VI Висновки

Використання розглянутого методу АВС аналізу в питаннях захисту інформації дозволяє швидко та зручно визначити механізми захисту, які необхідно застосовувати для підтримання належного рівня політики інформаційної безпеки. Тим самим створюються оптимізовані, ефективні та економічно обґрунтовані системи захисту інформації, оскільки основні витрати спрямовані на найбільш вразливі місця об'єкту, а інші – зменшуються за умови відсутності порушень політики безпеки. Зокрема, це вибір комплексу засобів захисту, який забезпечує захист від актуальних на даний момент вразливостей.

Використання методу АВС аналізу надає можливість надалі:

- обирати оптимальні системи захисту інформації, які забезпечують захист від актуальних загроз та вразливостей;
- створювати економічно ефективні системи захисту, які будуть гарантувати належний рівень захищеності, а витрати на неї не будуть приводити до збитків, пов'язаних з перевищенням вартості системи величини можливого отриманого прибутку від інформації, що захищається;
- здійснювати зручно та швидко адаптацію систем захисту до умов, що змінюються.

Таким чином, для вибору оптимальних, ефективних та економічно обґрунтованих систем захисту інформації доцільно застосовувати метод АВС аналізу. Напрямом подальшого дослідження може бути розробка методик використання методу АВС аналізу на етапах побудови систем захисту інформації.

Література: ***1.*** *Де Гроот М. Оптимальные статистические решения [Текст]: пер. с англ. / М. Де Гроот; ред. Ю. В. Линник. – М.: Мир, 1974. – 491 с.* ***2.*** *Баутов А. Экономический взгляд на проблемы информационной безопасности. [Електронний ресурс] / Баутов А.// Открытые системы – 2002. – № 2. – Режим доступу: http://www.osp.ru/os/2002/02/181118/. – Назва з екрану.* ***3.*** *Adi Shamir. Turing Award lecture [Електронний ресурс] / Adi Shamir. – 2004 – Режим доступу: http://www.financialcryptography.com/mt/archives/000147.html. – Назва з екрану.* ***4.*** *АВС-анализ [Електронний ресурс] – Режим доступу: http://www.abc-analysis.ru/. – Назва з екрану.* ***5.*** *Фишер Андрей. Методы выделения групп в АВС анализе [Електронний ресурс] / Фишер Андрей. – Режим доступу: http://www.transmap.ru/articles/view/169. – Назва з екрану.* ***6.*** *ДСТУ 3396.0-96. Захист інформації. Технічний захист інформації. Основні положення.* ***7.*** *ДСТУ 3396.1-96. Захист інформації. Технічний захист інформації. Порядок проведення робіт.* ***8.*** *Методы АВС анализа номенклатурных групп [Електронний ресурс]: (По книге «Модели м методы теории Логистики», Лукинский В. С. и др.). – Режим доступу:http://www.adviss.ru/content/view/45/7/. – Назва з екрану.* ***9.*** *Аналіз АВС [Електронний ресурс]. – Режим доступу: http://www.basegroup.ru/glossary/definitions/abc_analysis/. – Назва з екрану.* ***10.*** *Отчет по уязвимостям за второй квартал 2008 года [Електронний ресурс] / Валерий Марчук. – Режим доступу: http://www.securitylab.ru. – Назва з екрану.* ***11.*** *ISO/IEC 27001:2005 Information technology – Security techniques – Specification for an Information. Security Management System.* ***12.*** *ISO/IEC 27005:2008 Information technology – Security techniques – Information security risk management.* ***13.*** *Черней Г. А. Оценка угроз безопасности автоматизированным информационным системам [Електронний ресурс]: – Режим доступа: http://security.ase.md/publ/ru/pubru01.html.* ***14.*** *НД ТЗІ 2.5-005-99 Класифікація автоматизованих систем і стандартні функціональні профілі захищеності оброблюваної інформації від несанкціонованого доступу. – 16 с.* ***15.*** *НД ТЗІ 2.5-004-99 Критерії оцінки захищеності інформації в комп'ютерних системах від несанкціонованого доступу. – 57 с.*

УДК 004.056.53

АВТЕНТИФИКАЦИЯ В ЗАЩИЩЕННЫХ ПРИЛОЖЕНИЯХ

Михаил Коломыцев, Светлана Носок

Национальный технический университет Украины «Киевский политехнический институт»

Анотація: **Наведено місце додатків в інфраструктурі комп'ютерної системи, розглянуто основні аспекти і задачі захисту додатків, проаналізовано різні способи автентифікації як засобу безпеки додатків та висвітлено загальні вимоги до реалізації механізму автентифікації.**

Summary: **The article outlines the place of applications in the infrastructure of computer system. The main aspects and tasks of application protection are described. Different ways of authentication as a method of application protection are analysed, and general reguirements concerning realization of authentication mechanism are mentioned.**

Ключові слова: **Автентифікація, додатки, операційна система, система управління базами даних.**

I Место приложений в инфраструктуре компьютерной системы

Приложение – это компьютерная программа (или набор программ), которая выполняется под управлением операционной системы. Приложение использует сервисы операционной системы и других обслуживающих программ с целью выполнения необходимых пользователю (или другим программам) задач.

Существуют различные подходы к классификации приложения. С точки зрения архитектуры приложения будем различать следующие виды приложений.

– Серверное приложение. Его задачей является организация взаимодействия с клиентским приложением, получение или передача данных по запросам клиентского приложения.

– Клиентское приложение. Его задачей является организация запросов к серверному приложению для предоставления информации пользователям. Оно предоставляет пользователям возможность создавать или модифицировать информацию.

– Автономное (standalone) приложение. Использует файловую систему для хранения и получения данных с целью предоставления их пользователям.

Типичная архитектура программной системы определяет взаимодействие приложения через программный интерфейс с операционной средой и инфраструктурой компьютерной системы. Инфраструктура компьютерной системы включает в себя промежуточное ПО и другие служебные протоколы, не программируемые совместно с приложением, а также платформу (системное программное и аппаратное обеспечение, драйверы и т. п.).

II Задача защиты приложений

Безопасность приложения, в конечном счете, означает гарантию того, что оно функционирует в режиме, обеспечивающем выполнение всех требований политики безопасности относительно самого приложения, данных, которыми оно манипулирует, системного окружения и пользователей. В безопасности приложения можно выделить следующие основные задачи:

- гарантировать, что данные, которые создаются, модифицируются, сохраняются и (или) передаются с помощью приложения, защищены от неавторизованного раскрытия, подмены, искажения или разрушения со стороны пользователей, внешних процессов и самого приложения;

- создать, если необходимо, дополнительные сервисы безопасности, которые в общесистемной инфраструктуре безопасности (сеть, операционная система, СУБД) либо не реализованы, либо неадекватны возможным угрозам; например, если механизм управления доступом сервера приложений не защищает файлы с конфиденциальной информацией от неавторизованного доступа, в приложение может быть включен программный интерфейс к криптографическим функциям инфраструктуры безопасности, с тем, чтобы данные хранились только в зашифрованном виде.

Для решения этих задач в приложении должен быть реализован набор сервисов безопасности. Их реализация возможна одним из трех способов:

- реализация сервиса безопасности внутри приложения; например, созданием собственной системы регистрации событий, связанных с безопасностью, отличной от систем регистрации ОС и СУБД;

- непосредственное обращение к сервисам безопасности промежуточного ПО или других инфраструктурных компонент; например, приложение может содержать программный интерфейс к подсистеме PKI (инфраструктура открытых ключей) для проверки сертификатов в процессе аутентификации пользователей;

- путем проверки, что защита обеспечивается функциями безопасности промежуточного ПО или инфраструктурных компонент; например, приложение может проверять флаги или счетчики, сигнализирующие о том, что файл прошел антивирусную проверку.

III Аутентификация как сервис безопасности приложения

Аутентификация – это механизм безопасности, предназначенный для проверки того, что пользователь, или процесс авторизован для доступа к приложению. Аутентификация защищает от неавторизованного доступа к приложению и его использования.

С учетом возможной архитектуры приложения можно выделить следующие виды аутентификации:

- аутентификация в изолированном (standalone) приложении; изолированное приложение – это приложение, не использующее сетевые или другие прикладные сервисы; как правило, такое приложение использует механизм аутентификации операционной системы; примерами таких программ являются программы Блокнот или Калькулятор в ОС Windows;

- аутентификация на сервере в клиент-серверном приложении; сервер – это приложение, создающее прикладные или сетевые сервисы для клиентских приложений в сети; примером такого сервера является FTP-server;

- аутентификация на клиенте в клиент-серверном приложении; клиентское приложение создает интерфейс к серверному приложению; такое приложение может быть браузером либо традиционным клиентским приложением, устанавливаемом на всех компьютерах, с которых производится доступ к серверу; клиентские приложения не создают сетевых или прикладных сервисов; примером таких приложений являются Outlook, Internet Explorer; они используют механизм аутентификации, предлагаемый операционной системой; если необходимо, они должны поддерживать механизм аутентификации сервера;

- комбинированная клиент-серверная аутентификация; многие приложения нельзя целиком отнести к серверному, либо к клиентскому приложению, поскольку содержат и серверные и клиентские компоненты; в этом случае можно произвести декомпозицию приложения на клиентскую и серверную часть и анализировать требования к аутентификации раздельно для каждой части.

IV Критические уязвимости сервиса аутентификации

Критическими, т. е. позволяющими получить практически беспрепятственный доступ к приложению, являются следующие уязвимости:

- для аутентификации допускаются недействительные или просроченные сертификаты PKI;
- пароли передаются в незашифрованном виде;
- для хранения паролей используется ненадежный метод шифрования;
- пароли «по умолчанию» после разворачивания приложения не изменены;
- при вводе пароля он отображается в окне ввода;
- ауттентификационные данные хранятся в приложении;
- учетные записи пользователей не блокируются после заданного количества неудачных попыток ввода пароля;
- структуры памяти, содержащие аутентификационные данные, не очищаются после завершения сессии пользователя.

Достаточно серьезная уязвимость создается также при использовании слабых паролей.

V Способы аутентификации

Аутентификация с помощью инфраструктуры открытых ключей (PKI)

В настоящее время инфраструктура открытых ключей (Public Key Infrastructure, PKI) считается наиболее прогрессивной технологией для создания систем управления доступом пользователей к информационным ресурсам, в том числе для реализации решений по созданию «единой точки входа».

Основой такой архитектуры являются центры сертификации. С помощью сертификатов, в частности, может выполняться аутентификация пользователей.

В примере, приведенном на рис. 1, пользователь Web-приложения отсылает серверу аутентификационные данные – информацию, подтверждающую его подлинность. Эта информация состоит из сертификата пользователя и набора идентификаторов, подписанных закрытым ключом пользователя. Web-сервер пересылает эту информацию серверу приложений, который аутентифицирует пользователя. Сервер приложений отсылает аутентификационные данные *предопределенного* пользователя баз данных (БД) серверу баз данных. Администраторы приложений и баз данных аутентифицируются сервером приложений и баз данных соответственно.

В случае, если Web-сервер является общедоступным, требование PKI-аутентификации пользователя Web-приложения сервером приложений (шаги 1 и 2), может быть необязательным. В этом случае, аутентификация может отсутствовать вообще, либо выполняться на основе паролей. Для лиц с административными правами PKI-аутентификация необходима всегда.

Приложение должно проверить, что:

- срок действия сертификата не закончен;
- сертификат используется по назначению;
- сертификат выпущен доверенным центром сертификации;
- сертификат не находится в списке отозванных сертификатов.

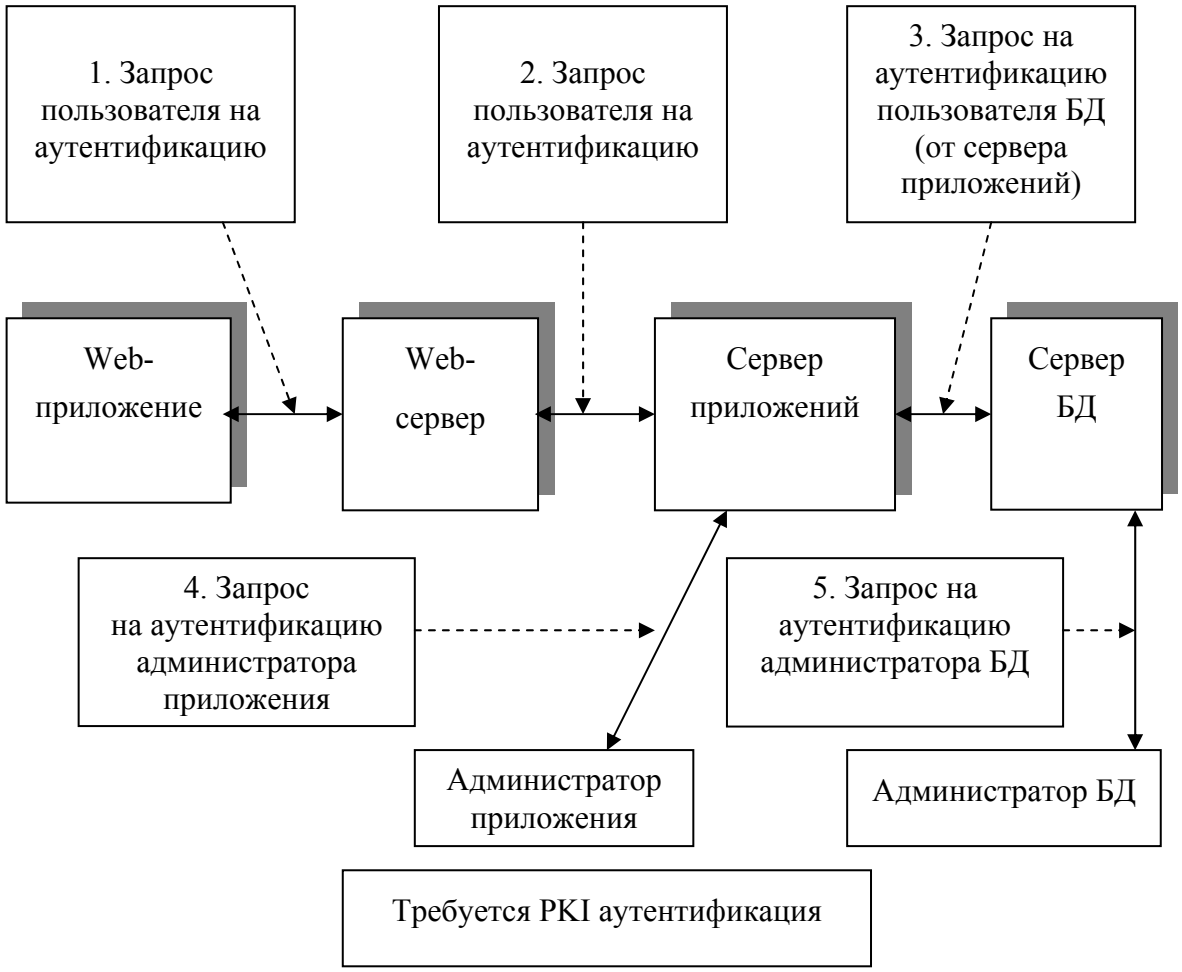

Рисунок 1 – Аутентификация пользователя при доступе к Web-серверу

Аутентификация с помощью паролей

В силу простоты реализации такой способ аутентификации широко распространен.

Аутентификация с помощью паролей может использоваться в приложениях, к которым не предъявляют высоких требований по безопасности. Кроме того, такой способ аутентификации может использоваться как дополнительная мера, для установления подлинности пары пользователь - сертификат в случае PKI-аутентификации.

В приведенном выше примере (рисунок 1) использование пароля как аутентификационной сущности возможно на этапах 1 и 2.

Пароль учетной записи пользователя защищенного приложения должен отвечать следующим требованиям:

- длина пароля – не менее 8 символов;
- пароль должен содержать символы верхнего и нижнего регистра клавиатуры, цифры и специальные (не алфавитно-цифровые) символы;
- пароль не должен содержать персональной информации пользователя (имя, номера телефонов, имя учетной записи, слова из словаря, набор повторяющихся символов и т. п.);
- максимальный срок жизни пароля не должен превышать 60 дней;
- в качестве нового пароля пользователь не должен использовать 10 предыдущих паролей;
- пользователь не должен иметь возможности менять свой пароль чаще, чем один раз в день; это ограничение не распространяется на пользователей с административными правами; они должны иметь право изменять свой пароль чаще, чем раз в день; кроме этого, они должны иметь возможность изменять пароли пользователя, если последний забыл его, либо пароль считается скомпрометированным.
- при смене пароля новый должен отличаться от предыдущего не менее чем на 4 символа.

Исходя из этих требований, механизм аутентификации защищенного приложения должен позволять администратору:

- назначать пользователям пароли и устанавливать срок их жизни;
- устанавливать режим необходимости смены пароля пользователем при первом доступе к приложению;
- устанавливать режим проверки истории паролей, не позволяющий пользователям использовать предыдущие пароли;
- устанавливать ограничения на пароли: длина пароля, наличие символов в верхнем и нижнем регистре, неалфавитных символов;
- устанавливать ограничения на множества паролей, которые пользователь использовать не может: слова из словаря, набор повторяющихся символов, имена учетных записей и т. п;
- устанавливать допустимое количество неудачных попыток аутентификации; если пользователь превысил допустимое количество попыток, его учетная запись должна блокироваться на определенное время.

VI Хранение аутентификационных данных

Как отмечалось выше, критическая уязвимость приложения возникает, если аутентификационные данные (пароли, имена пользователей, секретные ключи) хранятся в исполняемых файлах, либо к их хранилищу имеют доступ пользователи. Что бы исключить встраивания аутентификационной информации, следует использовать возможности операционной системы для безопасного хранения конфиденциальной информации, тщательно контролировать доступ к файлам с такой информацией и использовать проверенные криптографические средства для их защиты.

Для проверки того, что пользователь знает пароль, следует использовать процедуру верификации, не требующую хранения пароля. В качестве сущности, используемой при проверке, можно использовать хеш пароля.

В защищенных приложениях при верификации рекомендуется использовать итерационную процедуру вычисления хеша:

$$H_0 = H(pwd, salt) \quad (1)$$

$$H_n = H(H_{n-1}, salt) \quad (2)$$

где salt – большое случайное число, n >= 100000 – количество итераций.

Хеш H_n хранится вместе со значением salt**.** В процессе верификации приложение повторят процедуру вычисления хеша и сверяет полученное значение H_n с хранимым.

Если все же выбран вариант, при котором приложение сохраняет пароль, то он должен быть зашифрован с помощью проверенного и надежного алгоритма. Ключ шифрования, значение salt и зашифрованный пароль должны храниться в защищенном месте, доступном только полномочным приложениям и пользователям. Степень защиты должна быть не ниже, чем степень защиты данных, к которым получит доступ пользователь, прошедший процедуру аутентификации.

Если приложение осуществляет передачу пароля пользователя по сети, то передаваться пароль должен только в зашифрованной форме.

VII Общие требования к реализации механизма аутентификации

Рассмотрим общие требования к реализации механизма аутентификации, не указанные в предыдущих разделах.

1. Аутентификация пользователей. Приложение должно проверить, что пользователь аутентифицирован, прежде чем получит доступ к важным данным или ответственным ролям. Нельзя аутентифицировать пользователя только по имени, обязательно должна предъявляться аутентифицирующая информация.

Если срок действия пароля пользователя истек, то он должен проходить процедуру аутентификации только после смены пароля.

Пользователь может менять только свой пароль. Лишь администратор приложения может изменять пароли других пользователей.

2. Аутентификация процессов. Приложение должно аутентифицировать любой процесс, программу, объект и иную сущность, которая взаимодействует с приложением от лица пользователя**.** Прикладные процессы, действующие от лица пользователей, должны быть аутентифицированы сервером, к которому они пытаются получить доступ. Уровень строгости аутентификации процессов должен быть не ниже, чем тот, который используется при аутентификации пользователей.

3. Аутентификационное предупреждение. Прежде чем аутентифицированный пользователь получит доступ к ресурсам приложения, он должен быть ознакомлен с такой информацией:

- к какой системе пользователь получает доступ;
- степень защиты приложением личной информации пользователя (privacy right);
- максимальный уровень конфиденциальности, с которым может работать пользователь;
- предупреждение о том, что действия пользователя регистрируются;
- мера ответственности пользователя за обрабатываемые данные.

Дополнительно может сообщаться:

- дата, время домен или адрес последнего подключения данного пользователя;
- количество неудачных попыток доступа после последнего удачного.

4. Выбор аутентифицирующих сущностей. Следует отдавать предпочтение аутентифицирующим сущностям, которые нельзя забыть и невозможно подделать, например, аппаратные токены, биометрию, одноразовые пароли, PKI или single side one (SSO - технология Микрософт).

5. Цепочки доверия. Цепочки доверия определяют отношения доверия между объектами инфраструктуры приложения (например, с помощью сертификатов). Для каждой пользовательской сессии или транзакции приложение должно убедиться, что установлена и поддерживается цепочка доверия между клиентом, сервером приложения и другими серверами.

6. Доверенный путь. Механизм аутентификации должен использовать доверенный путь, например криптографию. Этот путь должен инициироваться пользователем, а не приложением. Пароли и другая информация, используемая в процессе аутентификации должны шифроваться перед передачей их по сети. Надежность шифрования должна соответствовать степени конфиденциальности защищаемых данных.

7. Аутентификация групп/ролей. Если в приложении используется аутентификация на уровне групп или ролей, то прежде чем пользователь будет авторизован как член группы или роли, он должен пройти индивидуальную процедуру аутентификации.

8. База безопасности. Пароли и другая информация, используемая в процессе аутентификации, должны быть надежно зашифрованы перед сохранением. База сохраненной информации должна быть защищена от удаления или модификации.

9. Аутентификация лиц с административными полномочиями. Требования к аутентификации лиц с административными полномочиями должны быть более строгими (например, требования к длине и сложности пароля).

10. Аутентификация каждой сессии пользователя. Каждый раз, при инициализации новой сессии пользователь должен вводить пароль. Приложение не должно хранить пароль в файлах cookies, либо в программах (скриптах) на стороне клиента или сервера, позволяющим пользователю не вводить пароль при инициализации новой сессии.

*Литература: **1.** Ховард М., Лебланк Д. Защищенный код: Пер. с англ, - 2-е изд., испр. М.: Издательско-торговый дом "Русская Редакция", 2004. - 704 стр.: ил. **2.** ISO/IEC 9798-1:1997 Information technology -- Security techniques -- Entity authentication. **3.** Application security and development. Security technical implementation guide. V. 2, r.1, DISA 2008 **4.** G. McGraw, Software Security: Building Security In, Addison-Wesley Professional, 2006*

УДК 681.3

СИСТЕМА ЗАХИСТУ ІНФОРМАЦІЇ ВІД НСД. ВАРІАНТ РЕАЛІЗАЦІЇ АЛГОРИТМУ МНОЖИННОЇ АВТЕНТИФІКАЦІЇ

Вячеслав Василенко

Національний авіаційний університет

Анотація: ***Для систем захисту ієрархічних автоматизованих систем пропонується алгоритм множинної автентифікації з використанням можливостей програмно – технічних засобів управління доступом до ресурсів автоматизованих систем.***

Summary: ***For systems of protection the hierarchical automated systems is offered algorithm of multiple authentication with use of opportunities program – technical means of management of access to resources of the automated systems.***

Ключові слова: ***Технічний захист інформації, ідентифікація, автентифікація***.

І Вступ

Одним із різновидів організації взаємодії користувачів є наразі побудова сучасних автоматизованих систем як ієрархічних, розподілених. Це дозволяє забезпечити ефективний обмін різнорідною інформацією як в межах окремих робочих станцій, так і в межах локальних та розподілених мереж. Однак зручність використання таких мереж призводить і до можливостей несанкціонованого доступу до відповідних ресурсів