一种基于拟态安全防御的 DNS 框架设计

王禛鹏,扈红超,程国振
(国家数字交换系统工程技术研究中心(NDSC),河南郑州 450003)

摘 要: 目前针对 DNS 服务器的恶意攻击频发,如 DNS 缓存投毒攻击,而 DNS 安全拓展协议(DNSSEC)在大规模部署时仍面临许多难题. 本文提出一种简单易部署的,具有入侵容忍能力的主动防御架构——拟态 DNS(Mimic DNS,M-DNS)——保证 DNS 安全. 该架构由选调器和包含多个异构 DNS 服务器的服务器池组成. 首先选调器动态选取若干服务器并行处理请求,然后对各服务器的处理结果采用投票机制决定最终的有效响应. 实验仿真表明,相比当前传统架构,M-DNS 可以降低缓存投毒攻击成功率约 10 个数量级.

关键词: DNS; DNS 缓存投毒攻击; 拟态安全防御; 动态异构冗余
中图分类号: TP393 **文献标识码:** A **文章编号:** 0372-2112 (2017)11-2705-10
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2017.11.018

A DNS Architecture Based on Mimic Security Defense

WANG Zhen-peng, HU Hong-chao, CHENG Guo-zhen
(*National Digital Switching System Engineering & Technological R&D Center, Zhengzhou, Henan* 450003, *China*)

Abstract: A simple and practical approach is required immediately to safeguard the Domain Name System (DNS) because there are increasing attacks on DNS (such as DNS cache poisoning) and various problems when deploying Domain Name System Security Extensions (DNSSEC) on a large scale. In this paper, we present Mimic DNS (M-DNS), a non-intrusive, tolerant and proactive security architecture, to deal with it. M-DNS is comprised of a scheduler and a server pool which consists of several heterogeneous DNS servers. The scheduler dynamically schedules the DNS servers to handle the requests in parallel and adopts the vote results from the majority of the servers to determine valid responses. Simulation results demonstrate that compared with current traditional frameworks, approximating 10 orders of magnitude reduction in cache poisoning attack probability is acquired when employing M-DNS.

Key words: DNS; DNS cache poisoning attack; mimic security defense; dynamic heterogeneous redundancy

1 引言

当前,DNS(Domain Name System)作为互联网最重要的基础设施之一[1],维护着域名体系和 IP 地址空间的相互映射关系,其重要性不言而喻. 然而,作为复杂的软件系统,其不可避免地存在未知漏洞和后门,易受攻击. 2016 年 3 月 11 日,国家信息安全漏洞共享平台网站发布安全公告①,指出 BIND 存在 3 个高危漏洞. 国家互联网应急中心公布的《2015 年中国互联网网络安全报告》②指出了针对 DNS 服务器的多起攻击行为. DNS 的安全形式不容乐观.

DNS 缓存服务器(又称作递归服务器,Recursive

收稿日期: 2016-10-28;修回日期: 2016-12-06;责任编辑:蓝红杰
基金项目: 国家自然科学基金青年基金(No. 61309020, No. 61602509); 国家自然科学基金创新群体项目(No. 61521003);国家重点研发计划项目(网络空间拟态防御技术机制研究)(No. 2016YFB0800100, No. 2016YFB0800101)

① 关于 ISC BIND 存在多个拒绝服务高危漏洞的安全公告,http://www.cnvd.org.cn/webinfo/show/3809,2016.
② 2015 年中国互联网网络安全报告,http://www.cert.org.cn/publish/main/upload/File/2015annualreport.pdf,2016,04.

Name Server,后文简写为 RNS)设于用户和权威服务器(Authoritative Name Server,后文简写为 ANS)之间,采用缓存机制提升整体服务效率. 然而,由于 DNS 协议的简单认证模式和"First Answer Wins"原则使其易受缓存投毒攻击(DNS cache poisoning),尤其是新型投毒攻击 Kaminsky①,使得 DNS 安全问题成为研究热点.

现有防御方案主要分为三类:一类是加密认证,其中典型代表就是 DNS 安全拓展协议 DNSSEC,由于会引入信任锚问题[2],以及分片攻击[3,4]、和放大攻击问题②,尤其是 RNS 的迭代查询会放大上述问题[5],使得 DNSSEC 部署进展缓慢,据《中国域名服务安全状况与态势分析报告 2015》③(后文简称为《报告》)数据显示,DNSSEC 在 RNS 中的部署率仅为 0.9%;第二类是异常检测,如文献[6,7]利用攻击前后 IP 熵值的变化实现检测,然而异常检测本身易引入"用户伪装入侵检测"问题[8],且防御方法较为被动;第三类是增加 DNS 报文的信息熵,使攻击者更难"猜中"真实报文. 如端口号随机化技术,在域名前添加随机字符串的 WSEC DNS[9],区别域名大小写的 0x20-bit[10],还有通过空间上多个查询[11,12]或时间上多次查询[13,14]增加整体的信息熵.

本文针对 RNS 的工作特点,提出了拟态 DNS 服务器(M-DNS)的安全架构,该架构基于拟态安全防御的动态异构冗余模型(Dynamic Heterogeneous Redundancy,DHR),利用多个服务器、多次查询(后文称之为预缓存机制)以及动态调度策略来增加整体的随机性,可以有效防御 DNS 缓存投毒攻击和未知漏洞后门.

2 M-DNS 架构

2.1 系统架构

如图 1 所示,M-DNS 架构由 DNS 池和选调器组成,DNS 池由若干运行不同 DNS 软件的服务器构成. 选调器由数据平面和控制平面组成,其中前者完成数据分流功能;后者负责执行体选择和判决,由选调模块、判决模块以及 DNS 服务器状态信息组成. 具体地,选调模块根据服务器状态信息动态选取若干作为活跃集,并向选调器数据平面下达分流指令. 选调器判决模块对收到的各个执行体的应答响应进行综合处理,将判决结果作为最终响应返回数据平面,并更新服务器状态信息.

2.2 M-DNS 选调器的安全防护

选调器作为 M-DNS 的核心组件,且位于系统的最前端,易受攻击. 本文采用"控制与转发分离"的原则保障选调器安全:其中数据平面位于选调器前端,可以设计为一种无状态的硬件转发功能,如采用 NetFPGA-10G④可编程板卡实现,使得攻击者难以攻击;而控制平面位于后端,虽然呈现为一种有状态维护的功能,易受

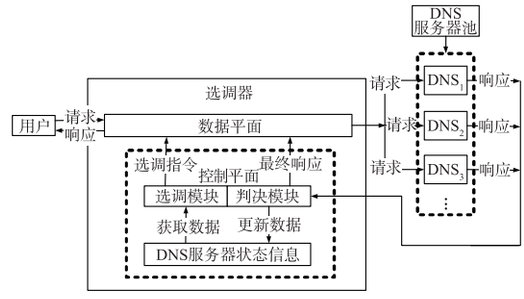


图1 M-DNS逻辑框架

攻击,但是对于外部用户不可见,实现了"状态隐藏"(state-masking),无法从外部对其实施有效攻击.

2.3 M-DNS 系统的安全分析

M-DNS 的异构冗余性增加了攻击者漏洞挖掘的成本,而动态性则增加了攻击者系统探测时的不确定性[15,16],并且破坏了系统突破时的连续性[17]. 这些特性也使得系统具备良好的入侵容忍能力,即使是超出了拟态防御边界的情况(半数以上执行体被攻击),当前被攻击的执行体在下一个周期可能变为非活跃状态,从而无法起到作用. 当然,对于半数以上执行体已被攻击成功的周期,M-DNS 可能无法提供可靠的服务,对此可以通过缩短切换周期的方式,降低攻击者攻击的有效时间以及产生影响的时间.

2.4 M-DNS 的可行性分析

动态异构冗余模型有以下前提假设:(1)系统输入输出一一对应;(2)正常情况下,异构执行体对同一输入有同一输出. 所以下面对 M-DNS 应用该模型时的问题进行解决说明.

(1)系统输入输出一一对应问题. 某些大型互联网公司或在内容分发网络中,为实现负载均衡,可能将一个域名对应为多个 IP. 例如,对某个域名的查询可能返回 IP^1,IP^2,IP^3 或 IP^2,IP^1,IP^3 等,对此可以将这 3 个 IP 视作一个集合,从而满足域名到 IP 地址集的一一对应.

(2)正常情况下一致输出问题. RNS 之间缓存内容不一致有两种可能的原因[11]:其一就是一个域名对应多 IP 的情况;其二是 DNS 的缓存机制造成的,对此可以将其作为一次缓存投毒攻击对待,予以重新查询.

---

① D. Kaminsky. It's the end of the cache as we know it http://www.blackhat.com/presentations/bh-jp-08/bh-jp-08-Kaminsky/BlackHat-Japan-08-Kaminsky-DNS08-BlackOps.pdf

② Security Bulletin: DNSSEC Amplification DDoS, https://www.akamai.com/cn/zh/multimedia/documents/state-of-the-internet/dnssec-amplification-ddos-security-bulletin.pdf,2016.

③ 中国域名服务安全状况与态势分析报告 2015,http://www.cnnic.cn/gywm/xwzx/rdxw/2016/201602/W020160225366132699889.pdf,2016.

④ NetFPGA-10G Project. https://github.com/NetFPGA/NetFPGA-public/wiki.

3　M-DNS 设计与实现

3.1　M-DNS 的服务流程

拟态 DNS 服务器服务流程如图 2 所示.

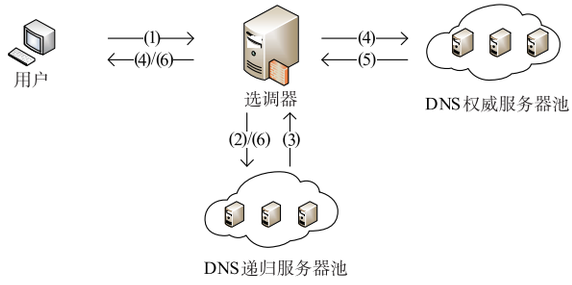


图2　原理流程图

具体服务步骤如下:

(1)用户发出 DNS 查询请求;

(2)选调器动态选取 *nos* 个服务器分发查询请求;

(3)服务器对查询请求进行处理:命中缓存,则返回应答;反之,则向 ANS 发查询请求.

(4)选调器根据各服务器的响应进行处理:

(*a*)若查询请求命中了 *noh* 个服务器的缓存:

(i)若所有应答一致,即 $IP_1 = IP_2 = ,\cdots, = IP_{noh}$,则将该结果返回给用户;

(ii)反之,若不一致,即 $IP_1 = IP_2 = ,\cdots, = IP_{noh}$不成立,则对各个结果进行统计,记录保存 nIP_1, $nIP_2,\cdots,nIP_{noh}$,其中 nIP_x 表示结果为 IP_x 的票数,如表 1 所示;并将未命中缓存的 RNS 的查询请求发给 ANS 处理;

(*b*)若全未命中,即 $noh = 0$,则将 RNS 的查询请求发给 ANS 进行处理;

(5)ANS 处理查询请求,并返回应答;

(6)选调器对 ANS 的应答进行统计(若是之前命中缓存但结果不一致的情况,就在已有了统计数据基础上统计各个结果的票数):

(*a*)如果某种结果 IP_r 通过了大数判决,即满足 $nIP_r \geqslant nos/2$,则:

(i)若是全票通过大数判决,即 $nIP_r = nos$:则将 IP_r 作为最终结果返回给用户,并将发给 RNS 进行缓存;

(ii)若是非全票通过大数判决,即 $nos/2 \leqslant nIP_r < nos$,则在预缓存队列中查询该域名对应的记录,如表 2:

①若没有该域名记录,则添加该域名以及对应的 IP_r,统计次数为 1;然后将该结果返回给用户;

②若已有该域名,则与记录进行比对:若一致,则将该结果返回用户,并发给 RNS 进行缓存,再清除该记录;若不一致,则丢弃报文,重新向 ANS 发送查询请求,并清除该记录;

(*b*)若所有查询结果都没通过大数判决,即 $nIP_r \geqslant nos/2$ 都不成立,则向 ANS 重新发送查询请求;

表 1　统计数据 1

IP 地址	统计票数
IP_1	nIP_1
IP_2	nIP_2
…	…
IP_s	nIP_s

表 2　统计数据 2

域名	IP 地址	统计票数
域名 1	IP_1	1
域名 2	IP_2	1
…	…	…
域名 *n*	IP_s	1

流程图如图 3 所示.

需要指出的是,上述预缓存次数默认取值为 2,即两次查询结果一致即信任该缓存内容,对于安全等级高的场景也可以增加预缓存次数.

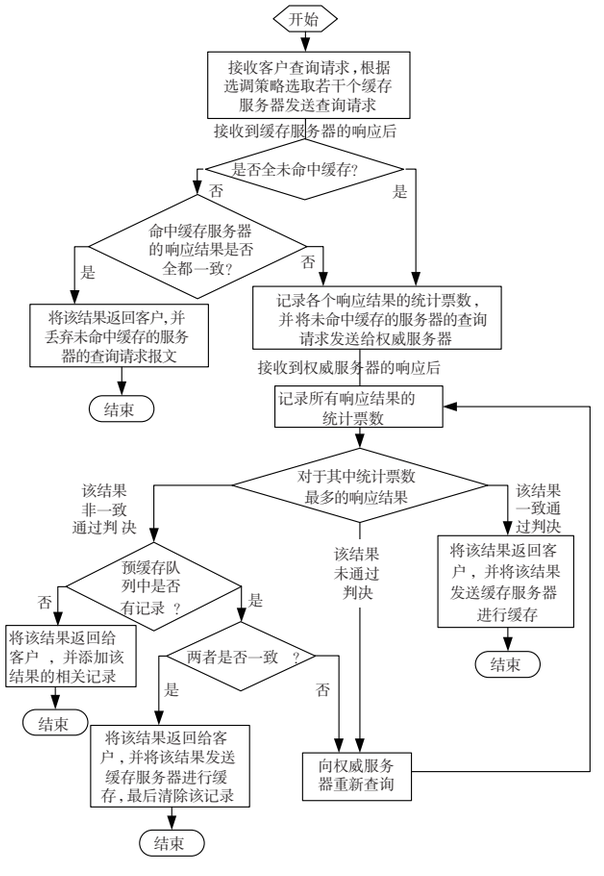


图3　工作流程图

3.2 动态调度

选调策略主要依据各服务器的状态信息而定,先给出 DNS 状态信息相关参数和定义.

3.2.1 参数及定义

定义 1 可信度(reliability):设服务器的集合为 A $\forall d \in A$,d 的可信度计算式如下:

$$d_{relia}=\begin{cases}1-e^{\frac{-z\cdot s}{1-s}}, & 0\leqslant s<1\\ 1, & s=1\end{cases} \tag{1}$$

s 的计算式如下,其中 d_{noa} 表示 d 受攻击的次数.

$$s=1-\frac{d_{noa}}{\sum_{p\in A}p_{noa}} \tag{2}$$

S. Akhshabi 等人[18]利用生物种群竞争的生存灭亡规律,分析了同层协议间竞争导致协议栈呈现沙漏状的缘由. 上式正是基于此思想给出的. 由于各服务器提供相同的服务功能,可以认为服务器池中服务器是一个生物种群,攻击者的攻击可视为捕食行为. 由式(2)可以看出,每个个体的生存力 s 都会受到攻击者捕食和种内竞争的影响:(1)服务器遭受攻击次数越多,其生存力就越小;(2)其他服务器遭受攻击次数越多,则该服务器生存力就越大. 综合式(1)(2)就可以得出可信度和被攻击次数的关系,如图 4 所示(假定总攻击次数为定值 10).

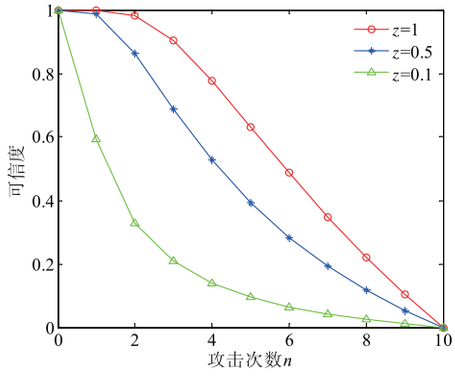


图4 可信度与攻击次数n的关系图

本文中参数 z 取值为(0,1),反映了选调器对可疑服务器的敏感程度,即当服务器出现异常后,其可信度的下降程度. z 值越小,服务器出现异常后可信度越低,下一个周期选取它的概率越小,系统安全性能也就越高(拟态防御边界以内时). 当然 z 值也不宜过小,否则可能导致其他服务器负载相应增加. 所以 z 的取值应当综合考虑系统的安全性能和工作效率.

定义 2 负载量(load):设服务器的集合为 A,,d 的负载量计算式如下:

$$d_{load}=\frac{d_{count}}{\sum_{p\in A}P_{count}} \tag{3}$$

其中 d_{count} 表示在一定周期内,选调器向 d 发送查询请求的数量. 本文用各服务器相对负载衡量其负载情况.

定义 3 选取系数($choose\_rate$):设服务器的集合为 A,,d 的选取权值计算式如下:

$$d_{choose\_rate}=\frac{a\times d_{relia}+b\times(1-d_{load})}{\sum_{p\in A}a\times p_{relia}+b\times(1-p_{load})} \tag{4}$$

其中 a,b 为常数. 由上式可见,选取系数是安全和性能方面的综合考量,可信度越高,选取系数越大;负载量越大,选取系数越小.

3.2.2 选调策略

首先设置一个基本阈值,若服务器的选取系数小于该阈值,则将该服务器加入黑名单中,在 TTL 时间内不再选用该服务器(该服务器的缓存将会自动清除),之后移出黑名单,并初始化. 设不在黑名单中的服务器的个数为 N,则具体选调的步骤为:

(1)若 $N<3$,则进入随机模式:从服务器池中随机选取一个,发送查询请求;

(2)反之,生成随机奇数 nos $(3\leqslant nos\leqslant N)$,依选取系数为概率选取 nos 个作为活跃集分发查询请求.

由于可信度和负载量都会动态更新,所以选取系数会变化,从而使选调具有动态性. 更新的方法如下:

(1)对于负载量,统计一定时间内发送给 RNS 的查询请求数,然后按照式(3)计算;

(2)对于可信度,若服务器的应答与大数判决结果不一致,则视为被攻击了一次,然后按式(1)(2)计算.

4 理论分析

本文从理论上对比分析了 M-DNS 和典型防御方案的效果,对比指标为一个有效时间内缓存投毒攻击成功的概率. 需要用到的变量如表 3 所示.

由于 DNS 的“First Answer Wins”原则,攻击者的伪造报文必须在 ANS 的应答报文前到达,所以有效的伪造报文数量为:

$$NoR=NoF\times RT \tag{5}$$

为实现更高的成功率,攻击者会发送多个同一域名的查询请求,这样攻击者只需“猜中”q 个中任意一个即可. 攻击者采用生日攻击①,根据文献[19]的生日攻击算式,给出单个报文“猜中”的概率 P 和攻击成功 $P_{success}$ 的关系式如下:

$$P_{success}=1-\prod_{i=1}^{NoR}\left(1-\frac{P\cdot q}{q-P\cdot(i-1)}\right) \tag{6}$$

下面求证各种情况下的概率.

① Birthday attack, https://en.wikipedia.org/wiki/Birthday_attack.

表 3　变量名称

变量名称	变量含义
P	攻击者发送一个伪造报文“猜中”的概率
$P_{success}$	攻击者攻击成功概率
NoP	可用的端口号数
$NoID$	标识符数
LoD	域名的字符长度(不含句点)
NS	ANS 的冗余配置个数
NoF	攻击者每秒能够发送的虚假攻击报文个数
nos	选取的 DNS 个数
RT	DNS 查询请求的回复时间
NoR	RT 时间内,攻击者可以发送的伪造报文数量
q	攻击者事先发送的相同域名的查询请求数
n	预缓存次数
K	攻击成功时发送的伪造报文数量(随机变量)

(1)基本模型:仅标识符随机. 该模型下攻击者需要猜中标识符和特定的 ANS,用 P_1 表示这种情况下攻击者发送单个报文“猜中”的概率,则:

$$P_1 = \frac{q}{NoID \times NS} \tag{7}$$

(2)源端口随机化:相比基本模型,该模型下攻击者需要另外猜中端口号,用 P_2 表示这种情况下攻击者发送单个报文“猜中”的概率,则:

$$P_2 = \frac{q}{NoID \times NS \times NoP} \tag{8}$$

(3)0x20-bit:相比上一模型,该模型下攻击者需要另外猜中大小写混杂的域名,用 P_3 表示这种情况下发送单个报文“猜中”的概率,则:

$$P_3 = \frac{q}{NoID \times NS \times NoP \times 2^{LoD}} \tag{9}$$

以上三种情况可将 P_1,P_2,P_3、代入式(6), NoR 由式(5)可知,计算得出各自 $P_{success}$.

(4)本文方案:发送单个报文“猜中”的概率同情况(2)相同,得:

$$P_4 = \frac{q}{NoID \times NS \times NoP} \tag{10}$$

下面对本文方案运用的各机制分别计算求证.

(a)预缓存机制:攻击者需要对同一域名成功攻击 n 次,所以成功概率 $P_{pre\text{-}cache}$ 为:

$$P_{pre-cache} = \left(1 - \prod_{i=1}^{NoR}\left(1 - \frac{P_4 \cdot q}{q - P_4 \cdot (i-1)}\right)\right)^n \tag{11}$$

(b)拟态防御:攻击者需要成功攻击至少 $l = (nos+1)/2$ 服务器,假设攻击者成功攻击时发送给各服务器的报文数量分别为 K_1,K_2,…,K_l,由于各服务器相互异构,所以 K_1,K_2,…,K_l 相互独立,所以成功概率为:

$$\begin{aligned} P_{MSD} &= P(K_1 + K_2 + ,\cdots, + K_l \leqslant NoR) \\ &= \sum_{k_1=1}^{NoR} P(K_1 = k_1) \cdot P(K_2 + K_3 + ,\cdots, + K_l \\ &\leqslant NoR - k_1) \\ &= \cdots \\ &= \sum_{k_1=1}^{NoR}\sum_{k_2=1}^{NoR-k_1}\cdots\sum_{k_l=1}^{NoR-k_1-k_2-,\cdots,-k_{l-1}} P(K_1 = k_1) \\ &\quad \cdot P(K_2 = k_2) \cdot ,\cdots, \cdot P(K_l = k_l) \end{aligned} \tag{12}$$

其中

$$P(K_m = k_m) = \left(\frac{P_4 \cdot q}{q - P_4 \cdot (k_m - 1)}\right) \cdot \prod_{i=1}^{k_m - 1}\left(1 - \frac{P_4 \cdot q}{q - P_4 \cdot (i-1)}\right) \tag{13}$$

上述计算是假设服务器对外可见给出的:①攻击者知道选出的服务器,因而只向特定服务器发送伪造报文;②攻击成功时攻击者能瞬时感知,因而立即转向下一个目标. 而实际在动态异构冗余模型下,服务器对外不可见,而这样的假设增加了攻击者攻击效率,所以是攻击成功概率的上限.

(c)拟态防御 + 预缓存模型:这种情况下,攻击者有以下两种攻击成功的方式:

(i)全票通过大数判决:需要攻击成功所有 nos 个服务器,所以成功概率为:

$$\begin{aligned} P'_{pre-cache\&MSD} &= P(K_1 + K_2 + ,\cdots, + K_{nos} \leqslant NoR) \\ &= \sum_{k_1=1}^{NoR}\sum_{k_2=1}^{NoR-k_1}\cdots\sum_{k_{nos}=1}^{NoR-k_1-k_2-,\cdots,-k_{nos-1}} P(K_1 = k_1) \\ &\quad \cdot P(K_2 = k_2) \cdot ,\cdots, \cdot P(K_{nos} = k_{nos}) \end{aligned} \tag{14}$$

(ii)非全票通过大数判决:需要攻击成功 n 次,所以成功概率为:

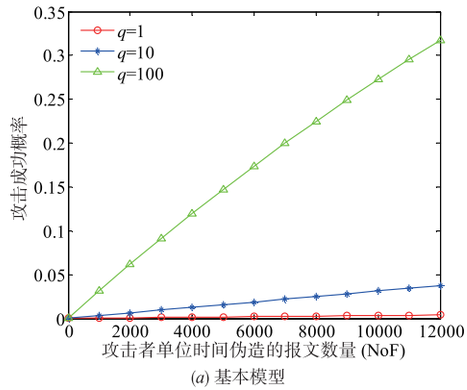
$$\begin{aligned} P''_{pre-cache\&MSD} &= (P(K_1 + K_2 + ,\cdots, + K_l \leqslant NoR))^n \\ &= (P_{MSD})^n \end{aligned} \tag{15}$$

5　仿真验证

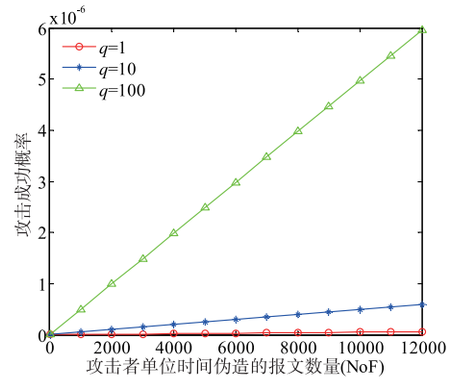
5.1　攻击成功率仿真分析

下面对理论分析中得到的概率,进行仿真验证,算式中参量的取值,如表 4 所示. 其中 RT 取为 0.1s,NS 为 4.8 是参考《报告》给出的数据;LoD 取为 12[18];n 取值为 2,nos 为 3,都是最简单的情况,所以是相应方案提供安全性能的下限. 攻击成功率和攻击者每秒发送报文数量的之间关系如图 5 所示.

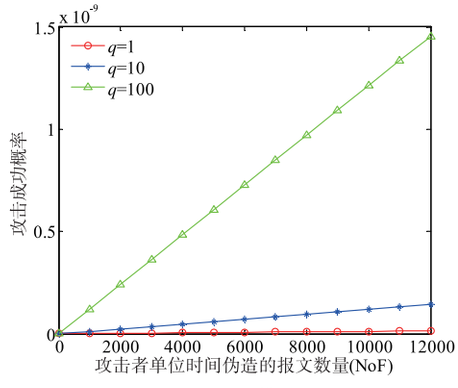
表 5 给出了 NoF 为 12000、q = 100 时各个模型的攻击成功概率的数量级.



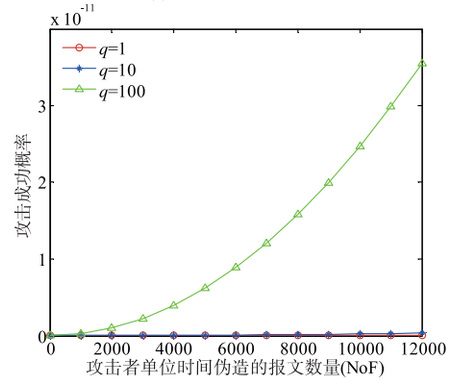
(*a*) 基本模型



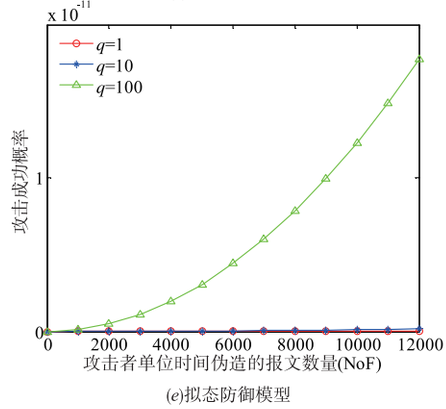
(*b*) 端口号随机化模型



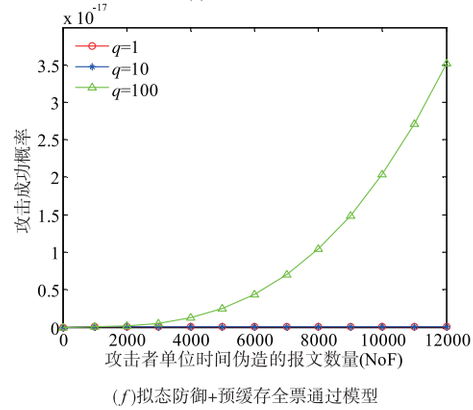
(*c*) 0x20-bit模型



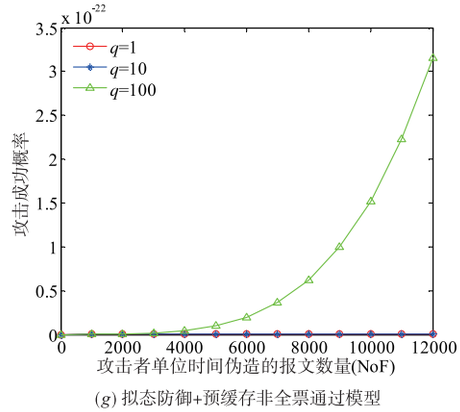
(*d*) 预缓存模型



(*e*)拟态防御模型



(*f*)拟态防御+预缓存全票通过模型



(*g*) 拟态防御+预缓存非全票通过模型

图5 攻击成功概率和攻击者单位时间伪造的报文数量关系图

表 4　变量仿真取值

变量名称	变量取值
NoP	64000
NoID	65535
nos	3
LoD	12
RT	0.1s
n	2
NS	4.8

表 5　仿真结果

模型	攻击成功概率数量级
基本模型	10^{-1}
端口号随机化	10^{-6}
0x20-bit 模型	10^{-9}
预缓存模型	10^{-11}
纯拟态模型	10^{-11}
预缓存及拟态全票通过模型	10^{-17}
预缓存及拟态非全票通过模型	10^{-22}

可以看出,对于基本模型,攻击者平均尝试 3 次攻击就可以攻击成功,端口号随机化一定程度上提升了安全性,0x20-bit 进一步提升了安全性,拟态防御模型和预缓存模型的数量级同为 10^{-11}. 而拟态防御结合预缓存机制的模型,攻击者通过攻击成功所有服务器的概率大于两次攻击成功半数以上的概率,所以其成功率最高仅为 10^{-17} 数量级,相比端口随机化策略、0x20-bit 提升攻击难度约 10^{10}、10^{7} 个数量级,可见 M-DNS 安全性能显著提升.

5.2　入侵容忍能力分析

下面 M-DNS 的入侵容忍效果. 本文从用户的角度考察服务器被攻击后的系统可靠性,以应答报文的正确率作为参考指标. 另外,为证明方案的入侵容忍能力,假设攻击者已具备“入侵能力”.

本文用 Python 进行仿真,参数设置为:域名查询空间为 100 个,服务器池中有 5 个服务器,TTL 为 20s,攻击者每 0.5 秒进行一次成功的缓存投毒攻击,每次投放 10 个带毒的缓存条目,查询请求每隔 0.5 秒到达,共发送 500 个请求. 最终仿真结果如图 6(*a*),横坐标表示攻击者每次可同时攻击的 RNS 个数,纵坐标表示各方案提供服务的正确率. 图 6(*b*)为 4.2.1 小节可信度指标中参数 z 分别为 0.1,0.5,1 时的对比图.

从图 6(*a*)中可以看出,对于 M-DNS,当攻击者每次能攻击的服务器数量 $NA=1$ 时,正确率为 100%,因为带毒服务器返回的应答不可能通过大数判决;当 $NA=2$ 时,正确率小幅下降,但仍高于其他模型;当 $NA=3$ 时,正确率开始大幅下降,这也验证了拟态防御边界,即被攻击的执行体不超过半数. $NA=4$、5 时,正确率已经与随机模式相差不大了.

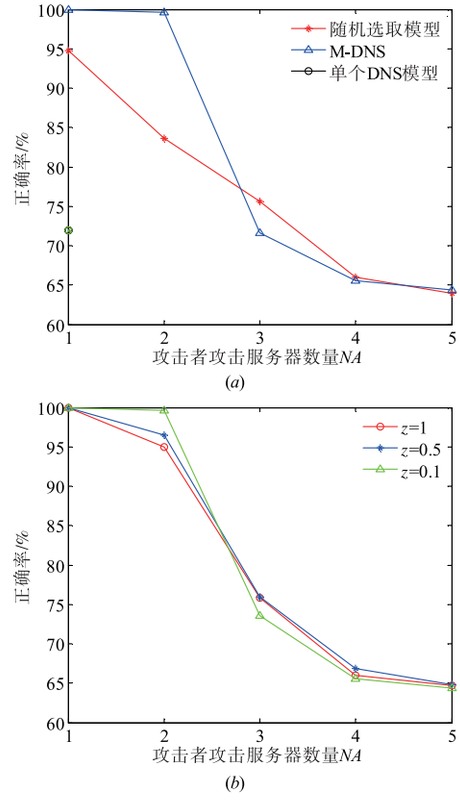


图6　正确率与攻击者攻击服务器数量*NA*的关系图

从图 6(*b*)中可以验证 3.2.1 小节关于参数 z 对系统安全性能影响的分析:当系统处于拟态防御边界以内($NA=1$、2)时,$z=0.1$ 时系统正确率高于 $z=0.5$ 和 $z=1$,所以 z 值越小,系统入侵容忍能力越高;但当超出拟态防御边界时,z 值越小,反而导致系统正确率越低,原因在于实际未受攻击的执行体因占少数而被选调器误判为“异常”,而受到攻击的因占多数而被判为“正常”,所以安全性能也就越低.

综上分析,当系统处于拟态防御边界以内时,本方案相比单个 DNS 模型提升了约 38% 的准确率,相比随机选取模型提升了约 5%($NA=1$)、19%($NA=2$)的准确率,可见 M-DNS 具备良好的入侵容忍能力.

5.3　代价分析

本文对单个 DNS 模型与 M-DNS 时延进行了对比分析. 涉及的相关符号如表 6 所示.

本文以 DNS 查询请求命中缓存的情况为例进行分析. 传统查询的时延 $T_{tradition}$ 和 M-DNS 时延 $T_{M\text{-}DNS}$ 可以分别表示为:

表 6 仿真结果

变量名称	变量含义
T_{rc}	RNS 与用户间时延
T_{ra}	RNS 与 ANS 间时延
T_{sc}	选调器与用户间时延
T_{sr}	选调器与 RNS 间时延
T_{sa}	选调器与 ANS 间时延
T_{proc_r}	服务器处理时延(传统的服务器查询时延)
T_{fd_r}	服务器转发时延
T_{proc_s}	选调器处理时延(选调时延和判决时延)
T_{fd_s}	选调器转发时延
$T_{addtion}$	额外增加时延

$$T_{tradition}=2T_{rc}+T_{proc\_r}+T_{fd\_r} \tag{16}$$

$$T_{M\text{-}DNS}=2T_{sc}+2T_{fd\_s}+T_{proc\_s}+2T_{sr}+T_{proc\_r}+T_{fd\_r} \tag{17}$$

所以,$T_{addition}=2T_{fd\_s}+T_{proc\_s}+2T_{sr}+2T_{sc}-2T_{rc}$,由于传播时延相比其他时延较小,可以认为 $T_{rc}\approx T_{sc}+T_{rs}$,所以 $T_{addtion}$ 可近似如下式:

$$T_{addition}\approx 2T_{fd\_s}+T_{proc\_s} \tag{18}$$

选调器处理时延 T_{proc_s} 包含选调和判决时延. 由于判决模块需要等待所有服务器的应答返回后做大数判决,所以是网络通信时延的最大值,而相比网络通信时延,选调器本身的处理和接收转发时延可忽略[20],所以 $T_{addtion}$ 进一步可近似为:

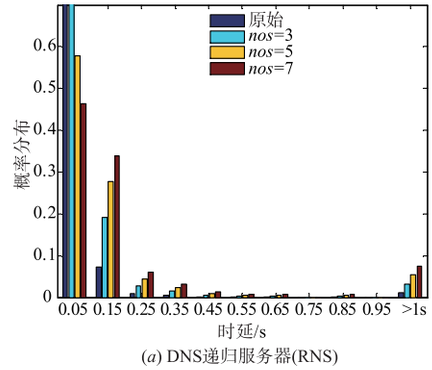
$$T_{addition}\approx T_{proc\_s} \tag{19}$$

为定量分析 T_{proc_s},根据《报告》中给出的 RNS 和二级及以下 ANS 查询时延的概率分布,计算 M-DNS 的查询时延 $\max(T_1,T_2,\cdots,T_{nos})$,当冗余个数 *nos* 分别为 3、5、7 时的概率分布,如图 7 所示.

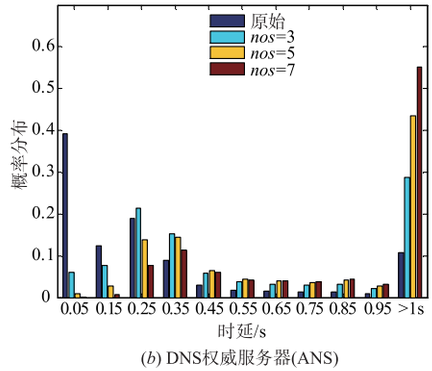
再求时延均值. 由于《报告》中时延最后一项给出的是 $>1s$ 的概率,无法计算具体均值,所以后面的计算中,假定 >1s 情况的均值为 1.05s(若实际情况大于该值,则时延还会相应增加). 得到如图 8 的时延均值. 进而得到额外增加的时延(与原时延的差值),如图 9. 以 5 个服务器为例,这种判决机制将额外增加 0.0858s 的时延.

对于未命中缓存的情况,由于时延还与各服务器的缓存情况相关,所以只分析向二级及以下权威服务器查询的时延,分析方法同前,如图 9 所示,以 5 个服务器为例,由判决机制带来的额外时延 $T_{addtion}$ 为 0.5057s(原时延均值仅为 0.2928s)

针对上述判决机制带来的额外时延问题,可以采用“先斩后奏”的策略——选调器接收少数应答后先进行一次简易判决,全票通过则先发给用户,否则等待后续完整的大数判决. 显然若采用这种策略,用户的等待时延可以相应降低. 总得来说,这种策略在不影响 RNS 的安全性的情况下,降低了用户的等待时延,而用户为此付出的代价就是安全性的降低,因此可以认为是效率与安全的“置换”.



(*a*) DNS递归服务器(RNS)



(*b*) DNS权威服务器(ANS)

图7 查询时延概率分布

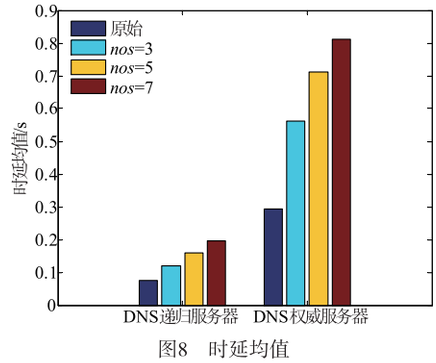


图8 时延均值

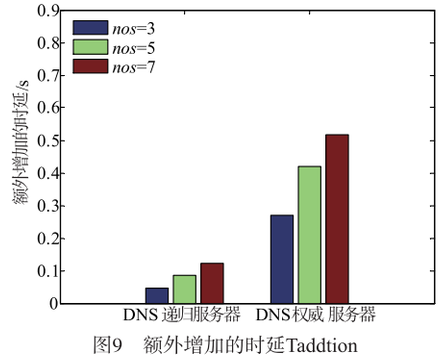


图9 额外增加的时延Taddtion

6 结束语

针对 DNS 缓存服务器的攻击是域名系统面临的主要威胁之一. 本文在分析研究已有防御机制的基础上,基于拟态防御思想,提出了 M-DNS 架构. M-DNS 通过构建异构冗余的选调空间,实现系统的动态调度,增加系统的不确定性,从而降低攻击成功概率.

当然,M-DNS 也存在一定的不足,主要表现在以下两点:(1)M-DNS 的选调器承担了较大的处理量,容易到达瓶颈和遭受 DDoS 攻击;(2)M-DNS 的判决机制会增加一定的时延. 下一步研究思路主要包括为选调策略和判决策略的优化,如代价分析部分提到的通过“先斩后奏”策略减少时延开销等.

参考文献

[1] Jin C, Hao Z Y, Wu Z G. Principles and defense strategies of DNS cache poisoning [J]. China Communications, 2009, 6(4): 17 – 22.

[2] Bassil R, Hobeica R. Security analysis and solution for thwarting cache poisoning attacks in the domain name system[A]. Proceedings of the 19th International Conference on Telecommunications [C]. Piscataway, USA: IEEE, 2012. 1 – 6.

[3] Herzberg A, Shulmanz H. Fragmentation considered poisonous, or: one-domain-to-rule-them-all. org[A]. Proceedings of IEEE Conference on Communications and Network Security[C]. Washington, USA: IEEE, 2013. 224 – 232.

[4] Gilad Y, Herzberg A. Fragmentation considered vulnerable [J]. ACM Trans on Information and System Security, 2013, 15(4): 1 – 31.

[5] Van Rijswijkdeij R, Sperotto A, Pras A. DNSSEC and its potential for DDoS attacks[A]. Proceedings of the Internet Measurement Conference [C]. New York, USA: ACM/USENIX, 2014. 449 – 460.

[6] Wu H, Dang X L, Zhang L, et al. Kalman Filter based DNS cache poisoning attack detection[A]. Proceedings of IEEE International Conference on Automation Science and Engineering[C]. Piscataway, USA: IEEE, 2015. 1594 – 1600.

[7] Wu H, Dang X L, Wang L. D, et al. Information fusion-based method for distributed domain name system cache poisoning attack detection and identification[J]. Institution of Engineering and Technology Information Security, 2016, 10(1): 37 – 44.

[8] 王秀利,王永吉. 基于命令紧密度的用户伪装入侵检测方法[J]. 电子学报,2014,42(6):1225 – 1229.
WANG Xiuli, WANG Yongji. Masquerader detection based on command closeness model[J]. Acta Electronica Sinica, 2014, 42(6): 1225 – 1229. (in Chinese)

[9] Perdisci R, Antonakakis M, Luo X P, et al. WSEC DNS: protecting recursive DNS resolvers from poisoning attacks [A]. International Conference on Dependable Systems and Networks[C]. Piscataway, USA: IEEE, 2009. 3 – 12.

[10] Dagon D, Antonakakis M, Vixie P, et al. Increased DNS forgery resistance through 0x20-bit encoding [A]. Proceedings of ACM Conference on Computer and Communications Security[C]. Alexandria, USA: ACM, 2008. 211 – 222.

[11] Yuan L H, Kant K, Mohapatra P, et al. DoX: a Peer-to-Peer antidote for DNS cache poisoning attacks[A]. Proceedings of International Conference on Communications [C]. Piscataway, USA: IEEE, 2006. 2345 – 2350.

[12] Yuan L, Chen C C, Mohapatra P, et al. A proxy view of quality of domain name service, poisoning attacks and survival strategies [J]. ACM Trans on Internet Technology, 2013, 12(3): 321 – 329.

[13] Fan L J, Wang Y. Z, Cheng X Q, et al. Prevent DNS cache poisoning using security proxy[A]. Proceedings of 12th International Conference on Parallel and Distributed Computing, Applications and Technologies [C]. Piscataway, USA: IEEE, 2011. 387 – 393.

[14] Mohan J, Puranik S, Chandrasekaran K. Reducing DNS cache poisoning attacks [A]. Proceedings of International Conference on Advanced Computing and Communication Systems[C]. Piscataway, USA: IEEE, 2015. 1 – 6.

[15] 邬江兴. 拟态计算与拟态安全防御的原意和愿景[J]. 电信科学,2014:30(7):2 – 7.
Wu Jiangxin. Meaning and vision of mimic computing and mimic security defense[J]. Telecommunications Science, 2014, 30(7): 2 – 7. (in Chinese)

[16] 邬江兴. 网络空间拟态防御研究[J]. 信息安全学报,2016,1(4):1 – 10.
Wu Jiangxin. Research on cyber mimic defense[J]. Journal of Cyber Security, 2016, 1(4): 1 – 10. (in Chinese)

[17] 扈红超,陈福才,王禛鹏. 拟态防御 DHR 模型若干问题探讨和性能评估[J]. 信息安全学报,2016,1(4):40 – 51.
Hu Hongchao, Chen Fucai, Wang Zhenpeng. Performance evaluations on DHR for cyberspace mimic defense[J]. Journal of Cyber Security, 2016, 1(4): 40 – 51. (in Chinese)

[18] Akhshabi S, Dovrolis C. The evolution of layered protocol stacks leads to an hourglass-shaped architecture[A]. Proceedings of ACM Special Interest Group on Data Communication[C]. New York, NY: ACM, 2011. 206 – 217.

[19] 孔政,姜秀柱. DNS 欺骗原理及其防御方案 [J]. 计算机工程,2010,36(3):125 – 127.
Kong Zhu, Jiang Xiuzhu. DNS spoofing principle and its

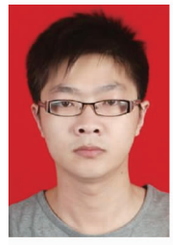
defense scheme[J]. Computer Engineering,2010,36(3):125 – 127. (in Chinese)

[20] Gummadi K,Saroiu S,Gribble S. King:Estimating latency between arbitrary Internet end hosts[A]. Proceedings of ACM Internet Measurement Workshop[C]. New York,USA:ACM,2002. 5 – 18.

作者简介

王禛鹏 男,1993 年出生于湖北黄冈. 现为国家数字交换系统工程技术研究中心硕士研究生. 主要研究方向为拟态安全防御.
E-mail:whuwzp@ whu. edu. cn

扈红超 男,1982 年出生于河南商丘. 现为国家数字交换系统工程技术研究中心副研究员. 主要研究方向为网络安全防御和新型网络体系结构.
E-mail:13633833568@139. com



程国振(通信作者) 男,1986 年出生于山东东营. 现为国家数字交换系统工程技术研究中心助理研究员. 主要研究方向为主动防御技术和 SDN 安全.
E-mail:guozhencheng@ hotmail. com