В целом, перечисленные библиотеки, фреймворки и технологии предоставляют широкий спектр возможностей разработчику – от поддержки различных операционных систем до развернутого набора инструментов для эффективного учета и контроля за временем сотрудников. Исходя из поставленных целей, были выбраны наиболее предпочтительные библиотеки и технологии, использование которых будет требовать минимального вложения ресурсов, как с финансовой точки зрения, так и с точки зрения сложности разработки.

В заключение можно отметить, что сейчас, системы подобного рода востребованы и активно развиваются. Каждая из систем имеет разный набор функций, положительных и отрицательных сторон.

ЛИТЕРАТУРА

1. Крейг УоллсSpring в действии. [Электронный ресурс] - Режим доступа: http://avidreaders.ru/read-book/spring-v-deystvii.html (дата обращения 15.12.2016).

2. Обзор библиотек, фреймворков и технологий для создания веб-приложений. [Электронный ресурс] - Режим доступа: https://compress.ru/article.aspx?id=9825 (дата обращения 20.03.2017).

УДК 004.588

Студ. В.С. Иконов
Науч. рук. Д.А. Радиванович
(кафедра программной инженерии, БГТУ)

ПРИЛОЖЕНИЕ «KEYKEEPER»

Прогресс подарил человечеству великое множество достижений, но тот же прогресс породил и массу проблем. Человеческий разум, разрешая одни проблемы, непременно сталкивается при этом с другими, новыми. Вечная проблема − защита информации. На различных этапах своего развития человечество решало эту проблему с присущими для данной эпохи особенностями. Изобретение компьютера и дальнейшее бурное развитие информационных технологий во второй половине 20 века сделали проблему защиты информации настолько актуальной и острой, насколько актуальна сегодня информатизация для всего общества.

Главная тенденция, характеризующая развитие современных информационных технологий — рост числа компьютерных преступлений и связанных с ними хищений конфиденциальной и иной информации, а также материальных потерь. Любое современное пред-

приятие независимо от вида деятельности и формы собственности не в состоянии успешно развиваться и вести хозяйственную деятельность без создания на нем условий для надежного функционирования системы защиты собственной информации.

Одним из способов защиты информации является ограничение доступа к информации и его предоставление только авторизированному пользователю, подтвердившему свою личность вводом уникальной, известной только ему информации, к примеру, логина и пароля. Было бы логично преобразовывать пароль пользователя в уникальную строку, не подлежащую обратному преобразованию, для защиты аккаунта пользователя в случае взлома злоумышленниками базы данных. Это позволяет нам делать хеширования — механизм преобразования входного массива данных произвольной длины в выходную битовую строку фиксированной длины. Также хорошей идеей является преобразовывать информацию, хранящуюся внутри аккаунта пользователя в целях ее сокрытия от неавторизированных лиц, но с возможностью возврата её в вид, в котором её задал пользователь, что позволяет технология шифрования. Для хеширования применим алгоритм MD5, а для шифрования — Advanced Encryption Standard.

MD5 (англ. *Message Digest 5*) — 128-битный алгоритм хеширования, разработанный профессором Рональдом Л. Ривестом из Массачусетского технологического института в 1991 году. Предназначен для создания «отпечатков» или дайджестов сообщения произвольной длины и последующей проверки их подлинности. Широко применялся для проверки целостности информации и хранения хешей паролей. Ниже представлен алгоритм его работы:

Шаг 1. Выравнивание потока

Сначала к концу потока дописывают единичный бит. Затем добавляют некоторое число нулевых бит такое, чтобы новая длина потока L ′ {\displaystyle L'} стала сравнима с 448 по модулю 512 L ′ = 512 × N + 448 {\displaystyle L'=512\times N+448} . Выравнивание происходит в любом случае, даже если длина исходного потока уже сравнима с 448.

Шаг 2. Добавление длины сообщения

В конец сообщения дописывают 64-битное представление длины данных (количество бит в сообщении) до выравнивания. Сначала записывают младшие 4 байта, затем старшие. Если длина превосходит

22 64 − 1 {\displaystyle 2^{64}-1} , то дописывают только млад-

шие биты. После этого длина потока станет кратной 512. Вычисления

будут основываться на представлении этого потока данных в виде массива слов по 512 бит.

Шаг 3. Инициализация буфера

Для вычислений инициализируются 4 переменных размером по 32 бита и задаются начальные значения шестнадцатеричными числами (порядок байтов little-endian, сначала младший байт):

A = 01 23 45 67; // 67452301h

B = 89 AB CD EF; // EFCDAB89h

C = FE DC BA 98; // 98BADCFEh

D = 76 54 32 10. // 10325476h

В этих переменных будут храниться результаты промежуточных вычислений. Начальное состояние ABCD называется инициализирующим вектором.

Определим ещё функции и константы, которые нам понадобятся для вычислений. Потребуются 4 функции для четырёх раундов. Введём функции от трёх параметров — слов, результатом также будет слово:

1-й раунд: $\mathrm{FunF}(X,Y,Z)=(X\wedge Y)\vee(\neg X\wedge Z)$ FunF (X , Y , Z) = (X ∧ Y) ∨ (¬ X ∧ Z) {\displaystyle \operatorname {FunF} (X,Y,Z)=(X\wedge Y)\vee (\neg X\wedge Z)} ,

2-й раунд: $\mathrm{FunG}(X,Y,Z)=(X\wedge Z)\vee(\neg Z\wedge Y)$ FunG (X , Y , Z) = (X ∧ Z) ∨ (¬ Z ∧ Y) {\displaystyle \operatorname {FunG} (X,Y,Z)=(X\wedge Z)\vee (\neg Z\wedge Y)} ,

3-й раунд: $\mathrm{FunH}(X,Y,Z)=X\oplus Y\oplus Z$ FunH (X , Y , Z) = X ⊕ Y ⊕ Z {\displaystyle \operatorname {FunH} (X,Y,Z)=X\oplus Y\oplus Z} ,

4-й раунд : $\mathrm{FunI}(X,Y,Z)=Y\oplus(\neg Z\vee X)$ FunI (X , Y , Z) = Y ⊕ (¬ Z ∨ X) {\displaystyle \operatorname {FunI} (X,Y,Z)=Y\oplus (\neg {Z}\vee X)} ,

Каждый 512-битный блок проходит 4 этапа вычислений по 16 раундов.

Шаг 4. Вычисление в цикле

Заносим в блок данных элемент *n* из массива 512-битных блоков. Сохраняются значения A, B, C и D, оставшиеся после операций над предыдущими блоками (или их начальные значения, если блок первый).

AA = A

BB = B

CC = C

DD = D

Результат вычислений находится в буфере ABCD, это и есть хеш. Если выводить побайтово, начиная с младшего байта A и закончив старшим байтом D, то мы получим MD5-хеш.

Advanced Encryption Standard (AES), также известный как Rijndael (Рэндал) — симметричный алгоритмшифрования (размер блока 128 бит, ключ 128/192/256 бит), принятый в качестве стандарта шифрованияправительством США. Процесс шифрования основан на работе со сложными алгебраическими структурами, главную роль в которых играют матрицы. AES является блочным шифром − разновидностью симметричногошифра, оперирующего группами бит фиксированной длины − блоками. Поэтому можно использовать во избежание появления одинаковых 16-байтовых блоков данных в шифротексте режим сцепления блоков шифротекста — один из режимов шифрования для симметричного блочного шифра с использованием механизма обратной связи. Каждый блок открытого текста побитово складывается по модулю 2 (операция XOR) с предыдущим результатом шифрования. Но вопрос, откуда брать данные для первой операции xor, первому блоку? Для этого нужен вектор инициализации — случайноечисло, которое используетсядляинициализацииалгоритмашифрования. Его участие в алгоритме отображено на рисунке.

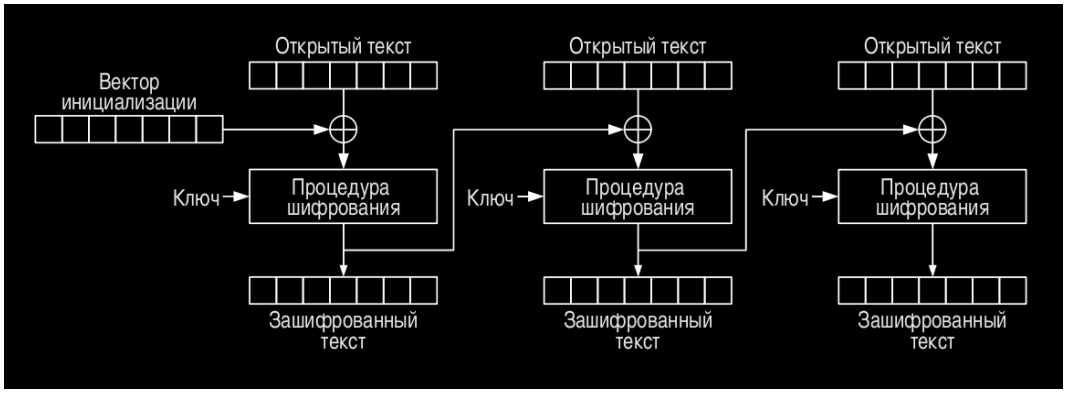

Рисунок – Схема работы режима режим сцепления блоков шифротек-

Но вопрос, откуда брать данные для первой операции xor, первому блоку? Для этого нужен вектор инициализации — случайноечисло, которое используетсядляинициализацииалгоритмашифрования и заранее задается пользователем.

ЛИТЕРАТУРА

1. AdvancedEncryptionStandart . [Электронные ресурс] – Режим доступа:

https://ru.wikipedia.org/wiki/Advanced_Encryption_Standard(дата обращения 25.03.2018)

2. MD5. [Электронные ресурс] – Режим доступа: https://ru.wikipedia.org/wiki/MD5 (дата обращения 16.03.2018)

3. Как устроен AES. [Электронные ресурс] – Режим доступа: https://habrahabr.ru/post/112733/(дата обращения 20.03.2018)

УНК 004.71

Студ. М.А. Левин
Науч. рук. доц. Д.В. Шиман
(кафедра программной инженрии, БГТУ)

ОЦЕНКА ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ ВНЕДРЕНИЯ И ИСПОЛЬЗОВАНИЯ СИСТЕМЫ ТЕРМИНАЛЬНОГО ДОСТУПА К УДАЛЁННОЙ ИНФОРМАЦИОННОЙ ИНФРАСТРУКТУРЕ

В настоящее время наблюдается активное внедрение информационных технологий практически во все сферы деятельности человека. Многие организации нуждаются в одновременной работе в разных городах или даже странах. При этом так же становится актуальными вопросы об организации удалённых рабочих мест для специалистов, находящихся в командировках или для специалистов, работающих удалённо. В этот момент очень выгодно иметь гибкую корпоративную инфраструктуру, легко и быстро расширяемую в соответствии с требованиями. Обычно такие системы базируются на виртуализации и туннелировании.

Виртуализация (сервера) − возможность запустить на одной физической машине несколько изолированных друг от друга виртуальных машин, каждая из которых считает, что работает на отдельной физической машине. Возможности виртуализации в таких решениях используются для выделения пользователям виртуальных машин, с которыми они смогут работать. При этом, сами машины будут храниться на сервере виртуализации предприятия.

Туннелирование в таких средах используется для организации системы доступа к удалённой локальной сети, чтоб, впоследствии, защищённо передавать данные между пользователем и его личной виртуальной машиной. Так же, решения, построенные на VPN, позволяют сократить количество необходимых внешних (публичных) IP-адресов для доступа ко всему пулу машин.