dport 和 TXID 字段。dport 和 TXID 都是16bit字段，两者都有2^{16}(65536)种可能值。对于 TXID，只能进行猜测，在被攻击的缓存 DNS 服务器收到正确应答包之前，越多的伪造应答包到达，命中率越高。对于 dport 可以有如下几种获取方法：

（1）猜测。若 TXID 和 dport 均采用猜测法，那么每个伪造应答包命中的概率仅为 1/(65536x65536)，攻击成功难度较大。

（2）有些 DNS 服务器软件，如 Bind 的一些版本，在每次进程启动到停止之间，其发送查询数据包的

```
Last login: Wed Oct 28 08:58:16 2009 from 10.0.35.3
[root@localhost ~]# dig @10.0.15.231 www.dnstest.com.cn

; <<>> DiG 9.3.4-P1 <<>> @10.0.15.231 www.dnstest.com.cn
; (1 server found)
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 28316
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 0

;; QUESTION SECTION:
;www.dnstest.com.cn.            IN      A

;; ANSWER SECTION:
www.dnstest.com.cn.     3       IN      A       159.226.39.14

;; AUTHORITY SECTION:
dnstest.com.cn.         3       IN      NS      ns1.dnstest.com.cn.

;; Query time: 139 msec
;; SERVER: 10.0.15.231#53(10.0.15.231)
;; WHEN: Wed Oct 28 18:57:43 2009
;; MSG SIZE  rcvd: 70
```
图5 查询配置域名

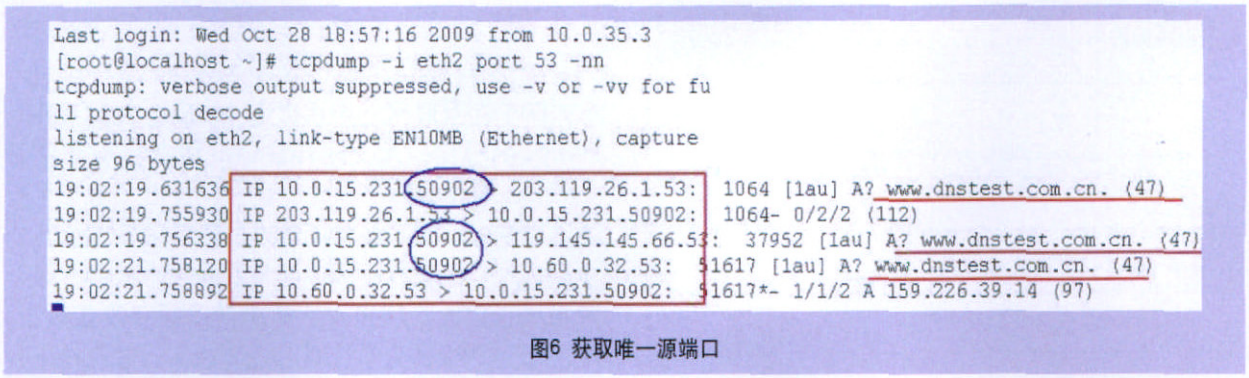

图6 获取唯一源端口

源端口都是唯一的[3, 4]。对于上述DNS服务器，可以通过其他手段获得其源端口，即伪造应答包的dport。例如，通过porttest.dns-oarc.net就可得知哪些DNS服务器存在源端口唯一的漏洞；若再掌握一台权威名字服务器，通过捕包即可获得存在漏洞的DNS服务器的源端口号。具体方法如下：

在一台可控DNS服务器上申请域名，例如：dnstest.com.cn，并为其配置一个权威名字服务器，例如：ns1.dnstest.com.cn。用dig命令发送查询域名dnstest.com.cn的数据包，如图5所示，其中10.0.15.231是存在源端口唯一漏洞的被攻击服务器IP。

发送查询包的同时，在配置的权威名字服务器ns1.dnstest.com.cn上，用tcpdump等工具捕包，即可获得被攻击目标10.0.15.231的源端口号。如图6所示。

至此，即可获得缓存投毒攻击的所需全部信息。

3.4 Kaminsky缓存投毒攻击实例

下面记录了一次Kaminsky攻击前后的全过程。

攻击目标DNS服务器：10.0.15.231，攻击目标域名：dnstest.com.cn。Kaminsky攻击前，在目标DNS服务器的缓存中，对域名dnstest.com.cn的查询结果如图7。

可见，缓存中没有关于域名dnstest.com.cn的记录信息。进行Kaminsky攻击，如图8所示。

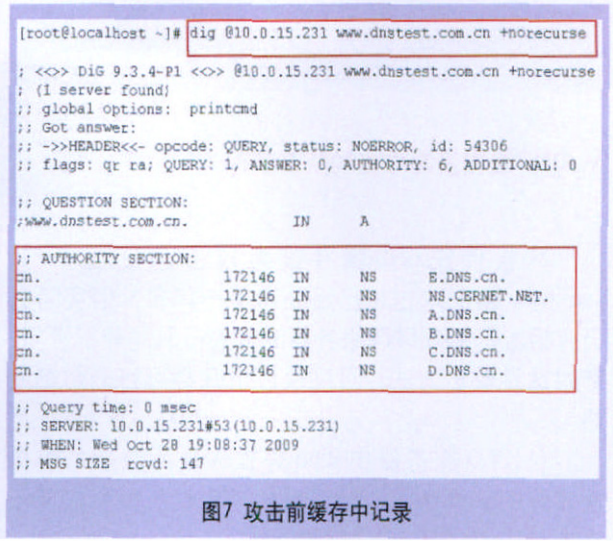

图7 攻击前缓存中记录

Kaminsky攻击结束后，在目标DNS服务器的缓存中，对域名dnstest.com.cn的查询结果如图9。

3.5 缓存投毒攻击存在的问题

对于源端口随机性比较好的DNS服务器[9]，需要同时猜测TXID和dport字段，攻击成功难度较大。

据缓存投毒攻击原理，对已在目标DNS Cache中的域名进行攻击不能成功。

一个域名通常会有不止一台权威名字服务器，对这样的域名查询时，会从多台权威名字服务器中随机选取一台进行应答，而攻击者无法估计选中的是哪一台，这就增加了攻击难度。

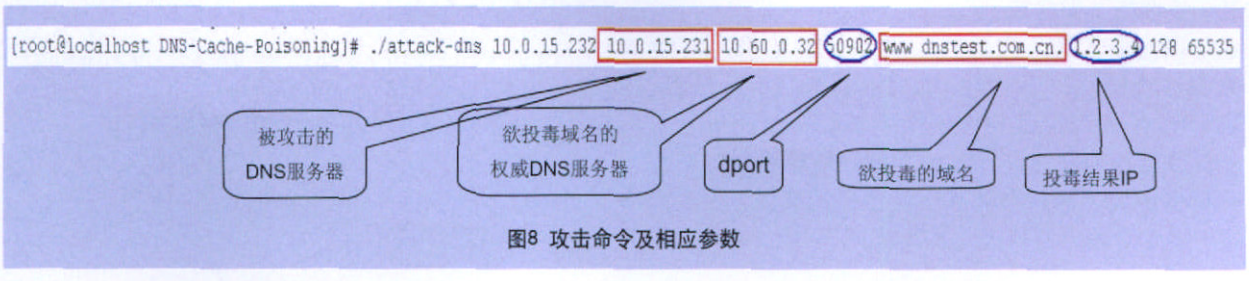

图8 攻击命令及相应参数

```
[root@localhost ~]# dig @10.0.15.231 www.dnstest.com.cn +norecurse

; <<>> DiG 9.3.4-P1 <<>> @10.0.15.231 www.dnstest.com.cn
; (1 server found)
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 63484
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 0

;; QUESTION SECTION:
;www.dnstest.com.cn.            IN      A

;; ANSWER SECTION:
www.dnstest.com.cn.     3       IN      A       1.2.3.4

;; AUTHORITY SECTION:
dnstest.com.cn.         3       IN      NS      ns1.dnstest.com.cn.

;; Query time: 1 msec
;; SERVER: 10.0.15.231#53(10.0.15.231)
;; WHEN: Thu Oct 29 11:32:08 2009
;; MSG SIZE  rcvd: 70
```

图9 攻击成功后缓存中记录

IV. DNS缓存投毒攻击的防御策略

尽管传统DNS缓存投毒攻击已出现多年，Kaminsky漏洞攻击也已经公布了一年多，但实际上仍有相当一部分DNS服务器没有进行漏洞补丁修复。针对这种攻击方式，可采取以下几种可行的防御策略：

(1)DNS服务器中Bind等软件采用源端口随机性较好的较高版本。源端口的随机性可以有效降低攻击成功的概率，增加攻击难度。

(2)增加权威名字服务器的数量。据调查，国际和国内在权威名字服务器部署的数量方面近几年均有所提升[5]，但应进一步加强。

(3)在现有DNS协议框架基础上，引入一些技巧性方法，增强DNS安全性。如在对DNS应答数据包的认证方面，除原查询包发送IP地址、端口和随机查询ID外，再增加其他可认证字段，增强认证机制。David Dagon等[2]就曾提出增加对查询域名大小写认证的算法，使DNS服务器在收到应答数据包后，在原认证基础上，再利用该算法对应答包与原查询包中查询域名大小写进行比对，若匹配，则说明是正确应答包，否则可能为攻击者伪造应答包，进一步增强了安全性。

(4)改进现有DNS协议框架，例如在DNS服务器上配置DNSSEC[8]，或引入IPv6协议机制。DNSSEC是专门保证DNS安全的机制，试图提升对应答数据包的弱认证方式以提高DNS安全性。建立在IPv6协议上面的DNS是一种全新的模式，目前尚处于理论阶段，有待进一步实施。

V. 结束语

DNS在互联网上应用广泛，其安全性关系整个Internet的稳定。本文对传统DNS缓存投毒和新型Kaminsky攻击进行了讨论分析，并提出了若干可行的防御策略。DNS缓存投毒攻击还有很多未公开的应用领域，危害性很大，若与其他技术结合，破坏性更强。因此，从根本上解决DNS缓存投毒攻击，全面加强DNS的安全，具有重要意义，这也是下一步研究的重点。 中国通信

（英文全文请参见75页）

参考文献

[1]D.Dagon, M.Antonakakis, K.day, X.Luo, C.P.Lee, W.Lee: Recursive DNS Architectures and Vulnerability Implications. In Proceeding of The 16th Annual Network and Distributed System Security Symposium(NDSS 2009), San Diego, CA, February 2009.

[2]D.Dagon, M.Antonakakis, P.Vixie, T.Jinmei, and W.Lee: Increased DNS Forgery Resistance Through 0x20-Bit Encoding. In Proceedings of the 15th ACM Conference on Computer and Communications Security (CCS 2008), October 2008.

[3]Amit Klein: BIND 9 DNS Cache Poisoning. http://www.trusteer.com/docs/bind9dns.html, 2007.

[4]Amit Klein: BIND 8 DNS Cache Poisoning. http://www.trusteer.com/docs/bind8dns.html, 2007.

[5]王圭. 域名系统安全性研究[D]. 博士学位论文，哈尔滨工业大学，2007.

[6]J.Stewart: DNS Cache Poisoning—The Next Generation [EB/OL]. (2007-08-25). http://www.secureworks.com/research/articles/dns-cache-poisoning/.

[7]T.Olzak: DNS Cache Poisoning: Definition and Prevention [EB/OL]. (2006-03-02). http://www.infosecwriters.com.

[8]M.Andrews: The dnssec lookaside validation(dlv) dns resource record, rfc 4431. http://tools.ietf.org/html/rfc4431, 2006.

[9]“Internet Systems Consortium BIND 9.4.1” (Internet Systems Consortium web page). http://www.isc.org/index.pl?/sw/bind/view/?release=9.4.1

[10]Ketil Froyn: “DNS Poisoning” (demonstration web page), 2003. http://ketil.froyn.name/poison.html.

Principles and Defense Strategies of DNS Cache Poisoning

Jin Chong[1, 2], Hao Zhiyu[1], Wu Zhigang[1]
[1]Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190
[2]Graduate University of Chinese Academy of Sciences, Beijing 100190

Abstract: DNS is one of the most important basic infrastructures of the Internet, attacks on which will prevent the Internet from working properly. Therefore, its security is receiving great concern. This paper analyzes the principles of both traditional and novel (Kaminsky) DNS cache poisoning, presents attack samples and describes the whole attack process in detail. After verifying the potential harm of DNS cache poisoning, the paper gives out several defense strategies.
Key words: DNS; cache poisoning; Kaminsky attack

I. INTRODUCTION

DNS (Domain Name System) is a multi-level distributed database system, whose basic function is to complete name resolution, namely providing a mapping relation between domain names and IP addresses, and therefore it brings much convenience to Internet users. DNS is the foundation and also one of the most successful applications of the Internet, and its security has drawn a lot of attention. DNS cache poisoning is one of the most hazardous methods in DNS attacks, which is also a hot topic in the research area. DNS cache poisoning causes damages by replacing the real IP address information of hosts' records in the cache of DNS server with the fake IP address information [6, 7]. This paper introduces the basic principles of both traditional and novel (Kaminsky) DNS cache poisoning and offers several defense strategies.

II. TRADITIONAL DNS CACHE POISONING

A DNS server only records all local authorized hosts. If the information of a host is to be queried, which is not local, a request message should be sent to the information holder (authoritative name server). To avoid the repetition for deliveries of such requests, the DNS server will save the results returned by the authoritative name server in the cache for a certain time, which **constitute** the DNS cache. By contaminating the DNS cache, namely replacing the real IP address information of hosts' records in the cache with the fake IP address information, DNS cache poisoning causes damages.

2.1 DNS Resolution

We briefly introduce the DNS resolution here before analyzing the principles of DNS cache poisoning, assuming that the domain name we are about to query is *www.ict.ac.cn*, namely the target domain name, and that the client as well as the default caching name server, namely default DNS server, meet the following conditions:

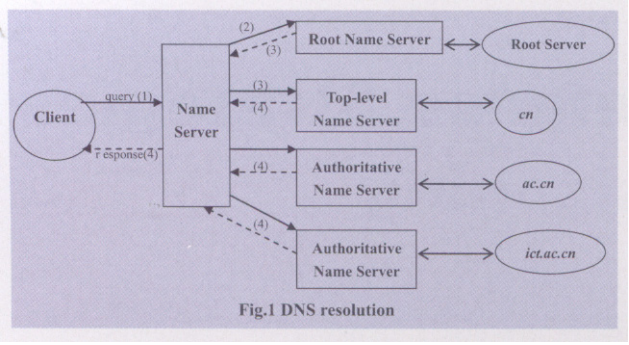

Fig.1 DNS resolution

(1) The client and the default caching name server start up for the first time so that there is no local cache information.

(2) The default caching name server is not the authoritative name server of the target domain name.

Figure 1 shows the query process. It works as follows:

(1) The client sends a request of recursive query of *www.ict.ac.cn* to the default caching name server.

(2) The default caching name server checks its local records. If there is such information, it returns the authoritative answer; else, it will check its local cache and return the result to the client if there exists. If there is no such record in the cache either, it will send an iterative query to a root name server. According to the assumption above, in this example an iterative query is sent.

(3) The root name server returns the address of the authoritative name server of "*cn*" domain, namely the top-level name server, and the default caching name server keeps sending the iterative query to the top-level name server.

(4) The top-level name server returns the address of the authoritative name server of "*ac.cn*" domain. The default caching name server keeps querying until getting the authoritative answer of *www.ict.ac.cn*. After that, it saves the authoritative answer in its local cache and returns the authoritative answer to the client, finishing this query process.

2.2 DNS Message Format

DNS protocol defines message formats for both queries and responses. Most of the time it uses UDP protocol to transfer the data packets. Table 1(a) shows the message format of DNS queries, and Table 1(b) shows the message format of DNS responses.

2.3 Principles of Traditional DNS Cache Poisoning

Currently, DNS adopts simple trust mechanism, which only verifies the destination IP address, the destination port and the random query ID of the received response data packet, and never makes any analysis of the legality of the response data packet. If the three fields above match, the response packet will be accept as the real response data packet. The DNS server will go on with the rest of the DNS resolution process and ignoring all the response packets that arrive later. Therefore, an attacker could pretend to be the authoritative name server and send fake response packets. If one of the fake response packets arrives at the DNS server before the real one, sent by the authoritative name server, and the three fields of the fake packet match the source IP address, the source port and the random query ID of the original query packet, the attacker could successfully contaminate the cache of the DNS server. Figure 2 shows the process of traditional DNS cache poisoning.

Table 1(a): message format of DNS queries

identification	flags
number of questions	number of answer RRs
number of authority RRs	number of additional RRs
questions	

Table 1 (b): message format of DNS responses

identification	flags
number of questions	number of answer RRs
number of authority RRs	number of additional RRs
questions	
answer (variable number of resource records)	
authority (variable number of resource records)	
additional information (variable number of resource records)	

In Figure 2, if one of the fake response data packets (the red ones) could arrive at the DNS server before the real one (the black one), and the three fields match [10], the attacker would succeed. After that, all the later queries for *www.ict.ac.cn* by the users of the attacked DNS server would be replied with the address *1.1.1.1*, which is actually designated by the attacker.

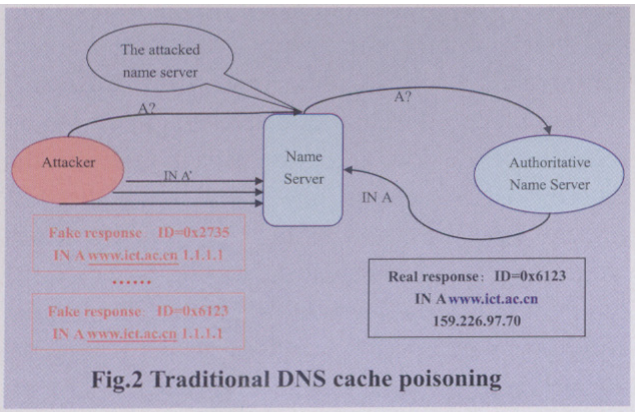

Fig.2 Traditional DNS cache poisoning

2.4 Deficiencies of Traditional DNS Cache Poisoning

Figure 3 shows the cumulative probability over time for an attacker to poison a DNS cache [1].

In Figure 3, the period of Active Attack is the time interval between the DNS server sending the query request packet to the domain name's authoritative name server and receiving the response data packet (namely Step (4) in Figure 1). TTL is the limit on the time period of keeping the existing information in the DNS cache.

(1) If there is no record of the target domain name in the DNS cache, according to the principles of traditional DNS cache poisoning, only if the attacker starts the attack during an Active Attack period and the fake response packet reaches the DNS server in this period could he contaminate the cache and attack successfully.

(2) If there is the record of the target domain name in the DNS cache, namely during the TTL period, the DNS server will read the record directly from the cache and no longer send the iterative request to the root name server and thus the attacker

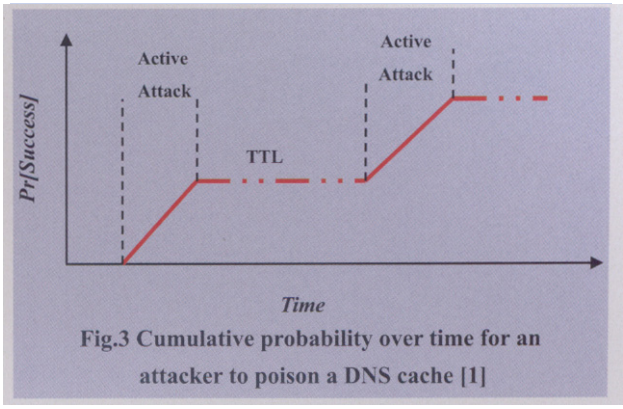

Fig.3 Cumulative probability over time for an attacker to poison a DNS cache [1]

will not succeed. In this case, the attacker has to wait until the record in the cache to be invalid (not more than one TTL) to meet the situation described in 1), if he wants to conduct a successful attack.

However, the Active Attack period is usually relative short while the TTL in most current DNS servers is set to be relatively long, which leads to that the available periods for effective attacks are quite short and thus the success possibility is low. That's why there is not so much attention paid to traditional DNS cache poisoning consistently, although it early turned up in 1990 and is very hazardous if the attack succeeds. However, the novel Kaminsky attack makes up for the deficiencies and increases the success possibility dramatically so that it has been paid much attention since announced.

III. KAMINSKY ATTACK OF DNS CACHE POISONING

In the summer of 2008, Dan Kaminsky announced a novel attack of DNS cache poisoning, which drew wide attention in the area of network security.

3.1 Principles of Kaminsky Attack

The contamination target in traditional DNS cache poisoning is the answer resource records which contain the IP addresses in the response data packet (Table 1(b)). However, the Kaminsky attack is upgraded, whose contamination target is the authority resource records in the response data

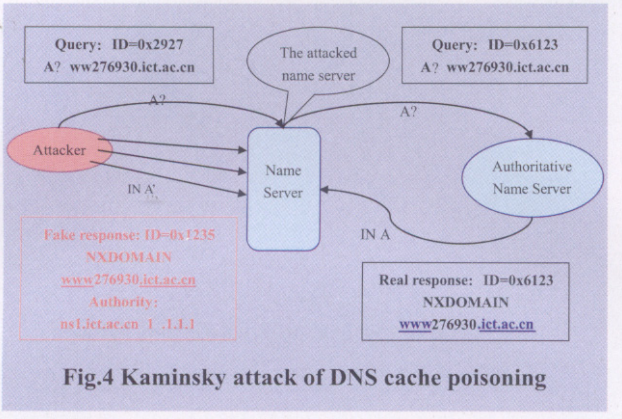

Fig.4 Kaminsky attack of DNS cache poisoning

packet (Table 1(b)). Figure 4 shows the process of the Kaminsky attack.

(1) The attacker sends a DNS query to the name server to be attacked, in which the hostname is a combination of the target domain name and a random series, namely *www276930.ict.ac.cn* in Figure 4, in which *ict.ac.cn* is the target domain name and *276930* is the random series. Obviously the queried hostname doesn't exist and the answer resource record in the response data packet returned should normally be NXDOMAIN (namely the hostname doesn't exist).

(2) The attacked name server will make the query according to the DNS resolution described in 2.1. At this moment, the attacker fakes response data packets (e. g. the red packet in Figure 4) and sends them to the attacked name server. If one fake data packet arrives at the attacked name server prior to the real response data packet (the black packet in Figure 4) and its destination IP address, destination port and random query ID successfully match the corresponding fields of the original query packet, the attacker succeeds.

(3) The answer resource record in the fake response data packet is the same as that in the real response data packet (NXDOMAIN, which means the hostname doesn't exist), but the authority resource record is the fake IP address of *ns1.ict.ac.cn* (*1.1.1.1*). Once the attacker succeeds, the authority resource record will be written into the cache of the attacked name server. During the time that the cache is valid, all queries of the domain names belonging to the name server *ns1.ict.ac.cn* will be sent to the IP address (*1.1.1.1*) which is actually designated by the attacker.

3.2 Advantages of Kaminsky Attack

(1) As mentioned in 2.4, there are several deficiencies in traditional DNS cache poisoning, namely the long time needed for a successful attack and the low success possibility, however, the Kaminsky attack makes up for such deficiencies. In the Kaminsky attack, when continuously attacking a domain name, random series will be appended to the target domain name in each query, and thus there are no records for the domain name of each query in the attacked DNS cache. If one attack is not successful, the attacker could replace the random series with a new one and repeat the attack, in which the problem that the effective time in attacks is quite short no longer exists. In this way, the Kaminsky attack saves the attack time dramatically and increases the success possibility effectively.(2) If the traditional DNS cache poisoning succeeds, it only contaminates a hostname record in the attacked DNS cache. After that, only the query for the same hostname will be sent to the IP address designated by the attacker. However, if the Kaminsky attack succeeds, it contaminates the domain name record of an authoritative name server in the attacked DNS cache, and then queries of all hostnames administered by that authoritative name server will be sent to the IP address designated by the attacker. Therefore, the damage caused by Kaminsky attack is much worse.

3.3 Information Needed for DNS Cache Poisoning Attack

3.3.1 Information Needed

Information needed in faking DNS response data packet should include two aspects: TCP/IP layer and DNS layer.

In TCP/IP layer, it should include *srcip* (IP address of the authoritative name server), *dstip* (IP address of the DNS server to be attacked), *sport* (port used by the authoritative name server, usually 53), and *dport* (port used when the attacked DNS server sends queries).

In DNS layer, it should include the queried domain name and TXID (query random ID, 16bit).

3.3.2 Acquisition Method

Srcip can be acquired through querying the NS of the target domain name, which can be obtained by commands such as *dig*, *nslookup* etc. Usually *sport* is 53. IP address information of the attacked DNS server can be searched in public network, and thus we can acquire *dstip*. The attacker of DNS cache poisoning sends a query data packet on his own initiative, and the queried domain name is made by the attacker himself, namely its value is known. For example, in 3.1, the queried domain name is *www276930.ict.ac.cn*.

The fields of *dport* and TXID are most difficult to be acquired. Both *dport* and TXID are 16bit fields, and they all have 2^{16} (65536) possible values. TXID can only be guessed. Before the attacked DNS server receives the real response packet, the more fake response data packets arrive, the higher the possibility of hit is. For *dport*, we have the following methods to acquire:

(1) Guess. If both TXID and *dport* are guessed, the possibility of hit for each fake response packet is only 1/ (65536*65536), which means the success rate of attack is very low.

(2) For some DNS server soft wares, such as several versions of *Bind*, during the period from starting up to shutting down of the process, the source port of sending query data packets is unchanging [3, 4]. For these DNS servers, we can acquire their source ports (*dport*) by some methods. For example, we could know what DNS servers have the loophole of unchanging source port on

```
Last login: Wed Oct 28 08:58:16 2009 from 10.0.35.3
[root@localhost ~]# dig @10.0.15.231 www.dnstest.com.cn

; <<>> DiG 9.3.4-P1 <<>> @10.0.15.231 www.dnstest.com.cn
; (1 server found)
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 28316
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 0

;; QUESTION SECTION:
;www.dnstest.com.cn.            IN      A

;; ANSWER SECTION:
www.dnstest.com.cn.     3       IN      A       159.226.39.14

;; AUTHORITY SECTION:
dnstest.com.cn.         3       IN      NS      ns1.dnstest.com.cn.

;; Query time: 139 msec
;; SERVER: 10.0.15.231#53(10.0.15.231)
;; WHEN: Wed Oct 28 18:57:43 2009
;; MSG SIZE  rcvd: 70
```

Fig.5 Query configured domain name

porttest.dns-oarc.net, and if we can control an authoritative name server in addition, we can acquire the source port by sniffing.

To be more specific, we can firstly ask for a domain name in a controlled DNS server, such as *dnstest.com.cn*, and configure an authoritative name server for it, such as *ns1.dnstest.com.cn*. Then we can send a query data packet of domain name *dnstest.com.cn* by *dig* command, as shown in Figure 5, in which *10.0.15.231* is the IP address of the DNS server to be attacked with the loophole of unchanging source port.

By sending the query packet, we can acquire the source port of the attacked DNS server *10.0.15.231* in configured authoritative name server with the help of some sniffing tools such as *tcpdump*, as shown in Figure 6.

Hence, we acquire all the information needed by the cache poisoning attack.

3.4 An Example of Kaminsky Cache Poisoning

We conducted an experiment of a Kaminsky attack.

```
Last login: Wed Oct 28 18:57:16 2009 from 10.0.35.3
[root@localhost ~]# tcpdump -i eth2 port 53 -nn
tcpdump: verbose output suppressed, use -v or -vv for fu
ll protocol decode
listening on eth2, link-type EN10MB (Ethernet), capture
size 96 bytes
19:02:19.631636 IP 10.0.15.231.50902 > 203.119.26.1.53:  1064 [1au] A? www.dnstest.com.cn. (47)
19:02:19.755930 IP 203.119.26.1.53 > 10.0.15.231.50902:  1064- 0/2/2 (112)
19:02:19.756338 IP 10.0.15.231.50902 > 119.145.145.66.53:  37952 [1au] A? www.dnstest.com.cn. (47)
19:02:21.758120 IP 10.0.15.231.50902 > 10.60.0.32.53:  51617 [1au] A? www.dnstest.com.cn. (47)
19:02:21.758892 IP 10.60.0.32.53 > 10.0.15.231.50902:  51617*- 1/1/2 A 159.226.39.14 (97)
```

Fig.6 Acquire the exclusive source port

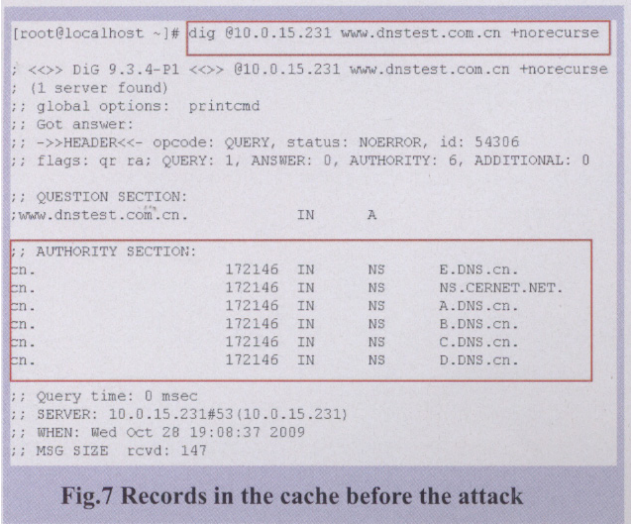

Fig.7 Records in the cache before the attack

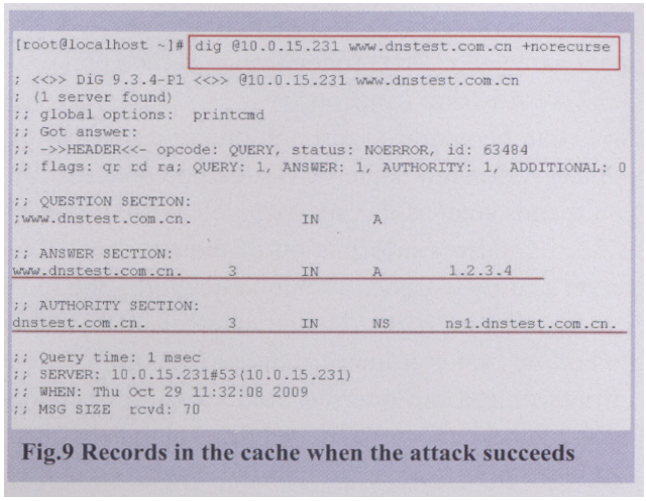

Fig.9 Records in the cache when the attack succeeds

The DNS server to be attacked was *10.0.15.231* and the target domain name was *dnstest.com.cn*. Before the Kaminsky attack started, the query result of the domain name *dnstest.com.cn* in the cache of the attacked DNS server is shown in Figure 7.

It can be found that there was no record of *dnstest.com.cn* in the cache.

Then we started the Kaminsky attack, as shown in Figure 8.

After the Kaminsky attack, the query result of the domain name *dnstest.com.cn* in the cache of the attacked DNS server is shown in Figure 9.

3.5 Problems in DNS Cache Poisoning Attack

(1) For DNS servers with better randomness in source port selection [9], we need to guess the fields of both TXID and *dport* at the same time, and therefore the attack is difficult to succeed.

(2) According to the principles of cache poisoning attack, the attack aiming at domain names which already exist in the DNS cache will not succeed.

(3) Usually a domain name has more than one authoritative name servers. For the query of such domain names, one of the authoritative name servers will be randomly chosen to answer the query, which can't be known by the attacker. Hence, the attacker is more difficult to succeed.

IV. DEFENSE STRATEGIES OF DNS CACHE POISONING

Although it has been several years since traditional DNS cache poisoning attack first appeared, and the Kaminsky loophole was announced more than one year ago, there are still a number of DNS servers that haven't applied any patch in practice. For such attacks, the following defense strategies could be adopted:

(1) Softwares such as *Bind* to be installed in DNS servers should choose higher versions with better randomness in source port selection. Randomness in source port selection can effectively decrease the possibility of successful attacks.

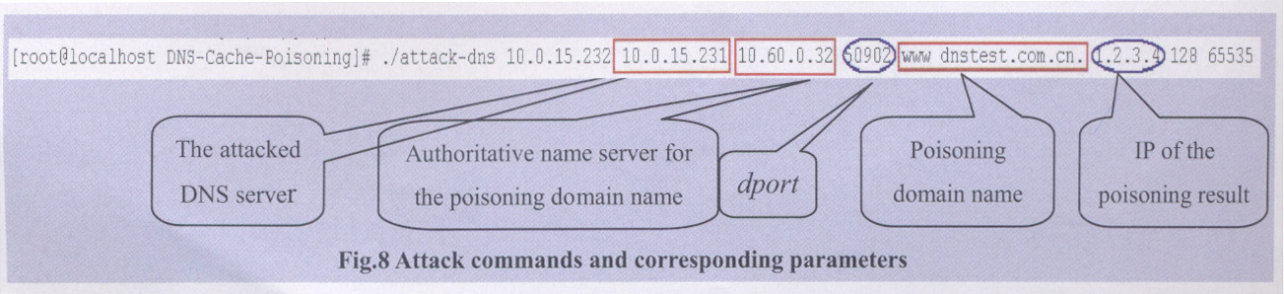

Fig.8 Attack commands and corresponding parameters

(2) Increase the number of authoritative name servers. According to surveys, there are more authoritative name servers deployed all over the world in recent years [5]. However, the number still needs to be larger.

(3) Bring in some technical methods for the existing DNS protocol framework to strengthen DNS security. For example, in the aspect of certificating DNS response packets, we could check more fields besides the IP address, the port and the random query ID. In [2], on the basis of original certification, the authors mixed the upper and lower case spelling of the domain name in the query packets to make DNS queries more resistant to poisoning attacks.

(4) Improve the existing DNS protocol framework such as configuring DNSSEC [8] in DNS servers and bringing in IPv6. DNSSEC is a mechanism that ensures DNS security, which tries to improve DNS security by strengthening the weak certification method for response packets. DNS established upon IPv6 is a brand-new model, which is still in the theoretical stage and waiting for further application.

V. CONCLUSION

DNS is widely applied on the Internet and its security influences the whole Internet. This paper introduced both traditional and novel (Kaminsky) DNS cache poisoning, and presented several defense strategies. There are still many unpublicized areas in which DNS cache poisoning could be applied and do great harm. Once combined with other technologies, DNS cache poisoning could cause much sever destructivity. Therefore, it is very meaningful to solve the problem radically and fully strengthen DNS security, on which our further research is also focusing. 中兴通信

(See page 17 for the Chinese version of this article.)

References

[1] D.Dagon, M.Antonakakis, K.day, X.Luo, C.P.Lee, W.Lee: Recursive DNS Architectures and Vulnerability Implications. In Proceeding of The 16th Annual Network and Distributed System Security Symposium(NDSS 2009), San Diego, CA, February 2009.

[2] D.Dagon, M.Antonakakis, P.Vixie, T.Jinmei, and W.Lee: Increased DNS Forgery Resistance Through 0x20-Bit Encoding. In Proceedings of the 15th ACM Conference on Computer and Communications Security (CCS 2008), October 2008.

[3] Amit Klein: BIND 9 DNS Cache Poisoning. http://www.trusteer.com/docs/bind9dns.html, 2007.

[4] Amit Klein: BIND 8 DNS Cache Poisoning. http://www.trusteer.com/docs/bind8dns.html, 2007.

[5] Wang Gui. Research On Domain Name System Security[D]. Dissertation for the Doctoral Degree. Harbin Institute of Technology. 2007

[6] J.Stewart: DNS Cache Poisoning—The Next Generation [EB/OL]. (2007-08-25). http://www.secureworks.com/research/articles/dns-cache-poisoning/.

[7] T.Olzak: DNS Cache Poisoning: Definition and Prevention [EB/OL]. (2006-03-02). http://www.infosecwriters.com.

[8] M.Andrews: The dnssec lookaside validation(dlv) dns resource record, rfc 4431.
http://tools.ietf.org/html/rfc4431, 2006.

[9] "Internet Systems Consortium BIND 9.4.1" (Internet Systems Consortium web page).
http://www.isc.org/index.pl?/sw/bind/view/?release=9.4.1

[10] Ketil Froyn: "DNS Poisoning" (demonstration web page), 2003.
http://ketil.froyn.name/poison.html.