ИСПОЛЬЗОВАНИЕ ГЕНЕРАТОРОВ ПСЕВДОСЛУЧАЙНЫХ ЧИСЕЛ ВО ВРЕДОНОСНОМ ПРОГРАММНОМ КОДЕ И СПОСОБЫ ЕГО ОБНАРУЖЕНИЯ

Хомутов Дмитрий Геннадьевич,
Ассистент, МИРЭА
Dragon96@yandex.ru

Аннотация. *Данная статья посвящена описанию возможных способов применения генераторов псевдослучайной последовательности во вредоносном коде. Основным объектом является генератор случайных чисел, а также методы работы с ним из кода. Приводится краткое описание современных типов генераторов и их возможное применение злоумышленником при разработке вредоносных программ. А также приведены способы обнаружения такого рода вредоносных программ современными антивирусными продуктами.*

Ключевые слова: *генератор псевдослучайных чисел, полиморфный вирус, вредоносный код, угроза, метаморфный вирус, антивирус.*

THE USE OF RANDOM NUMBER GENERATORS IN MALICIOUS SOFTWARE CODE AND HOW IT IS DETECTED

Homutov Dmitrii Gennadievich,
Assistant MIREA

Abstract. *This article is devoted to describing the possible application of pseudo-random sequence generator in malicious code. The main object is the random number generator, as well as how to work with them from the code. A brief description of the modern types of oscillators and their possible application in the development of an attacker malware. And also shows how to identify this type of malware modern antivirus products.*

Keywords: *pseudo random number generator, a polymorphic virus, malicious code threat, metamorphic virus, anti-virus.*

Введение

Псевдослучайным генератором (далее ГПСЧ) называют детерминированный полиномиальный алгоритм, позволяющий получить по короткому случайно выбранному аргументу, называемому затравкой (seed), существенно более длинную «псевдослучайную» последовательность битов. [3]

ГПСЧ на сегодняшний день используют повсеместно, начиная с криптографических протоколов и заканчивая пользовательскими приложениями, в качестве создания случайного поведения графических компонент и иные цели. ГПСЧ входят в состав набора функций современных операционных системах (ОС), содержащиеся как в крипто-, так и в обычных библиотеках.

С точки зрения защиты информации важную роль играет случайность бит в результирующем векторе, в то время как во вредоносных программах иногда требуется генерация последовательностей в соответствии с определенным законом или требуемым результатом.

1. Описание существующих генераторов

ГПСЧ классифицируются по алгоритму, лежащему в основе генерации случайных чисел. Поэтому в соответствии с [8] выделяют следующие классы:

1. На основе хеш-функций. На вход данному генератору подается набор данных полученные

из ОС. К этим данным добавляется несколько дополнительных параметров («соль», пользовательские строки и т.д.). И от всего этого набора данных вычисляется хеш-значение, которое и является результатом работы. Как правило, в основе таких генераторов используются устойчивые (вероятность нахождение прообраза крайне мала) к коллизиям хеш-функции.

2. На основе HMAC-функций. Это особый вид хеш-функций, значение которых зависит не только от функции, но и от ключа. На вход данному генератору подается также некоторый набор данных, полученный от ОС (в криптографии этими данными являются сообщения) и вырабатывается хеш с использованием выбранного ключа.
3. На основе симметричных шифров в режиме счетчика. Например. ГОСТ 28147- 89 в режим гаммирования.
4. На основе специальных однонаправленных функций, при работе которых вырабатываются случайные числа. Одним из примеров такого генератора является Dual EC DRBG (генератор на основе эллиптических кривых).

2. Применение генераторов

ГПСЧ реализованы в большинстве современных ОС и при необходимости можно получить весь их функционал из стандартных библиотек. Стоит учитывать тот факт, что в разных ОС доступ к генераторам осуществляется по разному: в Windows функции генерации реализованы в специальных библиотеках; в Linux ГПСЧ реализован в виде специального устройства.

При закрытие информации генераторы используются для выработки гаммы, которая накладывается на открытый текст. В результате получается зашифрованный текст, который возможно расшифровать, только зная начальное заполнение генератора[1].

Генераторы повсеместно используются в протоколах обмена данными для выработки общего ключа (SSL\TLS)[9]. Также их используют для генерации дополнительных параметров (“соли”, цифровой подписи) с целью защиты от типовых атак подмены сообщений[2].

Рассмотрим возможную последовательность действий нарушителем при использовании ГПСЦ для создания (модификации) вредоносных программ.

3. Использование ГПСЧ в полиморфных вирусах

ГПСЧ в полиморфных вирусах можно использовать различными способами. Однако сложность способ применения, не всегда влияет на сложность обнаружения вируса или отделения вредоносного от легального кода. Далее будет приведено несколько способов, которые могут встретиться в полиморфных вирусах.

Во-первых, простой полиморфизм – перестановка блоков (фрагменты, части) кода вируса. Тут случайная последовательность может использоваться как числовая последовательность, указывающая в каком порядке должны выстраиваться части вируса. При этом повторяющиеся блоки могут быть использованы для «замусоревания» кода и усложнения статического анализа на выявления основной ветви работы алгоритма.

Во-вторых. Генератор используется для выработки байтовой последовательности, которая впоследствии интерпретируется как набор машинных команд таким образом, чтобы не пострадал алгоритм работы тела вируса. В данном случае сгенерированный код является мусорным. Также некоторые инструкции могут не соответствовать инструкциям, которые допускаются к использованию в зараженной программе. В силу того, что разные компиляторы собирают по-разному исходный код, в результате этим сборкам может соответствовать вполне определенный список используемых инструкций, а также частота их встречаемости, как по отдельности, так и в составе набора команд. В результате вредоносный код будет без труда обнаружен на основании статистического анализа областей кода.

В-третьих. Генератор используется для выработки байтовой последовательности, которая используется для преобразования тела вируса в соответствии с определенными правилами, заложенными

[1] Не учитывая тот факт, что иногда генераторы могут отрабатывать несколько циклов в пустую.

в криптор[2]. Т.е. криптор состоит из обратимых инструкций, которые совершают арифметические преобразования над телом вируса с использованием определенных сгенерированных байт. Аналогичный алгоритм с обратными инструкциями (к использованным в криторе) и сгенерированные случайные байты закладываются в декриптор. В результате тело вируса гарантированно будет меняться от заражения к заражению. Однако сложность скрытия самого криптора позволяет антивирусам обнаруживать вредоносный модифицированный код.

Например (частный случай): случайная последовательность может использоваться как гамма, которая накладывается на вирус. При этом в декрипторе вируса всегда будет храниться инициализационный вектор генератора, чтобы восстановить гамму и тем самым расшифровать тело.

В-четвертых. Не совсем относится к ГСЧ, однако он во многом зависит от специального вида генератора, в котором учитывается вероятность встречи тех или иных машинных инструкций. В результате сгенерированный «мусорный код» не будет иметь лишнего и будет трудно отличим от легального.

В-пятых. Вирус может использовать выход с ГПСЧ для задания своего поведения на машине жертвы. Если функционал вируса содержит несколько возможных действий, то порядок их применения будет определяться некоторой функцией от полученной псевдослучайной последовательности (для затруднения детектирования и классификации вируса). Следует иметь виду, что данный пункт может применяться вместе с первым.

4. Доступ к ГПСЧ из вредоносного кода

В зависимости от типа генератора вредоносному коду необходимо совершить несколько вполне легальных, с точки зрения антивирусных программ, действий.

4.1. Подача генератору на вход параметра начального заполнения

Обычно начальным заполнением является строка пользовательских данных, дата, данные из буфера клавиатуры, снимок экрана (screenshot), данные от сетевого адаптера и др. Такой большой набор параметров призван гарантировать случайность выработанной последовательности. Однако, существуют вредоносные программы (ВП), которые генерируют заполнение ГПСЧ самостоятельно, используя при этом соответствующий набор функций ОС. Подобный метод можно рассматривать как демаскирующий признак и проявлять повышенный интерес (средствами антивирусной защиты) к подобному ПО.

В случае, если генератору не требуется начальный вектор заполнения, то задача выявления ВП усложняется.

4.2 Вызов различных функции генерации псевдослучайных последовательностей

ВП способна получить доступ к генератору несколькими путями, причем каждый из них имеет свои преимущества и недостатки, с точки зрения обнаружения. Первым и самым простым способом для реализации является загрузка нужной библиотеки ОС и обращение к функциям генератора.

Второй метод заключается в захвате конкретных функций cryptoApi, которые вызывают ГПСЧ. Однако подобный метод не является универсальным и легко обнаруживается.

5. Методы обнаружения

Выше (в пункте 3) были рассмотрены способы использования ГПСЧ во вредоносном коде, поэтому необходимо рассмотреть способы их обнаружения.

Что касается первого способа – использование блоков перестановок, то для обнаружения такого рода вредоносного программного кода достаточно обычного сигнатурного анализатора. Если один из блоков (или свертка из блоков) соответствует сигнатуре, то вирус всегда будет однозначного детектироваться. Однако существует возможность разбиения кода на очень маленькие блоки (что само по себе будет очень трудозатраным и повлечет генерацию вспомогательных таблиц переходов) для которых будет очень сложно составить сигнатуру.

Второй и четвертый способы также имеют малую степень угрозы, т.к. современные антивирусы способны отфильтровывать и сворачивать очевидные «мусорные» участки кода (команды типа «nop;», «inc ax; dec ax;» и т.д.). К тому же все подозритель-

[2] Участок кода, отвечающий за модификацию (в частности шифрование) тела вируса.

ные программы проходят через “песочницу”, где происходит динамическая отладка кода и весь мусорный код становиться очевидным. Отметим, что четвертой способ по сравнению со вторым устойчив к статистическому анализу, т.е. анализ при котором происходит проверка частоты встречаемости и соотношения различных команд относительно друг друга с целью выявления аномалий.

Выявление ВП, использующей для сокрытия третий способ не представляет особого труда. В случае, если вектор инициализации хранится в открытом виде, то для выявления вредоносного кода достаточно статического анализа. В случае, если вектор инициализации вырабатывается в процессе работы различных функций, то подмена адреса вызова процедуры инициализации ГПСЧ в таблице импорта ПО на функцию, реализованную в антивирусном продукте, позволяет получить необходимые данные.

Пятый способ является частным случаем нового поколения вредоносного программного обеспечения – метаморфные вирусы [10]. Эти вирусы опасны тем, что при каждом заражении они меняют свой алгоритм работы и как следствие свой программный код. Во время работ над статьей, автору не удалось обнаружить в открытых источниках публикаций, связанных с практическим применением подобного класса вирусов и заслуживающих внимания. Поэтому можно сделать вывод, что в силу сложности создания такого рода ВПО, появляются характерные признаки обычных вирусов, по которым существующие антивирусные продукты их обнаруживают.

Заключение

В данной статье рассмотрены различные способы использования ГПСЧ в полиморфных вирусах, а также возможные подходы по их обнаружению по характерным признакам современными антивирусными продуктами. Вызов генератора из вредоносного кода всегда проходит свободно и не вызывает подозрений у системы безопасности, за исключением методов перехвата api-вызовов. Предсказать параметры криптора или гамму выработанные генератором, не зная начальное заполнение, представляется не тривиальной инженерной и криптографической задачей.

Не смотря на выше изложенное, актуальной задачей для любого злоумышленника (вирусописателя) остается разработка метода скрытия криптора.

Список литературы

1. Вникаем в суть работы полиморфных генераторов // [Электронный ресурс] http://www.xakep.ru/magazine/xa/082/120/1.asp (дата обращения: 12.03.13)
2. Домарев В.В. Безопасность ИТ: Системный подход. – Изд: ТИД «ДС», 2008. – 994с.
3. Кузьминов Т.В. Криптографические методы защиты информации. – Новосибирск: Наука. Сиб. Предприятие РАН, 1998 г. – 194с.
4. Самоделов А. Криптография в отдельном блоке: криптографический сопроцессор семейства STM32F4xx // [Электронный ресурс] http://www.compeljournal.ru/enews/2012/6/4 (дата обращения: 8.03.13)
5. Хоглунд Г., Батлер Дж. Руткиты: внедрение в ядро Windows. – СПб.; Питер, 2007. – 285с.
6. Allasm.ru // [Электронный ресурс] http://allasm.ru/vir_11.php (дата обращения: 10.03.13)
7. Barker E., J. Kelsey. Recommendation for Random Number Generation Using Deterministic Random Bit Generators.//NIST Special Publication 800-90A, January 2012, - 136c.
8. Elaine Barker and John Kelsey. NIST Special Publication 800-90A. Recommendation for Random Number Generation Using Deterministic Random Bit Generators. Computer Security Division Information Technology Laboratory. January 2012. – 136 c.
9. RFC 5246 - The Transport Layer Security (TLS) Protocol Version 1.2
10. Szor P. Hunting for Metamorphic/ P.Szor, P.Ferrie// Virus Bulletin. - Sept, 2001. -P. 123-144.
11. VirusTotal // [Электронный ресурс] www.virustotal.com (дата обращения: 27.03.13)
12. WASM // [Электронный ресурс] http://wasm.ru (дата обращения: 15.03.13)
13. Варфоломей Собейкис. Азбука хакера 3. Компьютерная вирусология. — М.: Майор, 2006. - 512 с.