Статья Дырявый Word. Как спрятать боевую нагрузку в документе

Дырявый Word. Как спрятать боевую нагрузку в документе

В современных версиях Microsoft Office документы по умолчанию сохраняются в формате, основанном на Office Open XML, однако Microsoft не во всем следует открытому стандарту. Вариант Microsoft часто называют MOX, Microsoft Open XML. Он содержит некоторые вольности, создающие угрозы безопасности. Сегодня мы подробно разберем одну из них — так и не закрытую на момент написания статьи. Почти сразу после публикации чернового варианта OOXML началась битва за его стандартизацию. Вот краткая хронология версий.

- 2006 год — ECMA-376, первая версия;
- 2008 год — ISO/IEC 29500:2008 Transitional, переходная вторая версия;
- 2008 год — ECMA-376, part 2, и ISO/IEC 29500:2008 Strict, финальная вторая версия;
- 2011 год — ECMA-376, part 3, и ISO/IEC 29500:2011, третья версия;
- 2012 год — ECMA-376, part 4, и ISO/IEC 29500:2012, четвертая версия;
- 2015 год — ECMA-376, part 5, и ISO/IEC 29500-3:2015, пятая версия OOXML.

В 2016 году появились дополнения к пятой версии: ISO/IEC 29500-1:2016 и ISO/IEC 29500-4:2016. Работа над стандартом продолжается, а Microsoft допускает все больше проприетарных особенностей его реализации в новых версиях Office. Хуже того: компания не признает старые уязвимости, оставляя их в новых продуктах. Описываемая в статье дыра появилась в Microsoft Office 2013 и сохранилась вплоть до Office 2019.

ECMA-376 включает в себя три различные спецификации для каждого из трех основных типов документов Office — WordprocessingML для текстовых документов, SpreadsheetML для электронных таблиц и PresentationML для презентаций. Мы будем использовать WordprocessingML.

Я возьму на себя смелость указать на два критичных с точки зрения безопасности недостатка MOX, унаследованных от OOXML:

- возможность легкого редактирования внутренней структуры документов;
- отсутствие проверок на злонамеренную модификацию.

По сути, MOX и OOXML — это XML в ZIP. Это отличный hacker-friendly-формат, поскольку найти и заменить свойства объектов в нем исключительно просто даже без использования HEX-редакторов и прочих специфических утилит. Достаточно встроенной в Windows поддержки ZIP и «Блокнота». Весь код легко читается глазами и правится, как текст. Ни сверки контрольных сумм, ни каких-то иных специфических проверок при этом не выполняется. Word лишь проверяет целостность документа, которая не нарушается при подменах с соблюдением правил синтаксиса.

Если в документ вставлен объект, загружаемый с внешнего ресурса (например, ссылка на видео), то в соответствующей секции создается легко читаемая (и так же просто изменяемая) гиперссылка. Это прямая дорога к фишингу или запуску троянов в один клик.

Описание уязвимости

В базе уязвимостей MITRE есть много однотипных записей вида: «Microsoft Office… do not properly validate record information during parsing of (Excel spreadsheets / Word documents, Power Point presentations)… which allows remote attackers to execute arbitrary code or cause a denial of service (memory corruption) via a crafted file)». Проще говоря, проблемы парсинга XML в MS Office неисчерпаемы, как атом.

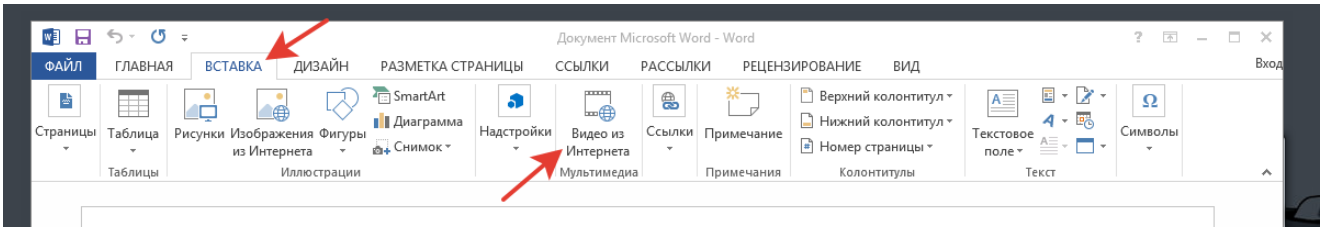
Я уже рассматривал одну из них в статье «Pass the Hash через Open XML». Сейчас мы реализуем другую атаку, также слегка поковыряв недра файла Word.

Начиная с Office 2013 в OOXML стал доступен класс WebVideoProperty. Он появляется в разметке при вставке в документ онлайн-видео и описывает параметры его воспроизведения через набор атрибутов.

Нас будет интересовать атрибут embeddedHtml. Он нужен для вставки внешних объектов и содержит ссылку на них (например, на видеоролик YouTube). Из-за того что этот параметр «знает», из какого закоулка интернета тянуть картинку видеозаписи, его нельзя опустить при парсинге.

Ищем объект подмены

Давай выполним простую атаку подмены и пощупаем уязвимость своими руками. Запускаем Word (требуется версия 2013 или выше, так как нам нужна полная поддержка ISO/IEC 29500 Strict) и идем в меню «Вставка».



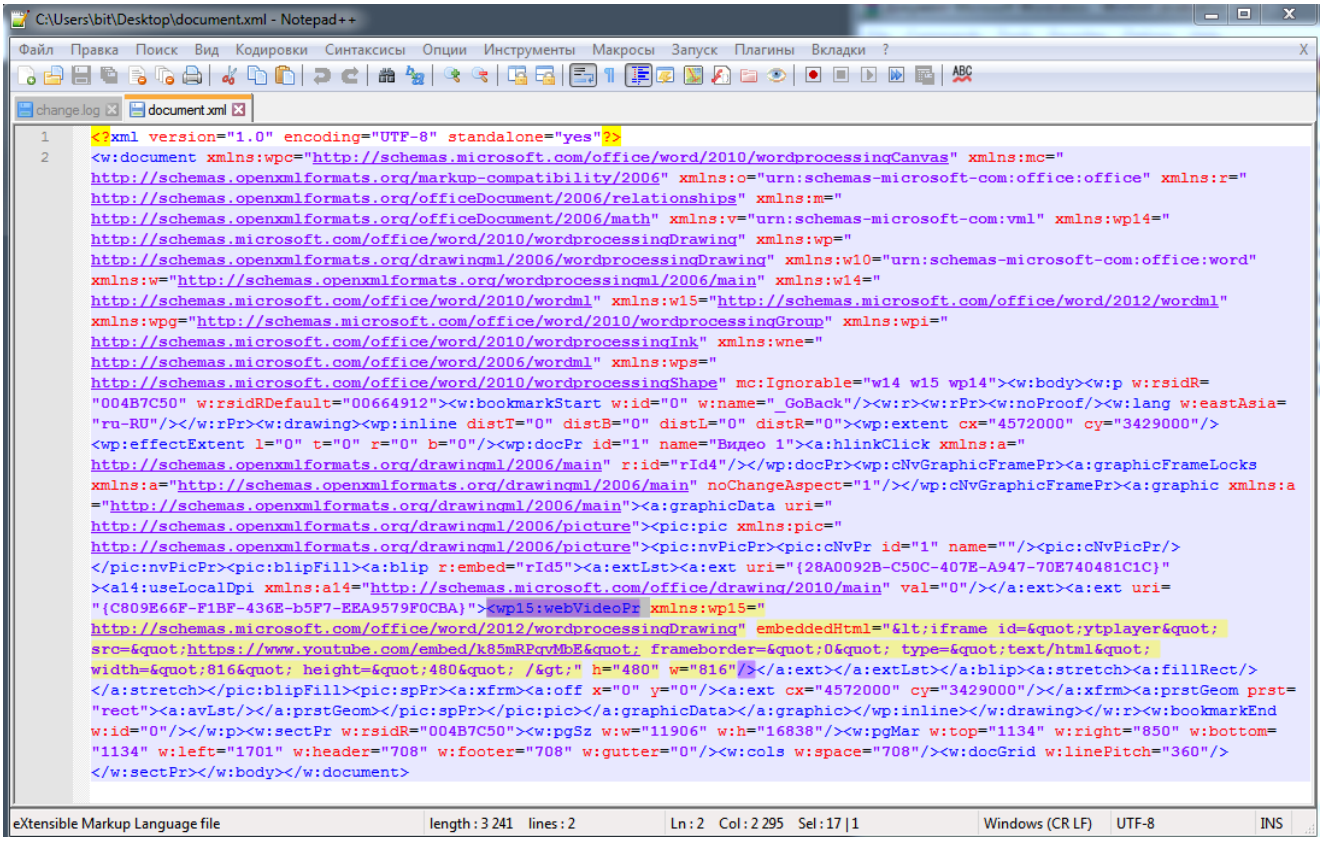
В появившемся окне в строчке напротив YouTube я просто вписал слово «видео» и выбрал первую понравившуюся картинку. Вставка ролика отобразилась на листе документа типичной превьюшкой.

Сохраним и закроем его. Обрати внимание, что размер файла почти не изменился. У меня он занимал считаные килобайты. Значит, вставленное видео не сохраняется локально, а всегда запрашивается из интернета по известной ссылке.

Следующим шагом нам надо заглянуть в нутро документа. Меняем расширение .docx на .zip, открываем любым архиватором и видим содержимое.

В папке \word нам нужен файл document.xml. Разархивируем и откроем его на редактирование (подойдет и простой Notepad, хотя Notepad++ удобнее из-за подсветки синтаксиса).

В скудной документации о классе WebVideoProperty указано, что в теле документа он именуется wp15:webVideoPr. Находим эту секцию и смотрим ее содержание.



wp15:webVideoPr в структуре файла document.xml

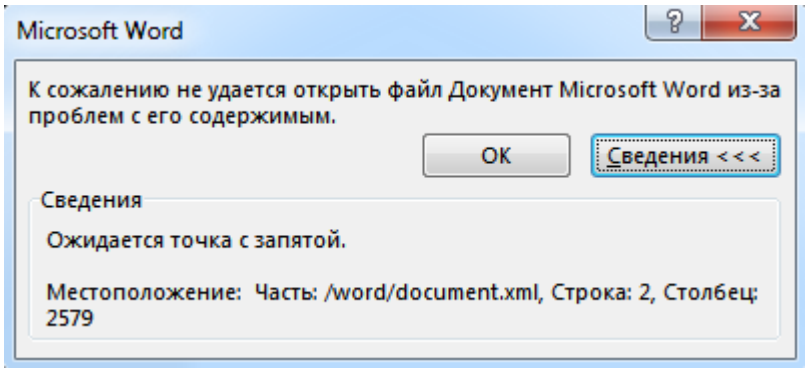
Конструкция изначально выглядит следующим образом:
Code:

```
embeddedHtml="&lt;iframe id=&quot;ytplayer&quot;
src=&quot;https://www.youtube.com/embed/k85mRPqvMbE&quot; frameborder=&quot;0&quot;
type=&quot;text/html&quot; width=&quot;816&quot; height=&quot;480&quot; /&gt;"
```

Атрибут embeddedHtml содержит iframe YouTube, который при замене на HTML или JavaScript будет выполняться. А это не что иное, как уязвимость!

Эксплуатация уязвимости
Если внимательно посмотреть на содержимое секции, то можно заметить, что символы < и >заменены на < и >. Это способ записи символьных данных без использования раздела CDATA. Они указывают парсеру на то, что эта часть документа не содержит разметки. Так же мы должны поступить со всеми нашими спецсимволами.

Еще стоит отметить, что все параметры отделены друг от друга точкой с запятой. Если мы пропустим хотя бы одну из них, то при открытии документа произойдет ошибка проверки целостности файла. Правда, Word облегчает задачу, подсказывая, где именно мы ошиблись (см. скриншот).



Ошибка Word при неправильном синтаксисе document.xml

Давай удалим все, что находится между кавычками, и попробуем вставить свой HTML-код. Сначала добавим отображаемую часть ссылки:
Code:

```
<H1>xakep</H1>
```

В нашем случае она будет выглядеть следующим образом:
Code:

```
embeddedHtml="&lt;H1&gt;Xakep&lt;/H1&gt;"
```

Теперь подменим исходную ссылку на ролик с YouTube своей. Например, загрузим какой-нибудь файл на компьютер жертвы.

В качестве «вредоносного сервера» я поднял дистрибутив Ubuntu 16.4 с Apache2 и положил в каталог /var/www/html два файла: условного зловреда и простенькую HTML-страницу со ссылкой на него. IP-адрес сервера в локальной сети — 192.168.1.20.

Далее нам нужно все это указать в embeddedHtml:
Code:

```
embeddedHtml="&lt;a href=&quot;http://192.168.1.20/1.html&quot; frameborder=&quot;0&quot;
type=&quot;text/html&quot; width=&quot;816&quot; height=&quot;480&quot; &gt;Хакер&lt;/a&gt;"
h="480" w="816"
```

Теперь сохраним наш измененный файл и запустим его.

Для первой проверки я подготовил имитацию жертвы — компьютер с Windows 10 (1803) и MS Office 2016 Professional Plus VL x86, который мы и будем атаковать.

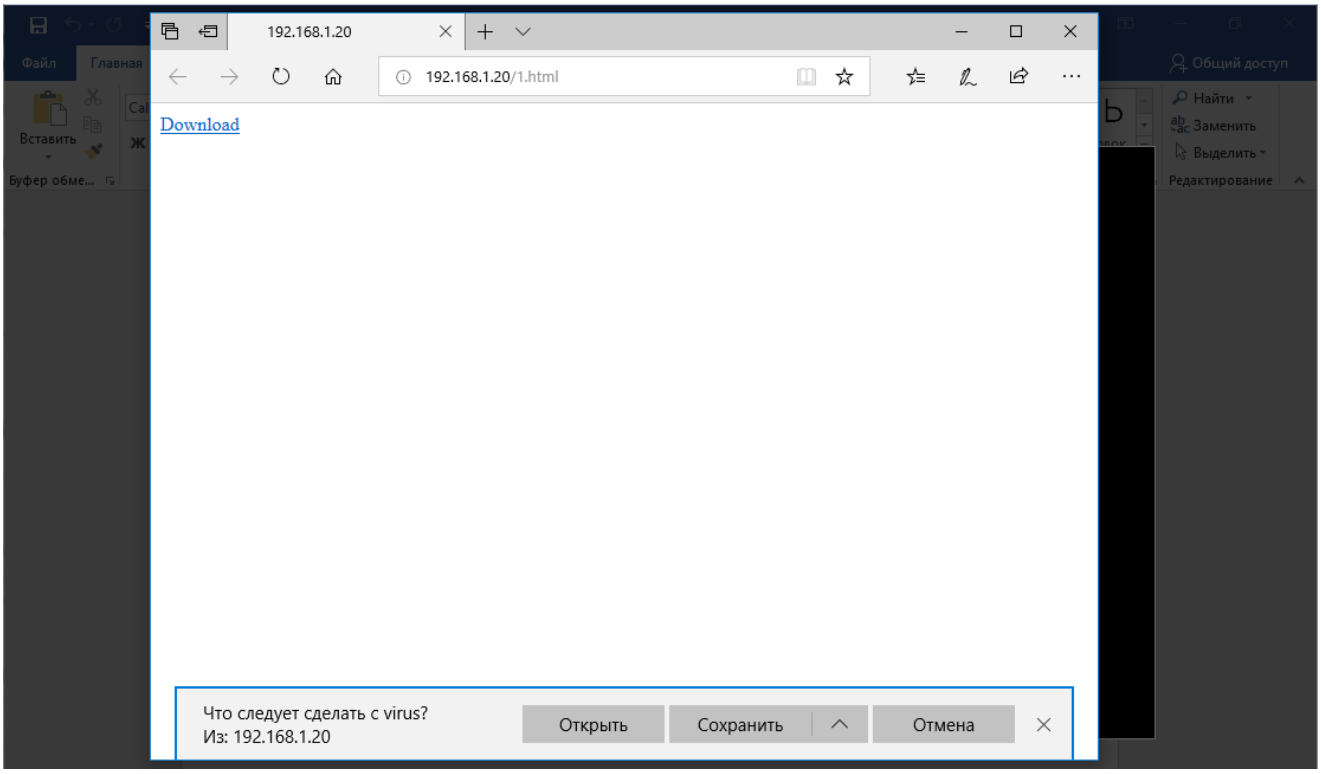
После запуска файла не видно ничего необычного. Открывается документ со вставленным видеороликом. При наведении курсора отображается корректная ссылка на него. Однако, если нажать на воспроизведение, вместо видео отобразится наша ссылка.

Действие после активации воспроизведения

В реальном сценарии вместо хакер лучше написать что-то более подходящее для фишинга. Например, Click to begin playback.

Может быть, со времен Office 2016 уязвимость уже закрыли? Давай проверим, сработает ли наш трюк в Microsoft Office Pro Plus 2019 и Windows 10 (1803).

Открываем тот же файл и пробуем запустить видео. Слово «хакер» так же подчеркнуто и выступает в качестве ссылки. При клике на него открывается Edge с нашей страницей на «злом сервере». На ней все та же ссылка для загрузки зловреда.



Открытие фишинговой ссылки при клике в окне предпросмотра видеоролика

Для большей наглядности я записал короткий ролик с демонстрацией атаки. Кстати, его тоже можно использовать как приманку и вставить в фишинговый файл Word (рекурсия!).

Примечания
Стоит уточнить ряд важных моментов. Трюк работает, если пользователь просто нажимает левой клавишей мыши на кнопку Play в превьюшке вставленного в

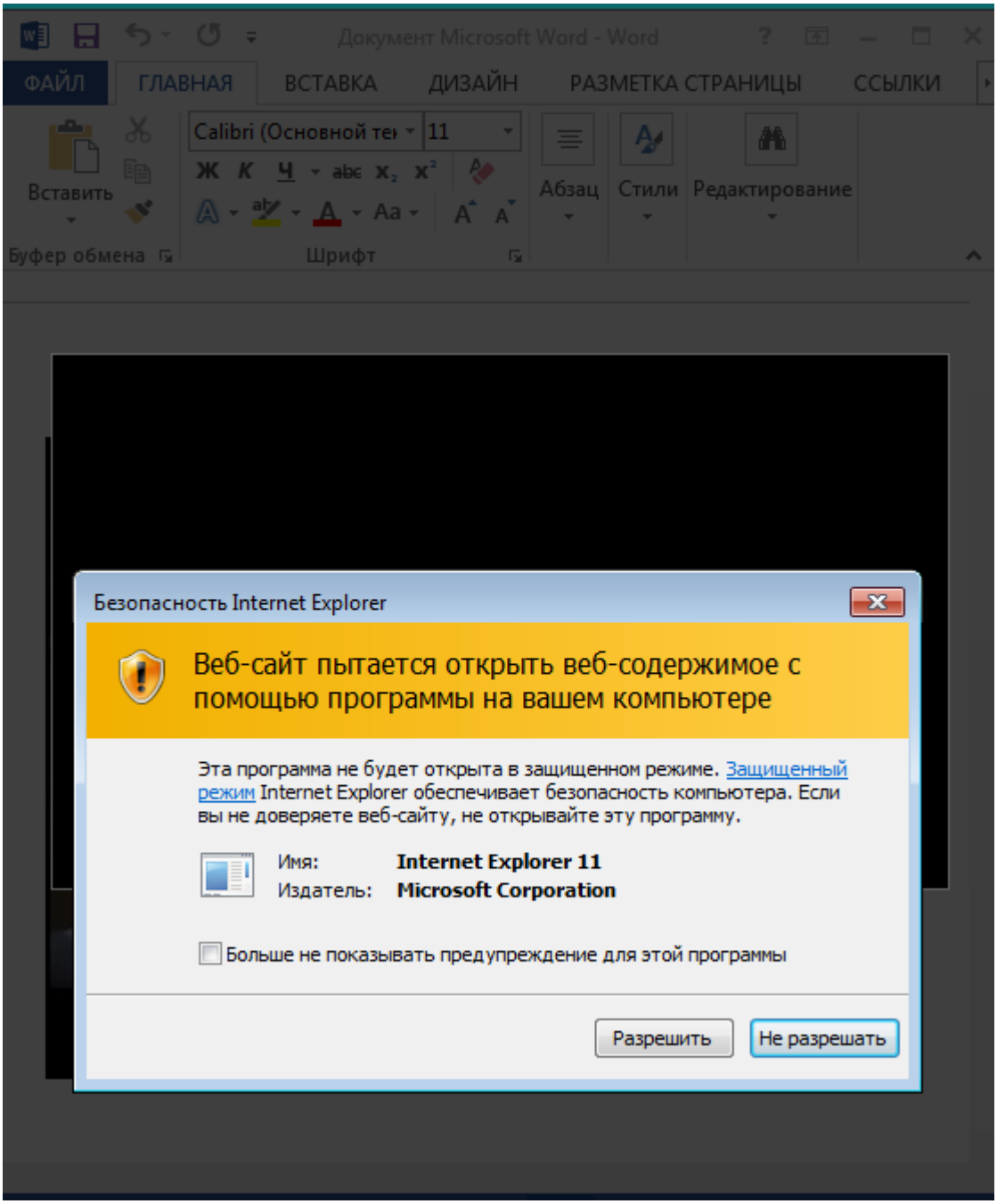
документ ролика. Так делает большинство, но продвинутые могут кликнуть с зажатой клавишей Ctrl. В этом случае начнется воспроизведение видео. Почему так происходит?

При нажатии левой кнопки мыши с клавишей Ctrl и без нее Word обращается к разным секциям документа. В первом случае он считывает подмененную ссылку из word\document.xml и предлагает выполнить переход по ней. Во втором — считывает оригинальную ссылку на видеоролик из word_rels\document.xml.rels и запускает его.

Если же подменить URL в обеих секциях, фишинговая ссылка будет отображаться при наведении курсора мыши на превьюшку. Такой грубый вариант атаки сработает только с самыми невнимательными пользователями.

Также заметь, что при нажатии левой кнопкой мыши с Ctrl откроется браузер, установленный по умолчанию. Если же просто кликать, то (вредоносная) ссылка всегда будет открываться в браузере. Это еще один вектор атаки.

В зависимости от версий ОС и Office, а также настроек безопасности у жертвы могут сработать дополнительные компоненты защиты. Например, Office 2010 ограниченно поддерживает OOXML. Он предложит разрешить редактирование документа прежде, чем позволит кликнуть на превьюшку. В Windows 7 IE выдаст предупреждение при открытии ссылки.



Предупреждение в Windows 7

С Windows 10 наблюдается совсем другая картина. В дефолтных настройках (а у потенциальной жертвы они, как правило, такие) ни IE, ни Edge ни о чем не предупреждают. Ссылка открывается без дополнительных действий. Отсюда можно сделать парадоксальный вывод о том, что новая ОС оказалась более уязвима к таким атакам.

Для дополнительной проверки я открывал файл с измененной ссылкой в следующих офисных пакетах:

- Open Office 4.1.6;
- Libre Office 6.1.3;
- Soft Maker Office 2018.

Все они официально поддерживают OOXML, причем в точном соответствии со стандартом ISO/IEC 29500. Ни один из них не подвержен рассмотренной уязвимости, поскольку все вольности Microsoft игнорируются. Фактически пользователь оказывается лучше защищен благодаря отсутствию поддержки проприетарной функции вставки онлайн-видео… но ведь, кроме роликов, в документы можно вставлять и другие объекты.

Выводы
В этом примере мы рассмотрели, как модифицировать XML-разметку офисных документов, чтобы выполнить подмену ссылки и подсунуть жертве зловреда. Вместо пугающей надписи «Хакер» можно подобрать что-нибудь более привлекательное. Например, указать, что для воспроизведения ролика необходимо скачать плагин или обновить плеер. Красочно оформляем страницу загрузки, внушаем, что все безопасно, и дело в шляпе. Белой или черной — решать тебе.

WWW

автор: 8bit (Роман Вегелин)
хакер.ру