МОДЕЛИРОВАНИЕ ОБНАРУЖЕНИЯ ИНФОРМАЦИОННЫХ АТАК НА ОСНОВЕ ТЕОРИИ КОНЕЧНЫХ АВТОМАТОВ

Гончаров В.В.[1], Гончаров А.В.[2], Мишенина О.В.[3]

Ключевые слова: *корпоративная сеть, сетевые технологии, сетевой запрос, информационная атака, информационная безопасность, конечный автомат, элемент, множество, подмножество элементов, графовая модель, случайная величина, функция и плотность распределения случайной величины.*

Аннотация

Цель работы: совершенствование научно-методических основ повышения качества контроля безопасности базовых протоколов сетевого взаимодействия на узлах корпоративной сети.

Методы: системный анализ и математический аппарат порождения и распознавания «правильно построенных» цепочек, образующих регулярные множества, задаваемые выражениями, позволяющими строить конечный автомат, допускающий в точности повторение цепочки соответствующего регулярного множества.

Результаты: разработана модель, описывающая базовый протокол сетевого взаимодействия узлов компьютерной сети, на автоматном языке P, каждый элемент которого соответствует базовому сетевому запросу и может быть корректно обработан аппаратно-программным комплексом сети; при этом язык P описывается с помощью конечных автоматов-распознавателей, выявляющих потенциально опасные запросы, элементы которых не соответствуют стандартам, описанным на базе данного языка. Обоснован вывод: включение в сигнатуру запроса времени инициирования и максимально допустимого времени его обработки позволит существенно сократить возможности реализации компьютерных атак данного типа.

Полученные результаты являются основой для создания соответствующего эффективного информационно-математического обеспечения аппаратно-программного комплекса безопасности сложных компьютерных сетей.

DOI: 10.21681/1994-1404-2023-1-41-51

Введение

Международная компания *Positive Technologies*, являющаяся крупнейшей организацией в сфере анализа защищенности [6] информационных систем, представила статистические данные об уязвимостях корпоративных информационных сетей: прикладное программное обеспечение — 56%; *web*-серверы — 44%; *web*-приложения — 13%; СУБД — 6%. То есть большинство уязвимостей в сети приходится на *web*-серверы, являющиеся неотъемлемой частью корпоративных сетей. На них могут быть реализованы как внешние, так и внутренние *web*-сервисы, приложения, базы данных, в которых может храниться конфиденциальная информация [1, 6, 8]. Часто почтовые серверы совмещаются с *web*-сервером. Около 28% успешных атак реализуются, используя уязвимости при эксплуатации *web*-приложений. В ходе нескольких внешних тестирований выявлены уязвимости, позволяющие в один шаг, без необходимости авторизации, удаленно выполнять команды операционной системы (ОС) с привилегиями *web*-приложения. Наиболее существенными атаками преодоления сетевого периметра являются: словарные пароли — 44%; уязвимости *web*-приложений — 28%; отсутствие актуальных обновлений — 16%; недостатки конфигурации — 8%; *web*-интерпретатор командной строки — 4%.

Общеизвестно, что *web*-сервисы и приложения работают с протоколами *HTTP* (*Hyper Text Transfer Protocol*):

[1] ***Гончаров Владимир Васильевич,*** *доктор технических наук, профессор, заслуженный работник высшей школы Российской Федерации, заведующий кафедрой математики Военной академии имени Петра Великого, г. Москва, Российская Федерация.*
E-mail: v_v_goncharov@mail.ru

[2] ***Гончаров Александр Владимирович,*** *соискатель Военной академии имени Петра Великого, г. Москва, Российская Федерация.*
E-mail: dinozavp@inbox.ru

[3] ***Мишенина Ольга Викторовна,*** *кандидат педагогических наук, доцент, профессор кафедры математики Военной академии имени Петра Великого, г. Москва, Российская Федерация.*
E-mail: o.v.mishenina@gmail.com

они составляют 50% от количества всех обрабатываемых протоколов на *web*-серверах. Поэтому актуальной практической задачей является разработка комплекса математических моделей защиты *web*-сервера [5, 9, 14], основанного на фильтрации *HTTP*-пакетов, не подходящих по своим семантическим параметрам под стандарты *HTTP*-запросов, описанных в стандарте *RFC* 2616 (*Request for Comments*).

Комплекс математических моделей имеет сложную иерархическую структуру взаимосвязанных программных модулей, обеспечивающих соответствующее управление доступом к сети, т. е. осуществляет идентификацию, аутентификацию и авторизацию запросов [1—3]. При этом одной из задач рассматриваемого комплекса является контроль легитимности поступающих на *web*-сервер сетевых запросов, в том числе информационных атак, маскируемых под запрос.

Поведенческая модель обнаружения информационных атак позволяет обнаружить атаку с помощью сетевых запросов, нарушающих базовый протокол сетевого взаимодействия на узлах корпоративной сети [4, 13].

Разработанная модель основывается на регулярном языке ***P***[4], представляющем собой формальный язык, удовлетворяющий определенным свойствам и позволяющий математически описывать механизмы порождения и распознавания «правильно построенных» цепочек, образующих регулярные множества[5]. В свою очередь, регулярные множества и регулярные выражения весьма близки. Но они представляют собой разные сущности: регулярное множество — множество, в общем случае бесконечное, а *регулярное выражение — это формула, схематично показывающая, как было построено соответствующее ей регулярное множество с помощью операций языка* (и эта формула конечна).

Американский математик Стивен Клини доказал[6], что каждый автоматный язык может быть задан формулой (регулярным выражением) и каждое регулярное множество может быть распознано конечным автоматом. Отсюда следуют два важных *следствия*:

– любой автоматный язык является регулярным множеством (или: для любого конечного автомата можно построить регулярное выражение, задающее распознаваемый этим автоматом язык);
– любое регулярное множество является автоматным языком (или: по любому регулярному выражению можно построить конечный автомат, допускающий в точности цепочки соответствующего регулярного множества).

Математической моделью процесса распознавания регулярного языка является вычислительное устройство, называемое конечным автоматом (КА). Термин «конечный» подчеркивает то, что вычислительное устройство имеет фиксированный и конечный объем памяти и обрабатывает последовательность входных символов, принадлежащих некоторому конечному множеству. Существуют различные типы КА; если функцией выхода КА (результатом работы) является лишь указание на то, допустима или нет входная последовательность символов, такой КА называют *конечным распознавателем*.

Формальная постановка задачи исследования

Разработанная модель основывается на регулярном (автоматном) языке ***P***, с помощью которого описывается базовый протокол сетевого взаимодействия узлов компьютерной сети[7]. Каждый элемент языка ***P*** соответствует базовому сетевому запросу и может быть корректно обработан программным обеспечением аппаратного комплекса на сети. Это значит, что такой запрос не является угрозой безопасности для сети. При этом язык ***P*** описывается с помощью конечных автоматов-распознавателей типа:

$$A =< S, X, Y, s_0, \delta, \gamma, F, s_a >,$$

где S — множество состояний; X — множество входных параметров; Y — множество семантических операторов, анализирующих данные на входе системы; $s_0 \in S$ — начальное состояние; $\delta: S \times X \to S$ — переходная функция; $\gamma: S \times X \to Y$ — функция распознавания семантических операторов, анализируемых на входе автомата; $F \subseteq S$ — множество конечных состояний, в которые переходит система при правильном распознавании элементов языка ***P***; $s_a \in S$ — конечное состояние, в которое переходит автомат при поступлении ей на вход элементов, не входящих в язык ***P***.

Предположим, что на вход конечного автомата для анализа поступает группа элементов, входящих в язык ***P,*** соответствующая сетевому запросу, поступающему на узел сети. Если после обработки группы элементов автомат перейдет в одно из конечных состояний F, это значит, что поступивший сетевой запрос может обрабатываться узлом сети, не несет в себе угрозу и не является элементом компьютерной атаки. Иначе, при переходе автомата в состояние s_a, необходимо констатировать факт обнаружения информационной атаки на сеть.

[4] Пентус А.Е., Пентус М. Теория формальных языков : учеб. пособие. М. : Изд-во ЦПИ при мехмате МГУ, 2004. 80 с.

[5] Суховеров В.С. Система автоматической обработки тематически ориентированных текстов с терминологическим словарем в формате регулярных выражений // Проблемы управления. 2019. Вып. 2. С. 41—46.

[6] Клини С.К. Введение в метаматематику. Математическая логика и рекурсивные функции // Физико-математическое наследие. М. : Юрайт, 2009. 528 с.

[7] Кудрявцев В.Б., Алешин С.В., Подколзин А.С. Теория автоматов : учебник. М. : Юрайт, 2018. 320 с.; Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологии, протоколы : учебник. СПб. : ИД «Питер», 2010. 944 с.

```
<Метод запроса> <Идентификатор ресурса> <Параметры доступа> <Версия HTTP> <CRFL>
[<Заголовок 1>:<Значение> <CRFL>
<Заголовок 2>:<Значение> < CRFL>
  ...
<Заголовок n>:<Значение> < CRFL>]
< CRFL>
```

Рис. 1. Формат HTTP–запроса

Этапы решения задачи

Примером поведенческой модели, в основе которой лежит работа конечных автоматов-распознавателей, служит типовая модель обнаружения компьютерной атаки на *web*-сервер. Известно, что конечный автомат будет выявлять потенциально опасные запросы, в данном случае это *HTTP*-запросы, элементы которых не соответствуют стандартам, описанным на базе языка P_{http}. На рис. 1 изображен формат *HTTP*-запроса, состоящий из нескольких полей и соответствующий стандарту *RFC* 2616[8].

Метод формирования *HTTP*-запроса: в стандартах *RFC* указаны основные методы запросов, такие как *POST, DELETE, GET, HEAD, PUT, TRACE, OPTIONS* [5, 9]. Идентификатор ресурса, которому направлен запрос, представляет собой запись в формате *URL* (*Uniform Resource Locator*). Параметры доступа используются для обработки входных данных на различных программах со стороны *web*-сервера.

Существуют различные версии *HTTP*-протокола, с помощью которого создан запрос: *HTTP* 0.9, *HTTP* 1.0, *HTTP* 1.1. Завершается первая строка запроса параметром *CRFL* (*CaReFuL mnemonic*), что означает переход на новую строку или возврат.

Далее в *HTTP*-запросе могут появляться строки с заголовками формата «Имя заголовка», «Значение», при этом разделение заголовков происходит при помощи параметра *CRFL*. Если в конце *HTTP*-запроса стоят два подряд символа *CRFL*, это значит, что запрос завершен.

Используя данную поведенческую модель, имеется возможность отфильтровывать в большом потоке данных определенные *HTTP*-запросы, которые могут предоставлять угрозу элементам сети, в частности, *web*-серверу. Такими могут быть запросы:

- не соответствующие синтаксису, стандартизированному в рамках стандартов RFC;
- которые не могут быть обработаны аппаратным комплексом сети на основе ПО;
- адресованные к несуществующим элементам web-сервера;
- с неподдерживаемой web-сервером версией протокола;
- с запрещенными заголовками.

Если в процессе анализа входящего трафика будет обнаружен хотя бы один из вышеперечисленных запросов, имеется основание констатировать факт обнаружения атаки на *web*-сервер. Чтобы отфильтровать такие *HTTP*-запросы, в разработанной модели будут использоваться конечный автомат $\mathrm{A}V_{http}$, который определяет язык P_{http}. Последовательные элементы языка P_{http} включают в себя все типы *HTTP*-запросов, направленные *web*-серверу. Последовательность символов X языка P_{http}, которая описывает *HTTP*-запрос, поступает на вход автомата $\mathrm{A}V_{http}$. Если в результате обработки последовательных элементов автомат переходит в конечное состояние F, то полученные *HTTP*-запросы не несут угрозы для сети. В противном случае будет определено существование информационной атаки.

Архитектура конечного автомата $\mathrm{A}V_{http}$, распознающего элементы языка P_{http}, состоит из пяти составных узлов (рис. 2):

1 — узел распознавания и анализа типа метода формирования *HTTP*-запроса;

2 — узел распознавания и анализа идентификатора ресурса;

3 — узел распознавания и анализа параметров доступа к ресурсу;

4 — узел распознавания и анализа версии *HTTP*-протокола;

5 — узел распознавания и анализа заголовков *HTTP*-запроса.

Для выполнения семантических операций в процессе работы конечного автомата используются следующие служебные переменные[9] [9]:

S_{method} — одномерный строковый массив, элементы которого содержат символьные идентификаторы разрешенных методов формирования *HTTP*-запросов;

L_{URL} — числовая переменная, определяющая максимальную длину идентификатора ресурса *web*-сервера;

S_{URL} — одномерный строковый массив, элементы которого содержат идентификаторы ресурсов, хранящихся на *web*-сервере;

L_{Nquery} — числовая переменная, определяющая максимальное количество параметров доступа к ресурсу *web*-сервера;

[8] URL: https://www.ietf.org/rfc/rfc2616.txt

[9] Кудрявцев В.Б., Алешин С. В., Подколзин А. С. Теория автоматов : учебник. М. : Юрайт, 2018. 320 с.

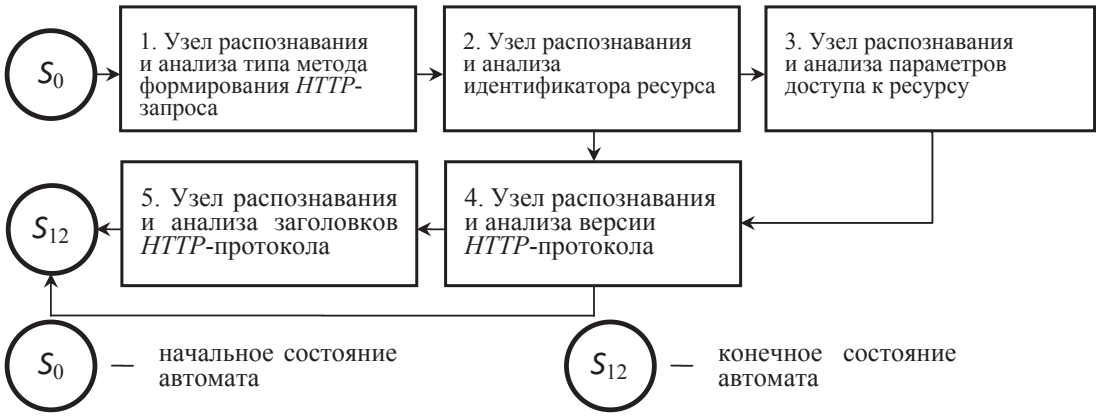

Рис. 2. Архитектура конечного автомата

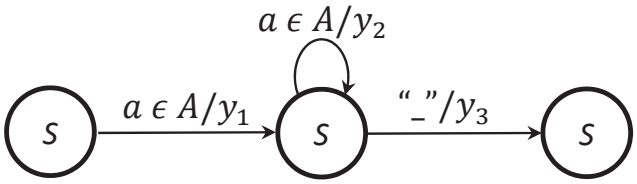

Рис. 3. Графовая модель узла распознавания и анализа типа метода формирования HTTP-запроса

$L_{VarLength}$ — числовая переменная, определяющая максимальную длину строкового имени параметра доступа к ресурсу *web*-сервера и имени заголовка *HTTP*-запроса;

$L_{ValLength}$ — числовая переменная, определяющая максимальную длину строкового значения параметра доступа к ресурсу *web*-сервера и значения заголовка *HTTP*-запроса;

$L_{NHeaders}$ — числовая переменная, определяющая максимальное количество заголовков в *HTTP*-запросе;

$S_{Versions}$ — одномерный строковый массив, содержащий номера версий протокола *HTTP*, которые могут обрабатываться защищаемым web-сервером;

$S_{Headers}$ — одномерный строковый массив, содержащий допустимые символьные имена заголовков *HTTP*-запроса, которые могут обрабатываться *web*-сервером;

Z — строковая переменная, предназначенная для временного хранения фрагментов анализируемого HTTP-запроса;

i, j ,k — числовые переменные, которые используются в качестве счётчиков.

Значения переменных Z, i, j, k формируются в процессе работы конечного автомата AV_{http}, а значения переменных L_{Nquery}, L_{URL}, $L_{VarLength}$, $L_{NHeaders}$, $S_{Versions}$, $S_{Headers}$ должны задаваться оператором перед началом работы автомата, с учётом требований стандартов *RFC* и специфики используемого ПО *web*-сервера. Для наглядности графового отображения[10] [3] блоков конечного автомата AV_{http} введём следующие обозначения: A — множество буквенных символов английского алфавита; N — множество, включающее в себя числовые символы от «0» до «9», символы «.», «#», «?», «/» и «%», а также символ подчеркивания; *NOP* — семантический оператор, не выполняющий никаких действий; «_» — пробельный символ; «*CRLF*» — символ, обозначающий возврат и перевод каретки на новую строку.

Описание условий перехода и семантические операторы, которые выполняются при каждом из возможных переходов в каждого узле конечного автомата AV_{http} (см. рис. 2), подробно представлены в Интернете[11] и соответствующих версиях *HTTP*-протокола.

Узел распознавания и анализа типа метода формирования HTTP-запроса первым начинает обработку входных символов, поступающих на вход автомата . Узел обеспечивает проверку того, что анализируемый *HTTP*-запрос сформирован на основе одного из методов, идентификаторы которых содержатся в переменной S_{method}. Графовая модель [10, 15] узла показана на рис. 3.

В первом блоке конечного автомата AV_{http} определено три состояния: $s_0, s_1, s_2 \in S$ и три семантических оператора: $y_2, y_3 \in Y$, которые выполняются при переходе автомата из одного состояния в другое. При условии перехода первого блока автомата AV_{http} в состояние s_2 обработка входных символов осуществляется узлом распознавания и анализа идентификатора ресурса.

[10] Гайдук А.Р., Плаксиенко Е.А. Анализ и аналитический синтез цифровых систем управления : учеб. пособие. М. : Лань, 2018. 272 с.

[11] URL: https://www.ietf.org/rfc/rfc2616.txt

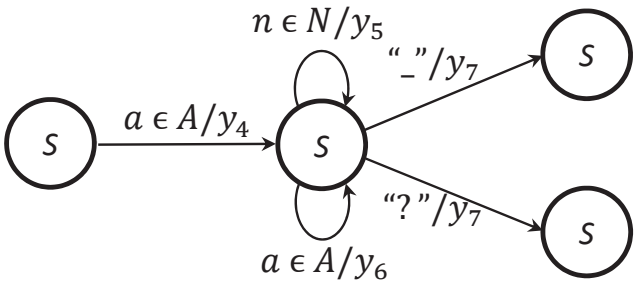

Рис. 4. Графовая модель распознавания и анализа идентификатора ресурса

Узел распознавания и анализа идентификатора ресурса предназначен для выделения из текста *HTTP*-запроса значения адреса ресурса и проверки его длины. Длина идентификатора ресурса, содержащегося в анализируемом *HTTP*-запросе, не должна превышать значения переменной L_{URL}. В процессе своей работы автомат узла также проверяет, что *HTTP*-запрос направлен одному из существующих ресурсов *web*-сервера, определённых в переменной. Графовая модель данного узла показана на рис. 4.

Во втором узле конечного автомата AV_{http} определено четыре состояния: $s_2, s_3, s_4, s_7 \in S$ и четыре семантических оператора: $y_4, y_5, y_6, y_7 \in Y$, которые выполняются при переходе автомата из одного состояние в другое.

Если второй узел автомата AV_{http} переходит в состояние s_4, то это означает, что в HTTP-запросе, кроме идентификатора ресурса, также содержатся параметры доступа, обработка которых осуществляется узлом распознавания и анализа параметров доступа к ресурсу. Переход автомата в состояние s_5 означает, что вслед за идентификатором ресурса сразу следует версия протокола *HTTP*, значение которой обрабатывается четвёртым узлом автомата AV_{http}.

Третий *узел распознавания и анализа параметров доступа к ресурсам web*-сервера выполняет следующие функции:

- проверку строковой длины имён и значений параметров доступа к ресурсам *web*-сервера. Максимальная длина имени параметра и его значения не должна превышать значений, определённых в переменных $L_{VarLength}$ и $L_{ValLength}$ соответственно;
- контроль числа параметров доступа, присутствующих в *HTTP*-запросе. Количество параметров не должно превышать значения, указанного в переменной L_{Nquery}.

Графовая модель узла распознавания и анализа параметров доступа показана на рис. 5.

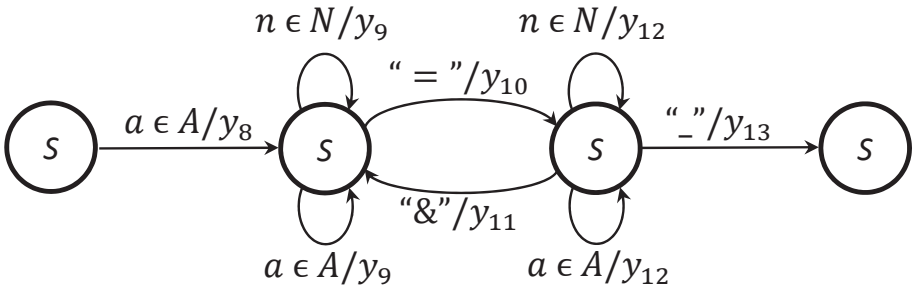

Рис. 5. Графовая модель узла распознавания и анализа параметров доступа

При условии, что третий узел автомата AV_{http} переходит в состояние s_7, автомат начинает обработку номера версии протокола, определённой в заголовке *HTTP*-запроса.

Четвёртый узел конечного автомата распознавания и анализа заголовков *HTTP*-протокола предназначен для проверки корректности номера версии, который указан в анализируемом *HTTP*-запросе. Номер версии считается корректным, если версия протокола, указанная в анализируемом *HTTP*-запросе, совпадает с одним из элементов строкового массива $S_{Versions}$. Графовая модель узла распознавания и анализа версии *HTTP*-протокола показана на рис. 6.

При условии, что четвёртый узел автомата AV_{http} переходит в состояние s_9, автомат начинает обработку оставшейся части *HTTP*-запроса.

Пятый узел конечного автомата AV_{http} (*узел распознавания и анализа заголовков HTTP-запроса*) предназначен для проверки длины имён и значений заголовков, содержащихся в *HTTP*-запросе. Блок проверяет, что количество заголовков *HTTP*-запроса не превышает значение переменной $L_{NHeaders}$, строковая длина имени каждого из параметров не превышает $L_{VarLength}$, а строковая длина значения параметра не превышает $L_{ValLength}$. Узел также обеспечивает проверку того, что в *HTTP*-запросе присутствуют

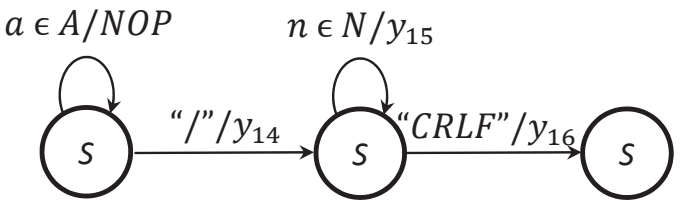

Рис. 6. Графовая модель узла распознавания и анализа версии HTTP-протокола

только те заголовки, которые определены в переменной $S_{Headers}$. Графовая модель узла распознавания и анализа параметров *HTTP*-запроса показана на рис. 7.

Если последний узел автомата переходит в состояние s_{12}, то это означает, что анализируемый *HTTP*-запрос не представляет опасности для *web*-сервера и не является частью сетевой атаки.

Обобщённая структура конечного автомата AV_{http}, который используется для анализа *HTTP*-запросов, показана на рис. 8.

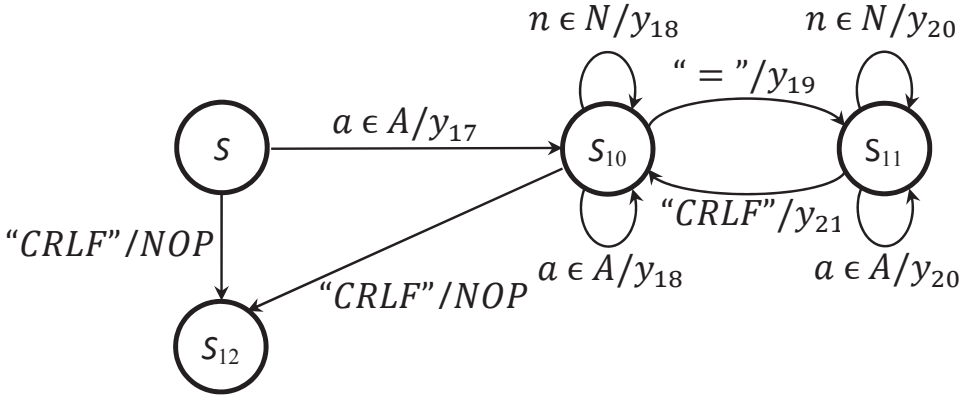

Рис. 7. Графовая модель узла распознавания и анализа параметров HTTP-запроса

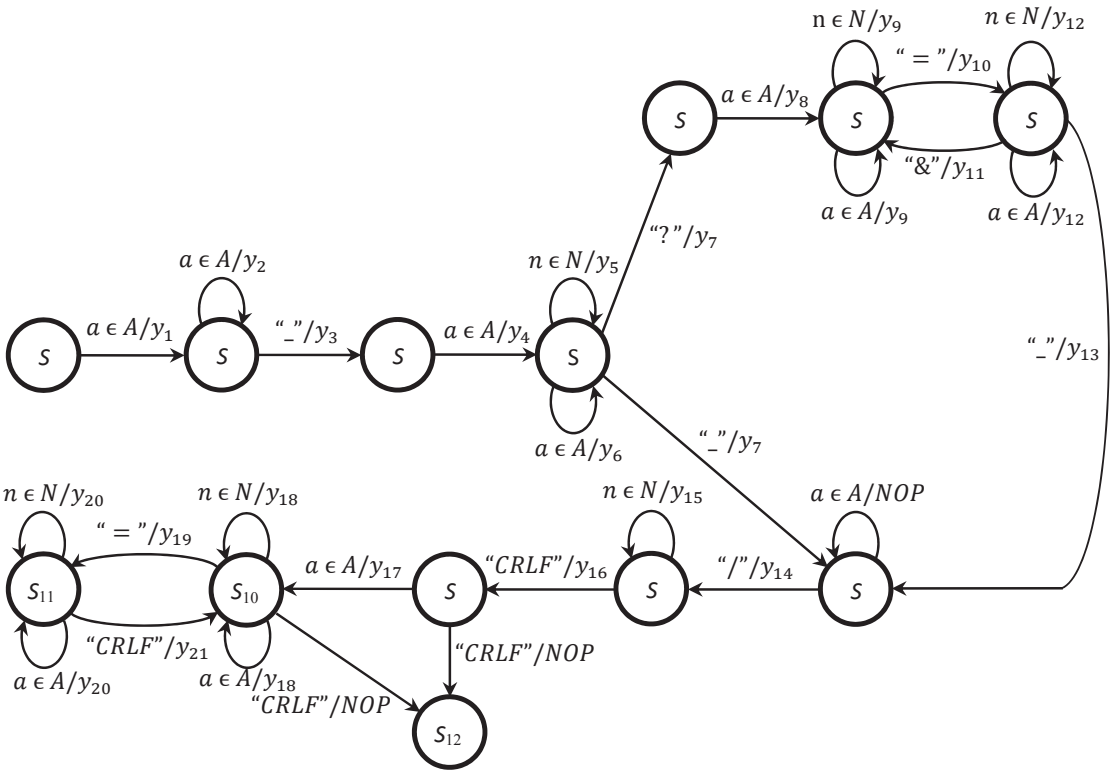

Рис. 8. Обобщенная структура конечного автомата AV_{http}, реализующего разработанную поведенческую модель процесса выявления атак на ресурсы web-серверов

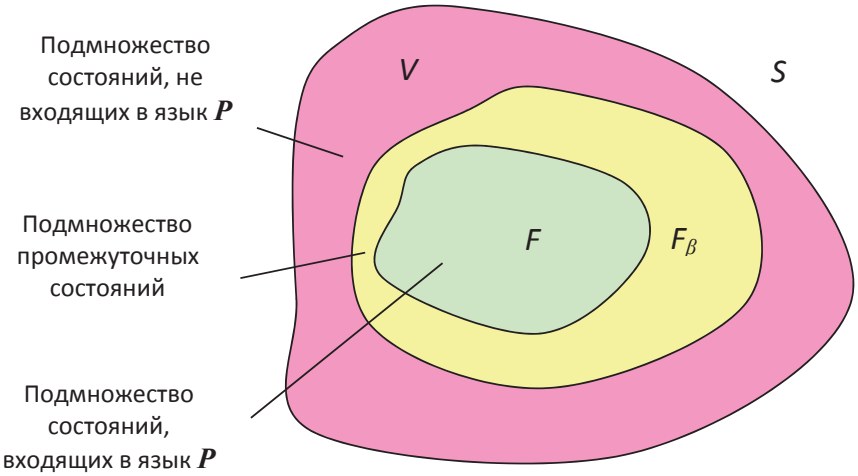

Рис. 9. Множество состояний конечного автомата

Рассмотренный подход к выявлению информационных атак на web-серверы путём анализа поступающих *HTTP*-запросов может быть использован для построения моделей защиты узлов других типов [12]. В этом случае должны подвергаться анализу запросы, сформированные на основе тех протоколов, по которым осуществляется взаимодействие с узлами автоматизированных систем передачи данных (АСПД). Так, например, для защиты почтовых серверов должны анализироваться запросы протоколов *SMTP (Simple Mail Transfer Protocol)* и POP*3 (Post Office Protocol Version 3)*, а для защиты файловых серверов — запросы протокола *FTP* (*File Transfer Protocol*).

Определение времени обнаружения информационной атаки

Многообразие протоколов, осуществляющих взаимодействие с элементами и узлами АСПД, в ряде случаев приводит к неоправданным задержкам обработки поступающих запросов[12] [5] и выдачи *незапланированного результата* (НЗР). Однозначно судить о причинах НЗР, который был сформирован web-сервером, без проведения дополнительных исследований не представляется возможным. Это могут быть как информационная атака, предполагающая «бесконечный цикл», так и ошибки программного обеспечения, алгоритмические сбои и др. Исходя из этого, целесообразно определить максимальное время обработки запроса, т. е. включить в его сигнатуру, наряду со временем инициирования, допустимое время его обработки, которое должно быть намного меньше выполнения деструктивных функций предполагаемой возможной информационной атаки заданного класса.

Рассмотрим процесс функционирования КА как дискретный марковский случайный процесс с непрерывным временем[13], состояния которого составляют множество $S=\{s_0,s_1,\ldots,s_n\}$. Из множества состояний S выделяется некоторое подмножество состояний $F=\{s_l,s_{l+1},\ldots,s_n\}$, $F\subset S$, которое не является замкнутым и, в свою очередь, не содержит в себе замкнутых подмножеств (рис. 9), а также подмножество F_β, являющееся промежуточным между множествами $V=(S-F)$ и F. В общем случае подмножество F_β может быть пустым, замкнутым и содержать замкнутые подмножества. «Физически» это означает, *во-первых* (когда подмножество пустое), мгновенный переход из подмножества F в подмножество V (нет промежуточных состояний), а *во-вторых* (в остальных случаях), переход в подмножество V запрещен, что на практике не всегда выполнимо.

Другими словами, если известно, что в какой-то момент $t=0$ КА находился в одном из состояний $s_i\in F$, то при $t\to\infty$ процесс хотя бы один раз перейдет из подмножества состояний F в подмножество состояний $V=(S-F)$ $(V\subset S)$, которое не является пустым: $\lim\limits_{t\to\infty} p_f(t)=0$, где $p_f(t)$ — вероятность непрерывного пребывания системы в момент времени t в состояниях подмножества F $[p_f(0)=1]$. Следовательно, в составе подмножества состояний F нет отдельных состояний или группы состояний без выхода. Состояния $(s_0,s_1,\ldots,s_{l-1})$ составляют подмножество $V=S-F$, а состояния $(s_l,\ldots,s_n)$ — множество $F=S-V$[14].

Известно, что в момент $t=0$ дискретный марковский случайный процесс находился в подмножестве состояний *F*,

$$\sum_{i=1}^{n}\tilde{p}_i(0)=1. \qquad (1)$$

[12] Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологии, протоколы : учебник. СПб. : ИД «Питер», 2010. 944 с.

[13] Вентцель Е.С., Овчаров Л.А. Теория случайных процессов и её инженерные приложения. 4-е изд. стер. М. : Высш. шк., 2007. 479 с.; Гайдук А.Р., Плаксиенко Е.А. Анализ и аналитический синтез цифровых систем управления : учеб. пособие. М. : Лань, 2018. 272 с.

[14] См.: Тараканов К.В., Овчаров Л.А., Тырышкин А.Н. Аналитические методы исследования систем. М. : Сов. радио, 1974. 240 с.

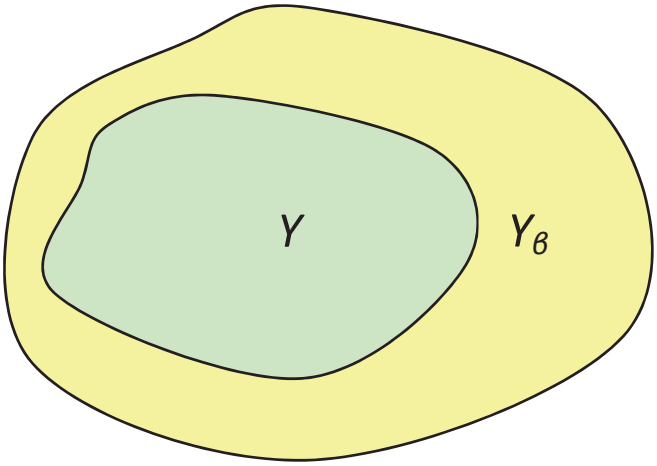

Рис. 10. Преобразованное состояние конечного автомата

Знак ~ над обозначениями вероятностей поставлен для того, чтобы отличить эти вероятности от обозначения вероятностей пребывания КА в любом состоянии s, принадлежащем множеству S.

Обозначим T случайную величину — время процесса обработки запроса по состояниям подмножества F до первого выхода из этого подмножества. Ввиду того, что само подмножество состояний F не является замкнутым и не содержит в себе замкнутых подмножеств, процесс при $t \to \infty$ может покинуть подмножество состояний F. В рассматриваемом случае случайное время T отсчитывается от момента $t = 0$.

Из множества состояний $V = S — F$ выделим подмножество состояний F_β $\{s_k,\ldots, s_{l-1}\}$ ($F_\beta \subset V$). Не исключается и случай, когда $F_\beta = V$ (в этом случае $k = 0$). К состояниям подмножества ($F_\beta \subset V$) отнесем лишь те, в которые возможен непосредственный переход из состояний подмножества F в состояния подмножества $V = S — F$, т.е. для любого состояния $s_j \in F_\beta$ найдется хотя бы одно такое состояние $s_i \in F$, что $L(s_i, s_j) = 1$, где L — маршрут из s_i в s_j.

Подмножество состояний F_β является «входными воротами» подмножества состояний V, через которые процесс функционирования переходит из подмножества состояний F в подмножество состояний V. Другими словами, к состояниям $(s_k,\ldots,s_{l-1})$, которые составляют подмножество F_β, отнесем лишь те состояния подмножества $V = S — F$, для которых выполняется условие:

$$\lambda_{ij}(t) \neq 0,\ (i=l,\ l+1,\ldots,\ n;\ j=0,\ 1,\ \ldots,\ l — 1;\ t > 0). \quad (2)$$

Назовем подмножество F_β β-окрестностью подмножества F. Тогда время T «блуждания» процесса по состояниям подмножества F до первого выхода из него равно времени, которое отсчитывается от начала «блуждания» (t=0) до первого попадания процесса в β-окрестность подмножества F, т. е. в одно из состояний подмножества F_β.

Очевидно, что закон распределения случайной величины T не изменится, если все состояния множества F_β сделать поглощающими (концевыми). Следовательно, при нахождении закона распределения случайной величины T можно ограничиться рассмотрением процесса блуждания системы по преобразованным состояниям множества $F + F_\beta$. Преобразование состоит в том, что подмножество F_β образуется только из поглощающих состояний.

Чтобы состояния подмножества $F_\beta = \{s_k,\ldots,s_{l-1}\}$ были поглощающими, достаточно положить

$$\lambda_{ji}(t) \equiv 0 \quad (3)$$

для всех $j = k$, ..., l — 1. В этом случае выход из подмножества состояний F_β будет невозможен.

Таким образом, для определения закона распределения случайной величины Т достаточно рассмотреть процесс «блуждания» по преобразованным состояниям подмножества $F + F_\beta$, при этом все состояния подмножества F_β состоят только из поглощающих состояний (рис. 10). Это утверждение равносильно тому, что рассматривается преобразованный подграф состояний подмножества $F + F_\beta$. При этом все ребра, выходящие из вершин графа, составляющих подмножество состояний F_β, исключаются (преобразуются). Преобразование сводится к тому, что вершины $(s_k,\ldots,s_{l-1})$ не имеют выходящих ребер.

Функция распределения случайной величины T (по определению) равна вероятности того, что к моменту времени t (t>0) процесс уже покинет состояния, образующие множество F, и, следовательно, попадёт в состояния, образующие подмножество F_β, так как в преобразованном подмножестве состояний $F + F_\beta$ он (процесс) никуда (кроме подмножества состояний F_β) в конце концов попасть не может. Ввиду того, что подмножество состояний F_β состоит только из поглощающих состояний, процесс, попав однажды в одно из состояний F_β, так там и останется.

Следовательно, функция распределения времени T равна вероятности того, что к моменту времени t процесс «блуждания» окажется в одном из состояний, принадлежащих подмножеству F_β для преобразованного подмножества состояний $F + F_\beta$:

$$F(t) = \sum_{s \in F_\beta} \tilde{p}_s(t),$$

$$F(t) = \sum_{j=k}^{l-1} \tilde{p}_j(t), \tag{4}$$

где индекс суммирования распространяется на все состояния подмножества F_β (β-окрестность подмножества F).

Найдем плотность распределения случайной величины T:

$$f(t) = \frac{d}{dt} F(t),$$

$$f(t) = \sum_{j=k}^{l-1} \dot{\tilde{p}}_j(t). \tag{5}$$

С другой стороны, на основании общего правила составления системы дифференциальных уравнений для вероятностей состояний с учетом (3) имеем

$$\dot{\tilde{p}}_j(t) = \sum_{i=l}^{n} \lambda_{ij}(t) \tilde{p}_i(t), \ (j = k, k+1, \ldots, l - 1) \tag{6}$$

Таким образом, плотность распределения случайной величины T можно найти из выражения

$$f(t) = \sum_{j=k}^{l-1} \sum_{i=l}^{n} \lambda_{ij}(t) \tilde{p}_i(t). \tag{7}$$

Для отыскания вероятностей $\tilde{p}_i(t)$ достаточно проинтегрировать систему уравнений для вероятностей пребывания процесса в состоянии подмножества F с учетом того, что переход из подмножества состояний F возможен только в подмножество состояний F_β, а последнее подмножество состоит из одних поглощающих состояний

$$\dot{\tilde{p}}_i(t) = -\sum_{h=l}^{n} \lambda_{ih}(t) \tilde{p}_i(t) - \sum_{h=k}^{l-1} \lambda_{ih}(t) \tilde{p}_i + \\ + \sum_{h=l}^{n} \lambda_{hi}(t) \tilde{p}_h(t), \tag{8}$$

где $i = l, l + 1, \ldots, n$.

Начальными условиями для этой системы дифференциальных уравнений являются условия (1).

Итак, для определения закона распределения времени «блуждания» процесса в незамкнутом подмножестве состояний F до первого выхода за пределы этого подмножества достаточно проинтегрировать систему дифференциальных уравнений для вероятностей состояний (9) при начальных условиях (1).

Очевидно, что таким образом можно решать задачу для любого дискретного марковского случайного процесса с непрерывным временем, для которого подмножество состояний F не является замкнутым и не содержит замкнутых подмножеств.

Иногда существует необходимость рассмотрения и более жесткого условия, состоящего в том, что математическое ожидание времени T пребывания в подмножестве состояний F должно быть конечным. В этом случае должно выполняться следующее неравенство[15]:

$$\lim_{t \to \infty} \int_0^t [1 - \sum_{j=k}^{l-1} \tilde{p}_j(\tau)] d\tau = \\ = \lim_{t \to \infty} \int_0^t [1 - \sum_{j=k}^{l-1} \sum_{i=l}^{\infty} \lambda_{ij}(\xi) p_i(\xi) d\xi] d\tau, \\ \lim_{t \to \infty} \int_0^t [1 - \sum_{j=k}^{l-1} \sum_{i=l}^{\infty} \lambda_{ij}(\xi) p_i(\xi) d\xi] d\tau < \infty. \tag{9}$$

При этом также возможен случай, когда подмножество состояний F_β является бесконечным. Даже когда незамкнутое подмножество состояний F является конечным и для любого t выполняется неравенство (2), равенство (9) может не выполняться. Поэтому полученный закон распределения времени в заданном подмножестве состояний конечного автомата необходимо проверять, чтобы выяснить, выполняется ли условие (9). Вместе с тем он позволяет получить приближенную оценку, удовлетворяющую решению общей задачи исследования.

Заключение

Современные вычислительные средства и построенные на их основе компьютерные сети различного назначения должны обладать надежной, своевременной, актуальной и адекватной информационной защитой [7, 11]. Практика показывает, что ущерб от нарушений информационной безопасности может привести к крупным финансовым потерям и даже краху компании, он растет из года в год, и отнюдь не линейно [4]. Исходя из этого, вопросы обнаружения информационных атак на основе формирования сетевых запросов требуют научно обоснованного решения. Математической основой разработанной модели аппаратно-программного комплекса является теория конечных автоматов. При этом обязательное условие — учет времени обработки поступающего запроса и времени его инициализации.

Перспективным направлением является создание интеллектуальных аппаратно-программных комплексов, в основу которых положен принцип структурной организации высокоорганизованных материй (например, аппарат теории нейронных сетей) [12]. Однако в этом случае возникает другая проблема — увеличение времени обработки сетевого запроса (при существующих в настоящее время вычислительных средствах), оказывающее непосредственное влияние на комплексные показатели

[15] Вентцель Е.С., Овчаров Л.А. Теория случайных процессов и её инженерные приложения. 4-е изд. стер. М. : Высш. шк., 2007. 479 с.

надежности (коэффициенты готовности, технического использования, оперативной готовности) АСПД в целом. Разумное соотношение этих параметров — сложная многокритериальная задача поиска рационального решения, обеспечивающих баланс требуемой защищённости и высоких комплексных показателей надежности.

Рецензент: ***Омельченко Виктор Валентинович,*** *доктор технических наук, профессор, заслуженный деятель науки и техники РСФСР, советник секретариата научно-технического совета ВПК «НПО Машиностроения», г. Москва, Российская Федерация.*
E-mail: omvv@yandex.ru

Литература

1. Анин Б.Ю. Защита компьютерной информации. СПб. : БХВ-Санкт-Петербург, 2016. 384 с.
2. Гончаров В.В., Гончаров А.В. Методика обоснования рационального варианта проверки телекоммуникационных систем и сетей // Правовая информатика. 2022. № 4. С. 39—48. DOI: 10.21681/1994-1404-2022-4-39-48 .
3. Додонов А.Г., Ландэ Д.В. Живучесть информационных систем. К. : Наукова думка, 2011. 256 с.
4. Кульба В.В., Ковалевский С.С., Шелков А.Б. Достоверность и сохранность информации в АСУ. М. : Синтег, 2004. 496 с.
5. Куроуз Дж., Росс К. Компьютерные сети. Нисходящий подход. М. : Эксмо, 2016. 912 с.
6. Ловцов Д.А. Теория защищенности информации в эргасистемах : монография. М. : РГУП, 2021. 276 с. ISBN 978-5-93916-896-0.
7. Ловцов Д.А. Обеспечение информационной безопасности в российских телематических сетях // Информационное право. 2012. № 4. С. 3—7.
8. Ловцов Д.А. Информационная безопасность и нетрадиционные угрозы // Федеральный справочник. Т. 8. Оборонно-промышленный комплекс России. М. : Центр стратег. исследований, 2013. С. 507—512.
9. Низамутдинов М.Ф. Тактика защиты и нападения на web-приложения. СПб. : БХВ-Петербург, 2005. 480 с.
10. Оре О. Графы и их применение. М. : Ленанд, 2015. 208 с.
11. Системный анализ и аналитические исследования: руководство для профессиональных аналитиков / А.И. Ракитов, Д.А. Бондяев, И.Б. Романов, С.В. Егерев, А.Ю. Щербаков. Под. ред. А.И. Ракитова. М. : РГГУ, 2009. 448 с.
12. Сети следующего поколения NGN / Под ред. А.В. Рослякова. М. : Эко-Трендз, 2008. 420 с.
13. Таненбаум Э.С., Уэзеролл Д. Компьютерные сети. СПб. : ИД «Питер», 2022. 900 с.
14. Фленов М.Е. Web-сервер глазами хакера. СПб. : БХВ-Санкт-Петербург, 2021. 257 с.
15. Харари Ф. Теория графов. М. : Ленанд, 2018. 304 с.

INFORMATION ATTACK DETECTION MODELLING BASED ON THE FINITE AUTOMATA THEORY

Vladimir Goncharov, *Dr.Sc. (Technology), Professor, Honoured Figure of Higher School of the Russian Federation, Head of the Department of Mathematics of the Peter the Great Military Academy, Moscow, Russian Federation.*
E-mail: v_v_goncharov@mail.ru

Aleksandr Goncharov, *external Ph.D. student at the Peter the Great Military Academy, Moscow, Russian Federation.*
E-mail: dinozavp@inbox.ru

Ol'ga Mishenina, *Ph.D. (Paedagogy), Associate Professor, Professor at the Department of Mathematics of the Peter the Great Military Academy, Moscow, Russian Federation.*
E-mail: o.v.mishenina@gmail.com

Keywords: *corporate network, network technologies, network request, information attack, information security, finite automaton, element, set, elements subset, graph model, random variable, random variable's distribution and density function.*

Abstract

Purpose of the paper: improving the research and methodological foundations for raising the quality of basic network interaction protocols security control in corporate network nodes.

Methods of study: system analysis and the mathematical apparatus for generation and recognition of 'regularly built' chains forming regular sets defined by expressions making it possible to build a finite automaton allowing to precisely repeat the chain of the corresponding regular set.

Study findings: a model was developed describing the basic computer network nodes interaction protocol using an automaton language P each element of which corresponds to a basic network request and can be correctly processed by the hard- and software system of the network. The language P is described using finite recognising automata detecting potentially dangerous requests whose elements do not comply with the standards described based on this language. A conclusion is justified that including the request initialisation and maximum permitted processing time in the request signature will make it possible to considerably reduce the possibilities for carrying out computer attacks of this type.

The results obtained are the base for developing appropriate efficient information and mathematical support for the hard- and software system for complex computer networks security.

References

1. Anin B.Iu. Zashchita komp'iuternoi informatsii. SPb. : BKhV-Sankt-Peterburg, 2016. 384 pp.
2. Goncharov V.V., Goncharov A.V. Metodika obosnovaniia ratsional'nogo varianta proverki telekommunikatsionnykh sistem i setei. Pravovaia informatika, 2022, No. 4, pp. 39—48. DOI: 10.21681/1994-1404-2022-4-39-48 .
3. Dodonov A.G., Lande D.V. Zhivuchest' informatsionnykh sistem. K. : Naukova dumka, 2011. 256 pp.
4. Kul'ba V.V., Kovalevskii S.S., Shelkov A.B. Dostovernost' i sokhrannost' informatsii v ASU. M. : Sinteg, 2004. 496 pp.
5. Kurouz Dzh., Ross K. Komp'iuternye seti. Niskhodiashchii podkhod. M. : Eksmo, 2016. 912 pp.
6. Lovtsov D.A. Teoriia zashchishchennosti informatsii v ergasistemakh : monografiia. M. : RGUP, 2021. 276 pp. ISBN 978-5-93916-896-0.
7. Lovtsov D.A. Obespechenie informatsionnoi bezopasnosti v rossiiskikh telematicheskikh setiakh. Informatsionnoe pravo, 2012, No. 4, pp. 3—7.
8. Lovtsov D.A. Informatsionnaia bezopasnost' i netraditsionnye ugrozy. Federal'nyi spravochnik, t. 8. Oboronno-promyshlennyi kompleks Rossii. M. : Tsentr strateg. issledovanii, 2013, pp. 507—512.
9. Nizamutdinov M.F. Taktika zashchity i napadeniia na web-prilozheniia. SPb. : BKhV-Peterburg, 2005. 480 pp.
10. Ore O. Grafy i ikh primenenie. M. : Lenand, 2015. 208 pp.
11. Sistemnyi analiz i analiticheskie issledovaniia: rukovodstvo dlia professional'nykh analitikov. A.I. Rakitov, D.A. Bondiaev, I.B. Romanov, S.V. Egerev, A.Iu. Shcherbakov. Pod. red. A.I. Rakitova. M. : RGGU, 2009. 448 pp.
12. Seti sleduiushchego pokoleniia NGN. Pod red. A.V. Rosliakova. M. : Eko-Trendz, 2008. 420 pp.
13. Tanenbaum E.S., Uezeroll D. Komp'iuternye seti. SPb. : ID "Piter", 2022. 900 pp.
14. Flenov M.E. Web-server glazami khakera. SPb. : BKhV-Sankt-Peterburg, 2021. 257 pp.
15. Kharari F. Teoriia grafov. M. : Lenand, 2018. 304 pp.