КОМПЬЮТЕРНАЯ ИНЖЕНЕРИЯ И ТЕХНИЧЕСКАЯ ДИАГНОСТИКА

УДК 681.326

АЛГОРИТМ HMAC — ЦИФРОВАЯ ПОДПИСЬ В РЕАЛЬНОМ МАСШТАБЕ ВРЕМЕНИ ДЛЯ ПРОТОКОЛА IPv6

ФРАДКОВ С.А.

Описывается алгоритм симметричной цифровой подписи HMAC. Проводится сравнительный анализ скоростных качеств реализаций функций хеширования, применяемых в алгоритме HMAC и претендующих на статус “стандартных”, при их использовании в ассоциациях безопасности протоколов IPsec/IPv6. Тесты проводятся на различных аппаратных платформах, употребляются реализации на языках C и Ассемблер.

1. Введение

С каждым годом увеличиваются размеры и популярность сети Internet. Принципы построения Internet активно применяются при построении Intranet-сетей масштаба предприятия. TCP/IP – наиболее используемый сетевой протокол в мире, несмотря на многие недостатки. С 1992 года, когда Internet был открыт для коммерческой деятельности, ежегодно увеличивается объем продаж всевозможной продукции в Internet. Развитие бизнеса заставляет различные предприятия опубликовывать сведения о себе или заполнять разные регистрационные карточки. Поэтому достаточно актуальными проблемами в Сети являются защита конфиденциальной информации и ограничение доступа к ресурсам Internet посредством аутентификации пользователей.

Протокол безопасности IPsec разработан IETF (Internet Engineering Task Force) для обмена конфиденциальной информацией в сетях TCP/IP и был изначально предназначен как опциональный элемент системы на базе протокола IPv4. В новом же протоколе IPv6, который полностью заменит IPv4 ориентировочно в 2008 году, реализация протокола IPsec является обязательной. IPv6/IPsec содержит механизмы безопасности, позволяющие выполнять аутентификацию/шифрование пакетов на “сетевом” уровне (по семиуровневой модели OSI). Криптографические алгоритмы для IPsec/IPv6 должны удовлетворять двум основным критериям: надежность/стойкость и производительность [1-3].

Протокол IPv6 добавляет к заголовку каждого пакета данных заголовок аутентификации, позволяющий на приемном конце определить источник/целостность пакета [4]. Классическая несимметричная цифровая подпись не способна работать в режиме реального времени. Под работой в режиме реального времени понимается в данном случае способность алгоритма формировать/верифицировать цифровую подпись пакета протокола IP со скоростью, сравнимой со скоростью обработки неподписанных IP-пакетов на конкретном компьютере. Именно поэтому в качестве стандартных средств цифровой подписи для IPv6 были предложены симметричные алгоритмы подписи, называемые MAC (Message Authentication Code). Развитие алгоритмов MAC привело к созданию HMAC (Hash Message Authentication Code), который является в настоящее время стандартом для использования в IPv6 и в других системах, требующих работу в реальном масштабе времени.

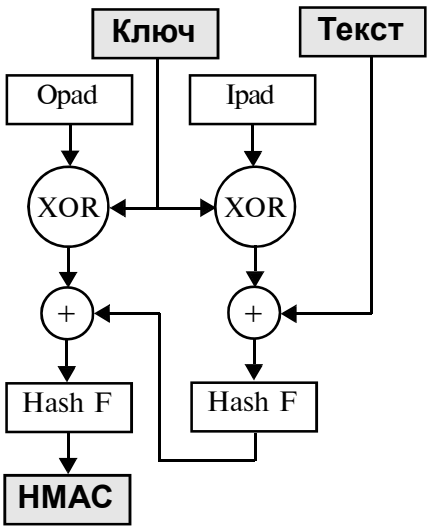
2. Алгоритм HMAC

Алгоритм HMAC построен на сочетании любой итеративной хеш-функции (например MD5 или SHA-1) и секретного ключа, употребляемого для генерации и проверки цифровой подписи. Основные достоинства использования функций хеширования заключаются в следующем:

– применение без модификаций существующих хеш-функций (в частности оптимизированных программных реализаций доступных хеш-функций);

– сохранение исходной производительности хеш-функций без существенного замедления работы;

– использование доказуемой стойкости хеш-функций.



Для алгоритма HMAC [6] необходима итеративная криптографическая хеш-функция (H) и секретный ключ (K). Функция хеширования оперирует при вычислении базовым блоком данных длиной в B байтов (для функций MD5 и SHA-1 B=64) и производит хеш длиной L байтов (для MD5 L=16, для SHA-1 L=20). Секретный ключ K может быть произвольной длины, при использовании ключа K длиной, превышающей B, его значение хешируется и в качестве ключа используется результирующая строка H(K) длиной в L байтов. В любом случае рекомендуемая минимальная длина ключа K — L байтов.

В алгоритме определяются две константы Ipad (B повторений байта 0x36) и Opad (B повторений байта 0x5C), ключ K (или значение H(K) в случае len(K)>B) дополняется нулевыми байтами до длины B байтов и значение HMAC вычисляется по формуле HMAC=H(Rxor Opad, H(K xor Ipad, Text)) (рисунок). На рисунке операция конкатенации обозначена окружностью с плюсом.

Стойкость и производительность механизма HMAC полностью зависит от свойств используемой функ-

ции хеширования H. HMAC предполагает наличие закрытого канала связи или использование специального протокола (ISAKMP, Oakley) для обмена секретными ключами.

В зависимости от применяемой функции алгоритм будет иметь название HMAC-MD5 или HMAC-SHA.

3. Исследование скоростных качеств функций хеширования

Интерес к самостоятельному исследованию скоростных характеристик функций хеширования MD5 и SHA-1 был вызван проблемой, сформулированной в работе [5], в которой автор проанализировал "узкие" места алгоритма MD5, привел абсолютные показатели скорости MD5 для различных компьютерных архитектур (табл.1) и в заключение сделал вывод о неспособности MD5 быть базовой функцией хеширования для протокола IPv6 из-за низкой скорости работы на существующем (1995 год) оборудовании и о необходимости заменить MD5 на что-нибудь другое в спецификации IPv6.

Таблица 1

CPU	CPU (MHz)	Cache (KB)		Speed (Mbit/s) MD5
		L1	L2	
Sun SPARCstation-2	40	-	64	13,8
Sun SPARCstation-10	50	16	1024	38,3
Sun SPARCstation-20	75	16	1024	58,9
Intel 486	66	8	-	33,3
Intel Pentium	90	8	512	46,7

Интерес еще более усилился после изучения солидной и основательной книги Б.Шнайера [9]. В разделе, посвященном функциям хеширования, он приводит результаты своих тестов на Intel 486SX-33 (табл.2). Результаты не могли не вызвать глубочайшего удивления: во-первых, скорость была приведена в Мбайт/с (что было отнесено на счет опечаток и исправлено на Мбит/с, так как скорость в 174 Мбайт/с – это слишком быстро: опечатка идентична в оригинальной американской и в переводной французской редакции книги), а во-вторых, скорости были диаметрально противоположными по отношению к [5] (сравните 174 Мбит/с на i486SX/33 МГц и 33,3 Мбит/с на i486/66 МГц). Получалось, что зря волновался J.Touch, и у MD5 есть большой запас по производительности.

Таблица 2

CPU	CPU (MHz)	Speed (Mbit/s)	
		MD5	SHA-1
Intel 486SX	33	174,0	75,0

Итак, я понял, что "верить нельзя никому", и занялся собственными тестами, а именно: решил протестировать скорость двух наиболее популярных функций хеширования MD5 и SHA-1. Реализация MD5 на языке C была взята из [7] и слегка прооптимизирована, а алгоритм SHA-1 был написан на C на основе [8]. Тесты проводились на аппаратных платформах Sun и Intel в Лаборатории исследований по информатике провинции Lorrain в г. Нанси (Франция) под управлением операционной системы Unix (SunOS 4.x/5.x на Sun SPARC и FreeBSD 2.2.6 на Intel). Исходный код на C компилировался на каждой рабочей станции с максимальной оптимизацией под конкретный процессор. В реализации каждого алгоритма буфер оперативной памяти размером 8 Кб циклически инициализировался последовательностью байтов (0x00, 0x01, ... 0xFF) и 1000 раз вычислялось значение хеш-функции с сохранением результатов от предыдущего вычисления. Во время экспериментов не проводилось тестирование скорости алгоритмов при обмене "память-диск", так как пакеты IP (теоретическая скорость обработки которых тестировалась) никогда не участвуют в подобном обмене. Результаты тестов приведены в табл. 3. Очевидно, что уже младшие модели семейства PentiumII позволяют получить на слабооптимизированной C-программе скорость, превышающую 100 Мбит/с, но так же очевидно, что в [9] указаны неправдоподобные результаты.

Таблица 3

CPU	CPU (MHz)	Speed (Mbit/s)	
		MD5	SHA-1
Sun SPARCstation-2 sun4c		14,2	5,5
Sun SPARCstation-5 sun4m		33,4	13,2
Sun SPARCstation-10 sun4m		33,8	13,3
Sun SPARCstation-20 sun4m		52,2	20,3
Sun Ultra-1 sun4u		76,1	30,5
Sun Ultra-2 sun4u		109,0	42,5
Intel Pentium MMX	200	86,3	33,6
Intel Pentium II	233	108,9	51,3
Intel Pentium II	266	123,7	58,6

В свете изложенного выше немалое удивление вызвала одна из отечественных публикаций, посвященных функциям хеширования [10]. Авторы привели таблицу скоростей из [9] как достоверные сведения о возможностях MD5 и SHA-1 и, кроме того, поместили таблицу с данными своих результатов (табл.5). Трудно представить себе язык программирования и компилятор, которые надо использовать для получения таких низких показателей (сравните данные из табл. 1, 4, 6 и табл. 5). Результаты, приведенные авторами публикации на Pentium-133 (табл. 5), в 3 раза уступают показателям, полученным на Pentium-133 (табл. 6), и в 2 раза усвтупают данным для Pentium-90 (табл. 1,4)).

Логичным продолжением исследований явился вопрос о том, насколько увеличит абсолютную скорость алгоритмов их реализация на ассемблере, и какой процентный выигрыш получится по отношению к Си-реализации. Интересные результаты уже были достигнуты в работах бельгийских ученых [11] (первая строка табл.4) и [12] (вторая строка табл.4). Суть предложенного метода (для архитектуры Intel)

Таблица 4

CPU	CPU (MHz)	MD5			SHA-1		
		Speed (Mbit/s)		Factor Asm/C	Speed (Mbit/s)		Factor Asm/C
		C	Asm		C	Asm	
Intel Pentium	90	59,7	113,7	1,9	21,2	48,7	2,3
Intel Pentium	90	59,7	136,2	2,28	21,2	54,9	2,58

состоит в ручной оптимизации ассемблерного кода функций хеширования, в целях параллельного выполнения сразу двух команд на конвейерах U и V процессоров семейства Pentium и в замене булевых функций, описанных в стандартах, на функционально аналогичные, но более быстрые. Для получения ответа на сформулированную выше задачу реализации MD5 и SHA-1 были переписаны на ассемблер с использованием идей из публикаций [11] и [12], а реализации на C были сильно прооптимизированы для семейства процессоров Intel x86. Скорость полученных реализаций была протестирована на нескольких компьютерах под управлением операционной системы Red Hat Linux 5.2 (ядро 2.0.36). Достигнутые результаты представлены в табл. 6.

Таблица 5

CPU	CPU (MHz)	Speed (Mbit/s)	
		MD5	SHA-1
Intel Pentium	133	25,6	7,2

4. Заключение

Результаты проведенных тестов (табл. 6) убедительно показывают, что скоростные характеристики как MD5, так и SHA-1 достаточны для большинства современных систем (модемное подключение на скорости 56,6/128 Кбит/с, локальная сеть Ethernet - 10 Мбит/с и локальная сеть Fast Ethernet (100 Мбит/с), но не достаточны для стандарта Gigabit Ethernet (1000 Мбит/с). Теоретическая пропускная способность сетей Ethernet – 35%, т.е. реальная пропускная способность сетей Fast Ethernet – 35-50 Мбит/с. Из опыта работы автор может сделать вывод, что пропускная способность сети не является ограничивающим фактором в скорости передачи данных. Главные проблемы создают винчестеры, основная масса которых способна "отдавать" данные всего лишь со скоростью 35-70 Мбит/с, процесс записи на винчестер в общем случае протекает еще медленнее. Максимальная достигнутая скорость в крупной корпоративной сети города Харькова, зафиксированная автором, была 94 Мбит/с и только в ситуации, когда 45 клиентских машин запустили процесс переиндексации базы данных FoxPro (не реляционная), сервер поместил 200 МБайт-базу в файловый кэш и "отдавал" данные из оперативной памяти.

Таким образом, из тестов видно, что выполнять вычисления хеша на скорости выше 100 Мбит/с алгоритм MD5 может и на древнем Intel Pentium-166. На современном разогнанном Intel Celeron алгоритм показывает просто фантастическую (для Fast Ethernet) скорость. Но кроме скорости нас интересует и стойкость. Уже доказано [13], что криптоаналитик, обладающий 10 миллионами долларов, может за 24 дня найти два фрагмента открытого текста, дающих один и тот же хеш MD5. Пока не удалось осуществить успешной атаки на функцию SHA-1 и, несмотря на то, что ее скорость в 2 раза ниже, чем у MD5, SHA-1 на 500 МГц-процессоре вычисляет хеш со скоростью 150 Мбит/с. Фирма Intel уже продемонстрировала процессор с частотой 1 ГГц и планирует серийный выпуск таких производительных микросхем.

Поэтому автор считает, что MD5 не соответствует своему положению стандартной функции для IPv6 не потому, что компьютеры не справятся с ее вычислением, а по прямо противоположному критерию - именно потому, что слишком быстро справятся. Ассоциации безопасности протокола IPv6 позволяют использовать любую хеш-функцию и любой алгоритм шифрования, поэтому пользователи могут применять ту функцию хеширования, которая наиболее подходит для данных конкретных задач, выбирая при этом либо скорость, либо стойкость.

Таким образом, проблема остается открытой – попытаться совместить скорость со стойкостью и найти среди существующих (создать новую) функцию хеширования со скоростью работы, сопоставимой с MD5 и стойкостью SHA-1.

Литература: 1. *Scott O. Bradler, Allison Mankin.* IPng. Internet Protocol Next Generation. Addison-Wesley, Reading, Massachusetts. 1995. 307 p. 2. *C. Huitema.* IPv6: The New Internet Protocol. Prentice Hall, Englewood Cliffs, New Jersey. 1996. 188 p. 3. *Stephen A. Thomas.* IPv6 and the TCP/IP Protocols. Wiley Computer Publishing. 1996. 481 p. 4. *Gisule Cizault.* IPv6: Théorie et pratique. O'Reilly, Paris. 1998. 285 p. 5. *Joseph D. Touch.* Performance Analysis of MD5. 1995. 92 p. 6. *H. Krawczyk, M. Bellare, R. Canetti.* HMAC: Keyed-Hashing for Message Authentication. [RFC 2104]. 1997. 121 p. 7. *R. Rivest.* The MD5 Message-Digest Algorithm. [RFC 1321]. 1992. 68p.8. NIST. Secure Hash Standard. [FIPS PUB 180-1]. 1995. 122p. 9. *Bruce Schneier.* Cryptographie appliquée, 2e édition. International Thompson Publishing France, Paris. 1997. 846p. 10. *Горбенко И.Д., Штанько И.А.* Функции хеширования. Понятия, требования, классификация, свойства и применение //Радиоэлектроника и информатика. 1998. №1. С.64-69. 11. *Antoon Bosselaers, Renй Govaerts and Joos Vandewalle.* Fast Hashing on the Pentium. 1996. 12. *Antoon Bosselaers.* Even Faster Hashing on the Pentium. 1997. 13. *P. Metzger, W. Simpson.* IP Authentication using Keyed MD5. [RFC 1828]. 1995. 132p.

Поступила в редколлегию 18.05.99

Рецензент: д-р техн. наук, проф. Хаханов В.И.

Фрадков Сергей Александрович, аспирант кафедры АПВТ ХТУРЭ. Научные интересы: сетевые технологии, защита информации. Увлечения и хобби: программирование, путешествия, иностранные языки. Адрес: Украина, 61726, Харьков, пр. Ленина, 14, тел: +38 (0572) 40-93-26.

Таблица 6

CPU	Bus (MHz)	CPU (MHz)	Cache (KB)		MD5			SHA-1		
			L1	L2	Speed (Mbit/s)		Factor Asm/C	Speed (Mbit/s)		Factor Asm/C
					C	Asm		C	Asm	
Intel 486DX4-100	33	100	8	-	34,55	41,66	1,21	9,98	14,19	1,42
Intel Pentium-120	66	133	8	256	85,62	103,48	1,21	23,43	29,59	1,26
Intel Pentium-150	66	166	8	256	105,22	128,34	1,22	29,07	36,57	1,26
Intel Pentium II-333	66	333	32	512	270,56	274,12	1,01	83,44	102,29	1,23
Intel Celeron-300a	110	495	32	128	403,27	408,49	1,01	124,25	151,30	1,22

é