УДК 004.056.53

О.В. Полубелова

АРХИТЕКТУРА И ПРОГРАММНАЯ РЕАЛИЗАЦИЯ СИСТЕМЫ ВЕРИФИКАЦИИ ПРАВИЛ ФИЛЬТРАЦИИ

Полубелова О.В. **Архитектура и программная реализация системы верификации правил фильтрации.**

Аннотация. В статье описывается общая архитектура системы верификации правил фильтрации межсетевого экрана, а также рассматриваются аспекты программной реализации этой системы. Реализация выполнена на основе применения метода «проверки на модели» (Model Checking). В качестве верификатора используется программная система SPIN. Также был разработан пользовательский интерфейс, который позволяет загружать данные о верифицируемой системе, правила политики фильтрации, управлять процессом верификации, а также в удобном виде представлять ее результаты. Кроме того, в предлагаемой системе реализована возможность применения различных стратегий разрешения аномалий.

Ключевые слова: архитектура системы верификации, сетевая безопасность: верификация, проверка на модели, аномалии правил фильтрации.

Polubelova O.V. **Architecture and software implementation of the system of verification of filtering rules.**

Abstract. The paper describes the general architecture of the system of verification of filter rules firewall and discusses aspects of the software implementation. The implementation is based on the method of "model checking". SPIN software system is used as a verifier. Also designed user interface is described in the paper. It allows to download data on verifiable system and filtering policy rules. The user interface includes elements for management verification process and presentation of its results. In addition, the proposed system allows using different strategies to resolve the anomalies.

Keywords: architecture of the verification system, network security, verification, model checking, filtering rule anomalies.

1. Введение.

Системы безопасности, основанные на политиках, не теряют своей популярности благодаря гибкости в управлении и удобству администрирования. В данной работе рассматриваются политики фильтрации, обеспечивающие управление потоками сетевого трафика. К устройствам, обеспечивающим решение этой задачи, относятся межсетевые экраны, которые работают на основе локальных политик. Настройка политик безопасности сети является сложной и подверженной ошибкам задачей из-за важности учета различных зависимостей правил, а также необходимости проверки согласованности новых правил со всей политикой. Эта сложность увеличивается по мере роста размера сети. Для успешного развертывания системы защиты и поддержания ее функционирования требуется проводить регулярный анализ политики

конфигурации всех сетевых устройств безопасности. Такой анализ позволяет выявить и разрешить различные конфликты и аномалии в политиках, которые могут привести к серьезным нарушениям безопасности и сетевым уязвимостям, таким как блокирование легитимного трафика, разрешение нежелательного трафика, а также небезопасные передачи данных. Например, согласно статистическим исследованиям ассоциации ICSA*, до 70% всех межсетевых экранов уязвимы из-за неправильной конфигурации и настройки.

Одним из методов, позволяющим снизить риски нарушения таких свойств безопасности, как целостность, доступность и конфиденциальность, является метод «проверки на модели» для верификации правил фильтрации. Метод «проверки на модели» применяется главным образом для верификации программного и аппаратного обеспечения, а также в различных других областях диагностики корректности функционирования компьютерных систем. Метод развился на базе исключительно строгих формальных математических теорий, в отличие от различных других методов тестирования и верификации. Необходимость получить формальное доказательство корректности верифицируемой системы существует в таких области применения, как критические инфраструктуры, медицинские системы, программное обеспечение для автомобилей и т.п. При обеспечении информационной безопасности компьютерных систем также существенно высокое качество верификации применяемых систем безопасности. Преимуществами данного метода являются его высокий уровень абстракции при представлении данных, что позволяет построить неперегруженную модель сложной системы компьютерной сети, выделяя лишь некоторые важные для верификации правил фильтрации функции, относящиеся к обработке сетевого трафика. Также к достоинствам высокого уровня абстракции относится возможность без существенной доработки повторно использовать разработанные модели для верификации различных компьютерных систем, защищенных межсетевыми экранами. Кроме того метод «проверки на модели» позволяет исследовать динамическое поведение системы, не включаясь в ее рабочий процесс.

Автор уже публиковал ряд статьей по данной тематике [1-8]. Эта работа посвящена главным образом архитектурным решениям и аспектам программной реализации системы верификации политики безопасности (СВПФ). В качестве верификатора используется программная система SPIN [9-13].

* URL: http://www.icsa.net

Статья организована следующим образом. В первом разделе проводится анализ релевантных работ. Во втором разделе описана предлагаемая архитектура СВПФ. Третий раздел посвящен программной реализации СВПФ. В заключении сформулированы результаты работы и основные направления дальнейших исследований.

2. Анализ современной литературы. Обзор литературы в этой статье посвящен главным образом программным системам, в которых выполняется различного вида анализ политики фильтрации, в том числе и на наличие аномалий.

В работе [14] рассматривается программная система для автоматической проверки выполнения конфигурации межсетевого экрана с возможностью настройки политики фильтрации и способа генерации трафика. В тестовой сессии на основе списка управления доступом (ACL), грамматики и в соответствии с пользовательскими профилями формируется большой набор различных политик.

В работе [16] рассматривается подход, который моделирует работу сети от начала до конца, включая систему контроля доступа, маршрутизаторы, межсетевые экраны и NAT. Модель представляет собой сеть в качестве конечного автомата, где заголовок пакета и местоположение определяет состояние. Переходы в этой модели определяются информацией из заголовка пакета. Семантика системы контроля доступа определяется на основе булевых функций с помощью бинарных диаграмм решений (BDDs). Затем используется логика деревьев вычислений (CTL) для определения всех возможных состояний этого пакета в сети, проверки доступности сети, а также требований безопасности. Этот подход реализован в программной системе под названием ConfigChecker.

Работа [17] также посвящена обнаружению аномалий в политиках фильтрации межсетевого экрана. Авторы разработали алгоритмы для обнаружения и исправления таких аномалий. Для каждого типа аномалий предлагается свой метод автоматической коррекции.

В работе [19] предлагается системно-структурный подход для тестирования политики фильтрации межсетевого экрана. Авторы определяют структурное покрытие (на основе критериев покрытия правилами и предикатами) политики межсетевого экрана на стадии тестирования. Для достижения высокого структурного покрытия авторы разработали четыре техники автоматизированной генерации пакетов: генерация случайных пакетов; техника, основанная на разрешении локальных ограничений; техника, основанная на разрешении глобальных ограничений, и техника, основанная на пограничных значениях. Такой

подход позволяет выявить различные нарушения в политике фильтрации еще на стадии конфигурирования.

В данной статье автор рассматривается аспект анализа аномалий фильтрации, который частично решается и в некоторых вышеперечисленных работах. Отличительной особенностью предлагаемого в данной работе подхода является применение метода «проверки на модели» с использованием линейной темпоральной логики (LTL, Linear temporal logic). Подобный подход также описан в работе [16], но авторы рассматривают специфические характеристики конфигураций сети, не анализируя в явном виде политику на предмет аномалий фильтрации.

В следующем разделе рассмотрим архитектуру системы верификации правил фильтрации.

3. Архитектура системы верификации правил фильтрации. СВПФ должна обеспечить отсутствие аномалий в политике безопасности. Для решения этой задачи, архитектура СВПФ должна содержать необходимые компоненты, обеспечивающее полное покрытие всех возможных противоречий с точки зрения алгоритмов их поиска и алгоритмов их разрешения или обхода.

Для поиска аномалий разработан модуль верификации, применяющий подход «проверки на модели». В отличие от задачи поиска задача разрешения противоречий включает в себя модификацию политики. Минимальная модификация заключается в деактивации некоторых из правил, в общем случае требуется также добавление и изменение правил политики. Физическая архитектура сети и, соответственно, ее ЯОС-описание (Язык описания системы) считается неприкосновенным. Модификация политики осуществляется СВПФ в двух режимах – пакетном и интерактивном.

В пакетном режиме СВПФ используется как библиотека, не имеющая интерфейса пользователя. В случае обнаружения конфликтов, СВПФ пытается их исправить автоматически при помощи стратегий разрешения по умолчанию и сообщает пользователю о результатах и принятых решениях. В соответствующем лог-файле выдается также список найденных аномалий.

В интерактивном режиме СВПФ выступает в роли отладчика политики, предлагая пользователю список найденных противоречий и способов разрешения. За один шаг администратор имеет возможность отладить политику для группы взаимно независимых противоречий, затем поиск конфликтов запускается снова.

Большинство стратегий разрешения реализуют алгоритм разрешения по умолчанию, который не зависит от модуля верификации, обнаружившего противоречие. Так, алгоритм, соответствующий стратегии deny take precedence (DTP), деактивирует разрешающее правило и ему не требуется обращение к модулю верификации, обнаружившему противоречие. Однако в сложных случаях решение задачи поиска содержит большую часть необходимых вычислений, требующихся для применения стратегии, и эти вычисления требуют применения методов, реализованных в модуле верификации. Поэтому, наряду с алгоритмом по умолчанию, модули верификации, как правило, реализуют свои собственные алгоритмы для каждой стратегии, соответствующей противоречию из области ответственности модуля.

На рисунке 1 представлена общая архитектура программной системы верификации правил фильтрации политики безопасности, на основе метода «проверки на модели».

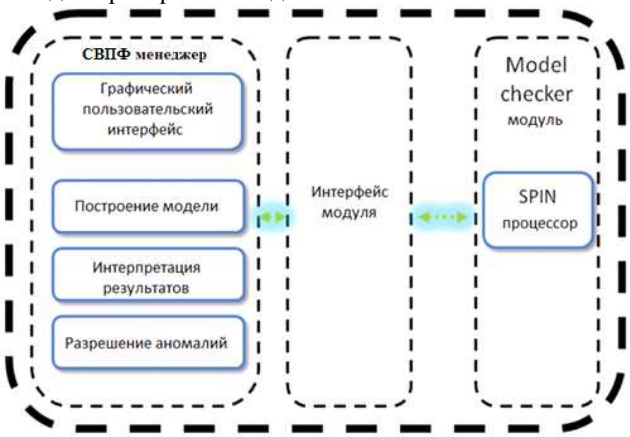

Рис. 1. Общая архитектура СВПФ

Система верификации состоит из трех частей: СВПФ менеджер, Интерфейс модуля, Model checker модуль. Компонент *СВПФ менеджер* предназначен для общего управления процессом верификации, предоставления графического пользовательского компонента (пакет *Графический пользовательский интерфейс*), формирования верифицируемой модели на основе входных данных (пакет *Построение модели*), интерпретации полученных результатов верификации (пакет *Интерпретация результатов*) и разрешения аномалий (пакет *Разрешение аномалий*).

Компонент «Графический пользовательский интерфейс» предназначен для взаимодействия с пользователем в графическом режиме. Пользовательский интерфейс позволяет задавать свойства верификации, проверять результаты и определять методы для разрешения обнаруженных конфликтов.

Компонент «Построение модели» отвечает за создание модели компьютерной системы на основании описания системы на языке ЯОС (язык описания системы) и модели системы безопасности, которая функционирует на основе политики, описанной на языке ЯОП (язык описания политик). Также этот компонент создает структуры обнаружения различных типов аномалий в верифицируемой модели. Компонент «Интерпретация результатов» преобразует результаты верификации в представление, стандартизированное в СВПФ.

Следующий компонент — Model checker модуль — обеспечивает запуск и вычисление верифицируемой модели, настройку всех параметров верификации. К ним относятся: необходимость обнаружения тупиковых состояний работы верифицируемой модели (т.е. когда система не может перейти ни в какое состояние); выбор типа спецификации – либо это свойства, которые должны всегда выполняться в системе (assert), либо свойства, которые никогда не должны выполняться (never), глубину поиска по состояниям, отчет о недостижимых состояниях и т.д. SPIN процессор вычисляет все возможные состояния модели для обнаружения конфликтов.

Входными данными для программной системы верификации является описание компьютерной системы, защищенной межсетевыми экранами.

Выходными данными являются информация об обнаруженных аномалиях и рекомендации для администратора политики безопасности.

4. Аспекты программной реализации.

На рисунке 2 представлена диаграмма основных классов СВПФ. На данном рисунке для наглядности опущены наследники классов ConflictTest и PolicyRule, обрабатывающие остальные категории правил политики: аутентификацию, фильтрацию, защиту каналов и операционные правила.

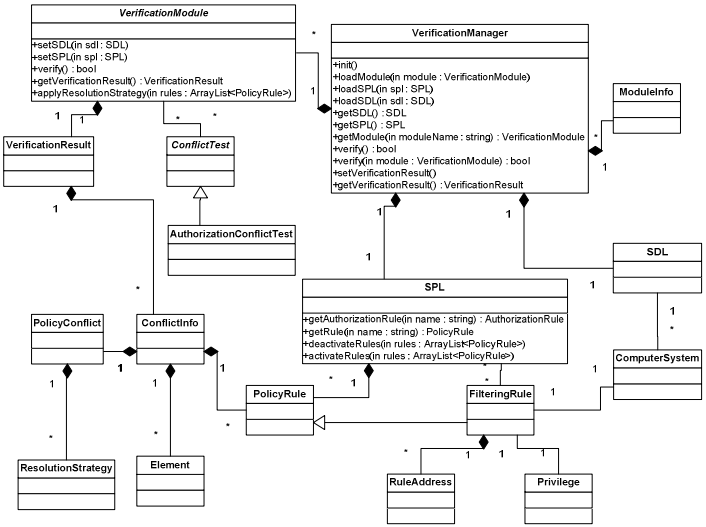

Рис. 2. Диаграмма классов СВПФ с примером классов для политики авторизации

На диаграмме показаны отношения между классами, а также указаны основные методы. Рассмотрим назначение этих классов.

Основные два класса СВПФ — это VerificationManager and VerificationModule. Класс VerificationManager вызывает модуль верификации для проверки политики безопасности. Модуль верификации, рассматриваемый в данной работе, основан на математическом методе «проверки на модели». Он наследуется от абстрактного класса VerificationModule. На схеме наследник класса VerificationModule опущен, т.к. будет рассмотрен более подробно ниже.

Класс VerificationResult содержит поле логического типа, которое используется для хранения результатов верификации, т.е. были ли обнаружены аномалии или нет.

Класс ConflictTest предоставляет общий интерфейс для процесса обнаружения конфликтов (аномалий). Наследники должны осуществлять обнаружение аномалий всех типов, например, FilteringAnomaliesTest.detect().

Класс PolicyConflict предназначен для описания конфликта, обнаруженного в политике, и ссылок на стратегии разрешения.

Класс ConflictInfo содержит типы и описания аномалий, набор правил, приводящих к аномалии, и возможные стратегии их разрешения.

Класс Element предназначен для описания элементов компьютерной системы. Он включает в себя описание сетевых узлов.

Класс SPL служит для представления верифицируемой политики безопасности.

Класс SDL задает представление SDL.

Класс ComputerSystem предназначен для представления компьютерной системы. Он соответствует аналогичному классу CIM модели и представляет описание хоста, сервера, интеллектуального сетевого устройства и т.д.

Класс ModuleInfo служит для описания модулей верификации.

Классы PolicyRule, FilteringRule, Role, Privilege предназначены для представления правил на языке SPL.

Класс PolicyRule описывает основной интерфейс для представления правил. Все детали реализуются наследниками, такими как Filtering Rule и др.. Класс AuthorizationRule реализует правила авторизации. Содержит список полей правила, привилегии, а также порядок правил в таблице доступа. Наследуется от абстрактного класса PolicyRule. Класс Privilege определяет полномочия на определенные действия.

SECManager включает Verification Manager класс. VerificationManager передает описание SDL и фрагменты SPL описания модулям верификации. Этот класс является фасадом СВПФ.

5. Заключение. В настоящей статье рассматривается общая архитектура СВПФ, а также аспекты ее программной реализации.

В разделе анализа современной литературы рассматриваются различные программные реализации средств проверки корректности политики фильтрации и работы сети в целом. Отличительной особенностью данной работы является ее акцент на обнаружении аномалий фильтрации и использование метода «проверки на модели». В качестве верификатора используется программная система SPIN.

Архитектура системы состоит из трех частей: СВПФ менеджер, Интерфейс модуля и модуль Model checker. Компонент СВПФ менеджер предназначен для общего управления процессом верификации и предоставления графического пользовательского компонента. Модуль Model checker обеспечивает запуск и вычисление верифицируемой модели, настройку всех параметров верификации.

С помощью разработанной СВПФ были проведены эксперименты, которые показали, что предлагаемый подход позволяет выявлять все аномалии в правилах фильтрации политики безопасности, однако имеет экспоненциальную вычислительную сложность в зависимости от количества верифицируемых правил. Таким образом, можно

заключить, что предложенная методика и реализованное программное средство могут быть использованы для малых и средних компьютерных сетей.

В дальнейшем планируется развить используемый подход для верификации других политик безопасности, например политики авторизации. Это позволить осуществлять комплексный подход функционирования сети.

Литература

1. *Котенко И.В., Юсупов Р.М.* Перспективные направления исследований в области компьютерной безопасности // Защита информации. Инсайд, № 2, 2006. С.46–57.
2. Полубелова О.В., Котенко И.В. Верификация правил фильтрации с временными характеристиками методом “проверки на модели” // Труды СПИИРАН. Вып.3 (22). СПб.: Наука, 2012. С.113-138.
3. *Kotenko I., Polubelova O.* Verification of Security Policy Filtering Rules by Model Checking // Proceedings of IEEE Fourth International Workshop on “Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications” (IDAACS’2011). Prague, Czech Republic, 15-17 September 2011. P. 706-710.
4. *Тишков А.В., Котенко И.В., Черватюк О.В., Лакомов Д.П., Резник С.А., Сидельникова Е.В.* Обнаружение и разрешение конфликтов в политиках безопасности компьютерных сетей // Труды СПИИРАН, Выпуск 3, Том 2. СПб.: Наука, 2006. С.102-114.
5. *Котенко И.В., Тишков А.В., Черватюк О.В., Лакомов Д.П.* Поиск конфликтов в политиках безопасности // Изв. Вузов. Приборостроение, Т.49, № 11, 2006, С.45-49.
6. *Котенко И.В., Тишков А.В., Черватюк О.В., Резник С.А., Сидельникова Е.В.* Система верификации политики безопасности компьютерной сети // Вестник компьютерных и информационных технологий, № 11, 2007. С.48-56.
7. *Котенко И.В., Тишков А.В., Сидельникова Е.В., Черватюк О.В.* Проверка правил политики безопасности для корпоративных компьютерных сетей // Защита информации. Инсайд, № 5, 2007. С.46-49; № 6, 2007. С.52-59.
8. *Черватюк О.В., Котенко И.В.* Верификация правил фильтрации политики безопасности методом “проверки на модели” // Изв. Вузов. Приборостроение, Т.51, № 12, 2008, С.44-49. ISSN 0021-3454.
9. *Holzmann G.* The Spin Model Checker Primer and Reference Manual // Addison-Wesley, 2003. P. 608.
10. *Manna Z., Pnueli A.* Temporal Verification of Reactive Systems: Safety // Springer-Verlag, New York, 1995. P. 530.
11. *Карпов Ю.Г.* Model checking. Верификация параллельных и распределенных программных систем // СПб.: БХВ, 2009. 560 с.
12. *Миронов А.М.* Математическая теория программных систем. http://intsys.msu.ru/staff/mironov/
13. On-The-Fly, LTL Model Checking with SPIN. http://netlib.bell-labs.com/netlib/spin/whatispin.html
14. *Ehab Al-Shaer, Adel El-Atawy, Taghrid Samak.* Automated pseudo-live testing of firewall configuration enforcement. IEEE Journal on Selected Areas in Communications V. 27(3). 2009. P. 302-314.
15. *Ting Yu; Dhivya Sivasubramanian; Tao Xie.* Security policy testing via automated program code generation // ACM International Conference Proceeding Series. 2009.

16. *Ehab Al-Shaer; Will Marrero; Adel El-Atawy; Khalid ElBadawi*. Network configuration in a box: Towards end-to-end verification of network reachability and security // Proceedings - International Conference on Network Protocols, ICNP. 2009. P. 123-132.
17. *Fei Chen; Alex X. Liu; Jeehyun Hwang; Tao Xie.* First step towards automatic correction of firewall policy faults // ACM Transactions on Autonomous and Adaptive Systems. V. 7(2). 2012.
18. *Linghao Zhang; Xiaoxing Ma; Jian Lu; Tao Xie; Nikolai Tillmann; Peli De Halleux*. Environmental modeling for automated cloud application testing // IEEE Software. V. 29(2). 2012. P. 30-35.
19. *JeeHyun Hwang; Tao Xie; Fei Chen; Alex X. Liu*. Systematic structural testing of firewall policies // IEEE Transactions on Network and Service Management.V. 9(1). 2012. P. 1-11.
20. *Alain J. Mayer, Avishai Wool, Elisha Ziskind.* Offline firewall analysis. Int. J. Inf. Sec. 5(3). 2006. P. 125-144.
21. *Khalid Al-Tawil, Ibrahim A. Al-Kaltham.* Evaluation and testing of internet firewalls. Int. Journal of Network Management 9(3): 1999. P. 135-149.
22. *Jan Jürjens , Guido Wimmel*. Specification-Based Testing of Firewalls (2001) // In proceeding of the 4th International Conference of perspectives of System Informatics (PSI'02). 2002. P. 308-316.
23. *Baier C., Katoen J.-P.* Principles of Model Checking // The MIT Press, 2008. P. 984.

Полубелова Ольга Витальевна — научный сотрудник лаборатории проблем компьютерной безопасности СПИИРАН. Область научных интересов: безопасность компьютерных сетей, включая управление политиками безопасности, верификация протоколов безопасности и систем безопасности, использование методов проверки на модели для обнаружения и разрешения конфликтов в политиках; онтологии в информационной безопасности, дескрипционные логики, СИЕМ-системы. Число научных публикаций — 25. ovp@comsec.spb.ru, www.comsec.spb.ru; СПИИРАН, 14-я линия В.О., д.39, Санкт-Петербург, 199178, РФ; р.т. +7(812)328-2642, факс +7(812)328-4450. Научный руководитель — Котенко И.В.

Polubelova Olga Vitalievna — researcher of Laboratory of Computer Security Problems, SPIIRAS. Research interests: computer network security, including policy management, verification of security protocols and security systems, model checking techniques for policy conflicts detection and resolution, ontology, description logic, SIEM systems. The number of publications — 25. ovp@comsec.spb.ru, www.comsec.spb.ru; SPIIRAS, 39, 14-th Line V.O., St. Petersburg, 199178, Russia; office phone +7(812) 328-2642, fax +7(812)328-4450. Scientific leader — I.V. Kotenko.

Поддержка исследований. В публикации представлены результаты исследований, поддержанные Министерством образования и науки Российской Федерации (государственный контракт 11.519.11.4008), грантами РФФИ, программой фундаментальных исследований ОНИТ РАН и проектами Седьмой рамочной программы Европейского Союза *SecFutur* и *MASSIF*.

Рекомендовано лабораторией проблем компьютерной безопасности СПИИРАН.
Заведующий лабораторией Котенко И.В., д-р техн. наук, проф.
Статья поступила в редакцию 05.03.2013.

РЕФЕРАТ

Полубелова О.В. **Архитектура и программная реализация системы верификации правил фильтрации.**

Системы безопасности, основанные на политиках, не теряют своей популярности благодаря гибкости в управлении и удобству администрирования. В данной работе предложен анализ политик фильтрации, обеспечивающих управление потоками сетевого трафика.

В статье описывается общая архитектура системы верификации правил фильтрации межсетевого экрана, а также рассматриваются аспекты программной реализации этой системы. Реализация выполнена на основе применения метода «проверки на модели» (Model Checking). В качестве верификатора используется программная система SPIN. Также был разработан пользовательский интерфейс, который позволяет загружать данные о верифицируемой системе, правила политики фильтрации, управлять процессом верификации, а также в удобном виде представлять ее результаты. Кроме того, в системе реализована возможность применить различные стратегии разрешения аномалий.

Рассматриваемая методика реализована в виде программного прототипа и успешно используется в рамках проекта Европейского союза MASSIF.

SUMMARY

Polubelova O.V. **Architecture and software implementation of the system of verification of filtering rules.**

Policy-based security systems do not lose their popularity due to the management flexibility and administration purposes. The work is devoted to analysis of filtering policy. This policy provides control of the network traffic.

The paper describes the general architecture of the system of verification of filter rules firewall and discusses aspects of the software implementation. The implementation is based on the method of "model checking". SPIN software system is used as a verifier. Also in the paper is described designed user interface. It allows to download data on verifiable system and filtering policy rules. The user interface includes elements for management verification process and presentation of its results. In addition, the proposed system allows using different strategies to resolve the anomalies.

The implemented software prototype successfully used in the framework of the European Union project MASSIF.