Ветров Н.А., Ветров А.Н.
Россия, г. Санкт-Петербург
Международный банковский институт
Санкт- Петербургский государственный электротехнический университет "ЛЭТИ"

ОСОБЕННОСТИ ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ НА УРОВНЕ ПРИЛОЖЕНИЙ В СРЕДЕ WWW С ИСПОЛЬЗОВАНИЕМ PHP

При создании Web-приложений часто пренебрегают обеспечением безопасности. Информационная безопасность охватывает очень широкий круг проблем и вынуждает следить за самыми последними достижениями в этой области. Систему безопасности нужно постоянно поддерживать, ее нельзя просто добавить к проекту после того, как он завершен и внедрен. Обеспечение защиты приложения в среде WWW является многоаспектной проблемой, которая начинается с носителя,- WEB-сервера,- это грамотное обеспечение механизмов исправлений и обновлений (как на уровне окружения приложения, так и на уровне операционной системы), укрепление сервера (вместо службы telnet крайне желательно использование ее защищенных альтернатив, например OpenSSH,- потенциально снижается интенсивность использования FTP и telnet и т.п.), постоянный мониторинг состояния сервера (анализ технических журнальных файлов сервера и среды), информированность (постоянная осведомленность как основа защиты).

Вопросы защиты также включают: разграничение и анализ прав доступа к каталогам ServerRoot, запрет на изменение параметров настроек сервера, защита файлов сервера:

```
# не разрешать доступ к файлам вне DocumentRoot
<Directory />
AllowOwerride None
Options None
Order deny, allow
Deny from all
</Directory>
# разрешить доступ к файлам в DocumentRoot
<Directory /usr/local/apache/htdocs>
AllowOwerride None
Options Indexes FollowSymlinks
Order allow, deny
Allow from all
</Directory>
# предоставление каждой группе приложений своего окружения
UserDir disabled
UserDir enabled alice bob
UserDir public_html
<Directory “/home/public_html”>
AllowOverride None
Options Includes NOEXEC SymbLinksIfOwnerMatch
Order allow, deny
Allow from all
</Directory>
UserDir disabled root
```

Управление включениями на стороне сервера SSI (собственные ограничения для каждого приложения, например,- Apache имеет параметр IncludeNOEXEC, который разрешает SSL, но запрещает пользователям запускать из них программы или сценарии CGI).

Разрешение на выполнение сценариев CGI только из определенного местоположения (можно запретить выполнение CGI, разрешив при этом выполнять сценарии PHP).

(подан в «МБИ» на межд. конф. «АПЭ и НТП (СЧ)» «МАН ВШ», март 2004 г.)

Размещение анализатора PHP вне иерархии каталогов WEB-сервера (исключается возможности паразитного злоупотребления анализатором).

Идентификация и аутентификация пользователя с помощью PHP (реализуется по принципу запрос-отзыв). Хотя реализация идентификации посредством PHP несколько сложнее, но позитивные результаты стоят приложенных усилий.

К ключевым преимуществам данного способа аутентификации следует отнести: она не может бьггь отменена (пользователь может "разрегистрироваться", что достижимо в слиянии с Apache,- так называемый контролируемый откат транзакции на любой стадии регистрации); у нее также может быть введен срок действия (обеспечивается ресурсосбережение и автоматизация мониторинга учетных записей,- регистрация автоматически становится недействительной по истечении определенного временного интервала); ее можно настраивать (ограничивают лишь уровень мастерства и воображение, разработчик полностью управляет процессом аутентификации); в основе аутентификации обеспечивается подключение баз данных (можно вести полный учет действий пользователей и использовать любые данные); она является транзакционно-итерационной с различными уровнями (технологические решения для каждого отдельного объекта, высокая степень гибкости и надежности); помимо аутентификации предоставляется возможность регистрации (высокая степень автоматизации управления учетными записями и анализ поведения пользователя, которому предоставляется возможность зарегистрироваться); полностью поддерживается CGI (универсальные возможности интерфейсного обмена между приложениями); возможность реализации проверки и учета IP-адресов (хотя это иногда не дает должного эффекта); использование криптографии (шифрование данных и создание контрольных сумм или дайджестов для восстановления информации); использование шифрования (использование и реализация алгоритмов, обеспечивающих работу с открытыми и закрытыми ключами, а также создание и обмен сертификатами); использование хэш-функций (часто используются для хранения паролей и идентификации некоторых фрагментов данных); применение механизма suEXEC в Apache (установка определенных ограничений на время периода выполнения приложения); обеспечение безопасности сценариев PHP (минимизация риска при запуске); надежность приложений, хранение и пересылка конфиденциальной информации (защита программного обеспечения от непроизвольного использования с внешней стороны); анализ данных пользователя (исключение возможности передачи сомнительной информации и нежелательных управляющих конструкций в составе входных данных).

При создании приложений для WWW часто пренебрегают обеспечением безопасности,- это объясняется тем, что защищенность трудно измерить количественно, а для рядовых посетителей ресурса (сайта) она остается незаметной. Для грамотных пользователей пробелы в системе безопасности легко обнаруживаются, а при этом последствия могут быть самыми непредсказуемыми, как правило, катастрофическими.

Доверять ли конфиденциальную информацию тем, кто не в состоянии обеспечить ее конфиденциальность?

(подан в «МБИ» на межд. конф. «АПЭ и НТП (СЧ)» «МАН ВШ», март 2004 г.)