УДК 004.056:061.68

Е.П. Тумоян, Д.А. Кавчук

Технологический институт Южного федерального университета, Таганрог, Россия

МОДЕЛИРОВАНИЕ СЕТЕВЫХ АТАК В ЗАДАЧАХ АВТОМАТИЧЕСКОГО АНАЛИЗА ЗАЩИЩЕННОСТИ ИНФОРМАЦИОННЫХ СИСТЕМ

В работе предложен новый метод и программное средство оптимизации автоматической проверки уязвимостей удаленной информационной системы. Оптимизация основана на построении вероятностного дерева атак и оптимизации обхода графа с использованием искусственных нейронных сетей. Программное средство интегрировано со сканером уязвимостей Nessus и средством эксплуатации уязвимостей Metasploit Framework.

Ключевые слова: *эксплуатация уязвимостей; граф атаки; искусственная нейронная сеть.*

Введение

Существующие в настоящее время сканеры уязвимостей обеспечивают обнаружение и идентификацию уязвимостей на основании косвенных признаков. Для проверки (валидации) обнаруженной уязвимости используются средства эксплуатации, такие как Immunity Canvas или Metasploit Framework. Одной из основных проблем является необходимость массовой проверки уязвимостей. Ручная проверка большого количества уязвимостей требует от эксперта большого количества рутинных операций. С другой стороны, средства автоматической проверки (например, auto_pwn в Metasploit) выполняют последовательный запуск эксплойтов с учетом простых критериев, таких как сервис, целевая операционная система, ранг эксплойта. Вследствие этого процесс проверки занимает значительное время. Целью данной работы является разработка метода и программного средства оптимизации процесса автоматической проверки уязвимостей удаленной информационной системы. Предлагаемый в работе метод оптимизирует поиск средств эксплуатации уязвимостей на основе использования вероятностного дерева атак и искусственной нейронной сети с учетом специфики операционной системы, под управлением которой работает исследуемая ПЭВМ.

Разработанный метод позволяет провести проверку уязвимостей за меньшее количество попыток эксплуатации при заданной доверительной вероятности обнаружения всех известных уязвимостей. Кроме того метод также позволяет проводить проверку многостадийных атак (например, атак связанных с эксплуатацией уязвимости в клиентском сервисе с последующим локальным повышением привилегий).

Существующие средства и академические разработки

Существующие средства анализа безопасности включают два типа средств: средства обнаружения уязвимостей (такие как Nessus Security Scanner, MaxPatrol, Nexpose) и средства тестирования на проникновение (Immunity Canvas, Metasploit Framework и другие). Средства тестирования на проникновение помимо основного назначения могут использоваться для проверки (валидации) уязвимостей, аудита паролей, модификации старых и создания новых эксплойтов и т.д.

Необходимость в проверке уязвимостей возникает поскольку сканеры уязвимостей демонстрируют ложные срабатывания. Вероятность ложных срабатываний снижается при использовании агентов внутри анализируемой системы, которые предоставляют точную информацию о конфигурации и состоянии системы: такие средства есть в продуктах Retina или MaxPatrol. Однако в ряде случаев использование агентов невозможно по техническим или организаци-

онным причинам. В таком случае актуальной становится задача оптимизации проверок уязвимостей с использованием информации, полученной сканером уязвимостей. В существующих системах данная проблема решается с учетом простых критериев, таких как ОС, сервис и ранг эксплойта. Соответственно, процесс проверки уязвимостей занимает значительное время и фактически плохо контролируем.

В настоящее время существует ряд академических работ, позволяющих построить модель эксплуатации уязвимостей (или более широко – сетевой атаки) и оценить эффективность различных вариантов атаки.

Данные модели используют разную математическую базу, но большинство из них основаны на конечных автоматах и представляют атаку как последовательность состояний автомата.

Деревья атак. Модель предложена Б. Шнайером в работе [1] в 1999 году. Деревья атаки представляют собой концептуальные диаграммы, которые описывают угрозы системе и возможные атаки, направленные на их реализацию. Деревья атак предоставляют возможность для введения оценок каждого шага по некоторым критериям, например, по времени выполнения, числу операций, оценочной стоимости и т.д. При этом последовательность шагов может быть оценена на основании критериев для каждого шага. В работе [1] приведены примеры оценки деревьев на основе двух критериев, однако, нет принципиальных препятствий для разработки алгоритма многокритериальной оценки.

Однако модель не обеспечивает средств для включения условий внешней среды – некоторого набора входных данных.

Улучшенные деревья атак. Модель описана в работе [2] в 2006 году. Модель опирается на использование деревьев и механизма конечных автоматов для моделирования уязвимостей протоколов и систем. Таким образом, модель представляет собой расширение и уточнение модели деревьев атак. Авторами приводится пример анализа безопасности протокола IEEE 802.11. Данная модель имеет те же недостатки, что и предыдущая.

Графы атак. Модель предложена коллективом исследователей в работе [3] в 2002 году. Видимо, модель графов атак основана на расширении модели деревьев атак. По сравнению с деревьями графы атак имеют следующие особенности:

- чвляются специализированным средством для описания сетевых атак;
- для описания используется граф, а не дерево;
- узлы графа представляют не концептуальные действия, а узлы сети, процессы программы, конфигурационные файлы, участки кода, переменные и т.д.

Моделирование систем на основе графов атак основано на конечных автоматах. Переходы между узлами осуществляются на основе применения детерминированных правил. При этом может учитываться текущее значение некоторых параметров системы, переменных и т.д. Целью является достижение необходимой вершины. Модель может предусматривать анализ условий, необходимых для достижения цели атаки, поэтому графы атак могут использоваться для оценки безопасности программной системы или компьютерной сети. Модель получила широкое распространение [3,4], поскольку основана на простой и хорошо исследованной математической базе (конечные автоматы), сама достаточно проста и очевидна. Данная модель была реализована в нескольких инструментальных средствах, например в Lockheed Martin ANGI, а также средстве AttackGraph Tool 0.5 [5].

Недостатком данной модели является то, что, как и в предыдущем случае, она является средством для оценки сложности нарушения безопасности информационной системы, а не моделирования и исследования атак. Другим существенным недостатком данной модели является применение аппарата конечных автоматов. Модель реальной информационной системы в большинстве случаев не ограничивается моделью автоматов, поскольку включает плохо контролируемые или скрытые факторы.

Interacting State Machines. Модель предложена коллективом исследователей в работе [6] в 2002 году. Модель основана на применении специального типа высокоуровневых конечных автоматов Interacting State Machines (ISM) для моделирования сложных систем. В данной работе ISM применяются для моделирования атакуемого протокола с целью обнаружения ошибок, приводящих к уязвимостям системы, т.е. решаются задачи, существенно отличные от моделирования атак.

Метод оптимизации проверки уязвимостей

При разработке метода мы используем модель улучшенных деревьев атак, дополненную средствами анализа условий внешней среды и входных данных.

Предлагаемый метод можно сформулировать следующим образом:

1) Получение информации об исследуемой системе – семейство, версия, service pack, список и баннеры сервисов.

$$osInf = \{S, Ver, SP, serv, servB\},$$

где S — семейство ОС; Ver = {ver_i} — конечное множество предположительных версий ОС; SP = {sp_i} — конечное множество предположительных service pack ОС; serv = {serv_i} — конечное множество сервисов; servB = {servB_i} — конечное множество баннеров сервисов.

2) Получение информации об уязвимостях.

$$vulnInf = \{vulns\},$$

где vulns — конечное множество уязвимостей системы.

3) Построение дерева атак для целевой системы. Узлы дерева представляют собой эксплойты, ребра – вероятности перехода от одного эксплойта к другому. Дерево включает все возможные эксплойты

для данной операционной системы и возможные переходы. Построение дерева выполняется с использованием эвристического алгоритма на основе информации об уязвимостях системы.

$$G = (V, E),$$

где V — множество вершин: V = {exp_i}, exp_i — множество допустимых эксплойтов: exp_i = F(osInf, vulnInf); E — множество ребер: E = {Pi}, Pi — множество вероятностей перехода между эксплойтов, выходной сигнал нейронной сети: Pi = neuronet(vulnInf).

4) Уточнение дерева атак. С использованием информации о системе производится оценка вероятности срабатывания эксплойта. Для оценки вероятности могут использоваться различные техники, в данной работе мы используем искусственную нейронную сеть:

$$pi = neuronet(osInf).$$

5) Обход дерева атак. Производится обход в глубину и усечение дерева атак. Мы обходим дерево от корня к листьям по пути с наибольшей условной вероятностью срабатывания эксплойтов и переходов между ними.

6) При обработке очередного узла мы запускаем эксплойт с наиболее подходящими для данной операционной системы и набора сервисов параметрами. Если очередной узел не сработал, то мы производим усечение всего поддерева с корнем в данном узле.

7) Построение последовательностей эксплойтов. Мы строим цепочки сработавших эксплойтов с указанием возможностей, которые может получить атакующий при выполнении эксплойтов из цепочки.

На рис. 1 представлен пример работы предложенной модели.

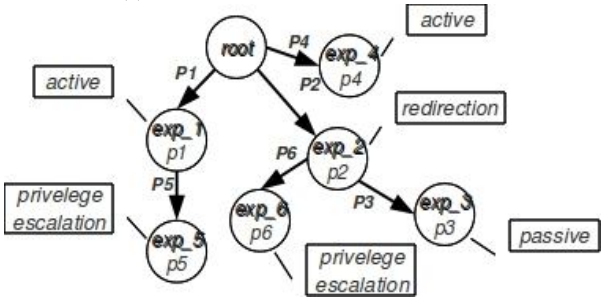

Рис. 1. Пример предложенной модели

Для решения задачи генерации вероятностей успешной реализации эксплойтов была выбрана искусственная нейронная сеть на основе радиальных базисный функций [7].

Такие сети относятся к классу обучаемых с учителем и решают задачу классификации в пространстве высокой размерности.

На вход сети поступает вектор свойств, ассоциированный с рассматриваемым состоянием дерева. Обучение сети производится на основе экспертной оценки, уточненной при построении конкретного дерева атаки.

Архитектура системы

Разработанная система включает следующие взаимодействующие компоненты: сканер уязвимостей Nessus Security Scanner, сервер системы Metasploit Framework и анализатор, разработанный в среде Matlab и взаимодействующий с другими компонентами системы. Архитектура системы представлена на рис. 2.

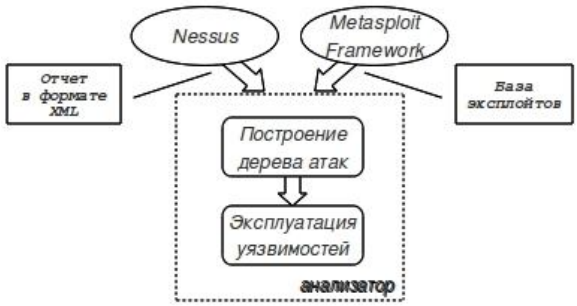

Рис. 2. Архитектура разработанной системы

Nessus Security Scanner

Nessus применяется для получения информации об исследуемой системе, а именно информации об уязвимостях системы, о типе и версии операционной системы, об открытых портах и активных сервисах, предоставляемых системой.

В настоящее время данная информация получается в ручном режиме и сохраняется в виде отчета сканера в формате XML, и затем этот отчет в качестве входного параметра передается программе-анализатору.

Metasploit Framework

Анализатор взаимодействует с сервером Metasploit Framework в автоматическом режиме.

Взаимодействие осуществляется посредством протокола XML-RPC. После подключения к серверу запрашивается база доступных эксплоитов, которая передаётся скрипту для дальнейшего анализа. Впоследствии системе Metasploit Framework отдается команда на запуск определенного эксплойта и получение результатов работы эксплойта.

Анализатор

Первым этапом работы анализатора в соответствии с предлагаемым методом является сканирование целевой системы. Сканирование производится с использованием сканера уязвимостей Nessus с целью определения типа и версии операционной системы, под управлением которой работает ПЭВМ, открытых портов и сервисов, работающих на ПЭВМ, и уязвимостей исследуемой системы. В настоящее время сканирование производится в ручном режиме, результаты импортируются в систему анализа. Однако, нет теоретических ограничений на получение информации от другого сканера.

Вторым этапом является получение базы эксплойтов из Metasploit Framework. Данный этап выполняется в автоматическом режиме.

Далее следует построение дерева атаки. Алгоритм построения дерева атаки базируется на модели деревьев атак с использованием искусственной нейронной сети для решения задачи генерации вероятностей срабатывания узлов построенного дерева и эвристическим алгоритмом построения связей между узлами. Первым этапом построения дерева атаки является выбор подходящих для эксплуатации целевой системы модулей на основе анализа полученной на предыдущих этапах информации. Вторым этапом построения дерева атаки является генерация вероятностей срабатывания выбранных на первом этапе эксплойтов посредством нейронной сети.

Выполняется визуальное отображение построенного дерева с цветовой индикацией вероятности срабатывания эксплойта. На данном этапе производится обход дерева в глубину. Для каждого узла дерева запрос на запуск эксплойта с необходимыми параметрами отсылается серверу Metasploit Framework. Производится проверка успешности срабатывания эксплойта. В случае, если эксплойт не сработал, все дерево, лежащее ниже, отсекается. В визуальном интерфейсе узлы помечаются как не сработавшие. В случае срабатывания эксплойта узел помечается как успешный.

Результаты тестирования системы

Разработанная система строит дерево атаки полностью, но затем запускает только активные эксплоиты. Разработан, но еще не интегрирован в систему запуск пассивных эксплоитов.

В ходе работы была подобрана следующая размерность сети: 8 нейронов входного слоя, 50 нейронов скрытого слоя и 1 нейрон выходного слоя.

Для обучения сети используется выборка из приблизительно шестидесяти векторов, сформированная синтетически.

В рамках тестирования системы для анализа эффективности ее работы были также проведены атаки на ПЭВМ с помощью инструмента автоэксплуатации, предоставляемого Metasploit Framework. Результаты проведенных экспериментов представлены в табл. 1.

Таблица 1

Результаты экспериментов

ОС	Metasploit Framework		Разработанная система	
	Количество опробованных эксплоитов	Первая открытая сессия	Количество опробованных эксплоитов	Первая открытая сессия
Windows XP SP3	102	22	10	5
Windows XP SP2	51	22	10	5
Windows 2000	51	46	10	5

Как видно из представленных результатов, предложенный метод доказал свою эффективность в сравнении со стандартным средством автоэксплуатации Metasploit Framework.

ПЭВМ, работающие под управлением Windows Vista и Windows 7, проэксплуатировать не удалось. Причиной является отсутствие серверных средств эксплуатации для данных операционных систем в Metasploit Framework.

В системе также реализована возможность эксплуатации систем Linux.

При некоторых доработках разработанная система может успешно применяться для анализа безопасности удаленных информационных систем, поскольку имеет ряд преимуществ перед существующими средствами:

- Анализирует и учитывает в своей работе информацию об операционной системе и сервисах исследуемой ПЭВМ.
- Рассчитывает вероятности срабатывания эксплоитов и запускает наиболее “вероятные” эксплоиты первыми.
- Строит дерево атаки.

Дальнейшие исследования

В рамках дальнейшего усовершенствования системы предполагается реализовать следующее.

Во-первых, производить построение и обход графа атаки, что включает запуск не только активных эксплоитов, но и пассивных, а также реализацию многостадийных атак.

Во-вторых, реализовать подключение к Nessus по протоколу XML-RPC для автоматического сканирования целевой системы.

Предполагается, что по завершении работы над системой, станет возможным ее использование для анализа безопасности даже без проведения стадии эксплуатации уязвимостей исследуемой системы.

Список литературы

1. Schneier B. Attack Trees [Электронный ресурс] /Brus Schneier // Dr. Dobb's Journal, 1999. Режим доступа: http://www.schneier.com/paper-attacktrees-ddj-ft.html.

2. Camtepe, S.A. A Formal Method for Attack Modeling and Detection [Электронный ресурс] /Seyit Ahmet Camtepe, Bulent Yener // TR-06-01, Rensselaer Polytechnic Institute, Computer Science Department. 2006. Режим доступа: http://citeseer.ist.psu.edu/751069.html.

3. Sheyner, O. Automated Generation and Analysis of Attack Graphs /Oleg Sheyner, Joshua Haines, Somesh Jha, Richard Lippmann, Jeannette M. Wing // Proceedings of the IEEE Symposium on Security and Privacy. Oakland, CA, USA, 2002. P. 273 – 284.

4. Jha, S. Two Formal Analyses of Attack Graphs /S. Jha , O. Sheyner, J. Wing// Proceedings of the 15th IEEE Computer Security Foundations Workshop. Nova Scotia, Canada, June 2002. P. 49-63.

5. Sheyner, O. AttackGraph Tool 0.5 [Электронный ресурс]. Режим доступа: http://www.cs.cmu.edu/~odobzins/scenariograph/as_files/AttackGraph-0.5.tar.gz

6. Von Ohiemb, D. Formal security analysis with Interacting state machines /David Von Ohiemb, Volkmar Lotz, Dieter Gollmann, Karjoth Günter, Michael Waidner// Lectute Neotes in Computer Science, 2002, № 2502. P. 212-228.

7. Хайкин С. Нейронные сети: полный курс, 2-е издание: Пер. с англ. – М. Изд. дом "Вильямс", 2006. – 1104 с.

Поступила в редколегію 22.03.2012

Рецензент: д-р техн. наук, проф. И.И. Турулин, ТТИ ЮФУ, Таганрог, Россия.

МОДЕЛЮВАННЯ МЕРЕЖЕВИХ АТАК В ЗАВДАННЯХ АВТОМАТИЧНОГО АНАЛІЗУ ЗАХИЩЕНОСТІ ІНФОРМАЦІЙНИХ СИСТЕМ

Е.П. Тумоян, Д.А. Кавчук

Представлені метод і програмний засіб оптимізації автоматичної перевірки уязвимостей видаленої інформаційної системи.

Ключові слова: *експлуатація уязвимостей; граф атаки; штучна нейронна мережа.*

A DESIGN OF NETWORK ATTACKS IS IN THE TASKS OF AUTOMATIC ANALYSIS OF PROTECTED OF THE INFORMATIVE SYSTEMS

E.П. Tumoyan, D.A. Kavchuk

A method and programmatic mean of optimization of automatic verification of brittleness of the remote informative system is presented.

Keywords: *exploitation of brittleness; count of attack; artificial neuron network.*