Кирилл В. Плаксий[1], Лидия Л. Кулагина[2], Андрей А. Никифоров[3],
Наталья Г. Милославская[4]
Национальный исследовательский ядерный университет «МИФИ»,
Каширское ш., 31, Москва, 115409, Россия
[1]e-mail: KVPlaksii@mephi.ru, http://orcid.org/0000-0002-8949-6772
[2]e-mail: lidusy_0104@mail.ru, https://orcid.org/0000-0003-1261-1119
[3]e-mail: andreinikiforov993@gmail.com, http://orcid.org/0000-0002-2726-0000
[4]e-mail: NGMiloslavskaya@mephi.ru, http://orcid.org/0000-0002-1231-1805

ИССЛЕДОВАНИЕ ВОПРОСОВ ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ ГРАФОВЫХ СУБД, ПРИГОДНЫХ ДЛЯ РАБОТЫ С БОЛЬШИМИ ДАННЫМИ, ПРИ ОБНАРУЖЕНИИ ДЕЛ ПО ОТМЫВАНИЮ ДОХОДОВ, ПОЛУЧЕННЫХ ПРЕСТУПНЫМ ПУТЕМ, И ФИНАНСИРОВАНИЮ ТЕРРОРИЗМА*

DOI: http://dx.doi.org/10.26583/bit.2020.4.05

Аннотация. Исследуются вопросы обеспечения информационной безопасности (ИБ) в популярных в настоящее время графовых системах управления базами данных (СУБД), способных работать с большими данными и хранить информацию, созданную в ходе генерации преступных дел по отмыванию доходов, полученных преступным путем, и финансированию терроризма (ОД/ФТ). Продолжая предыдущее исследования авторов, в данной работе ставится цель анализа угроз ИБ и уязвимостей графовых СУБД. Эти СУБД отличаются от реляционных видом хранимых данных и принципом их хранения, поэтому актуальной является проблема составления перечня угроз ИБ в связи с отсутствием такового в мировом масштабе. На основе анализа угроз ИБ для обычных СУБД и с учётом особенностей графовых СУБД, их структуры и уязвимостей конкретных графовых СУБД предлагаются собственный перечень угроз ИБ и методы по защите от них, а также некоторые рекомендации по устранению уязвимостей, используемых угрозами ИБ.

Ключевые слова: отмывание доходов, полученных преступным путем, финансирование терроризма, ОД/ФТ, информационная безопасность, большие данные, системы управления базами данных (СУБД), угрозы ИБ, уязвимости СУБД.

Для цитирования: ПЛАКСИЙ, Кирилл В. и др. ИССЛЕДОВАНИЕ ВОПРОСОВ ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ ГРАФОВЫХ СУБД, ПРИГОДНЫХ ДЛЯ РАБОТЫ С БОЛЬШИМИ ДАННЫМИ, ПРИ ОБНАРУЖЕНИИ ДЕЛ ПО ОТМЫВАНИЮ ДОХОДОВ, ПОЛУЧЕННЫХ ПРЕСТУПНЫМ ПУТЕМ, И ФИНАНСИРОВАНИЮ ТЕРРОРИЗМА. Безопасность информационных технологий, [S.l.], v. 27, p. 53–64, n. 4, 2020. ISSN 2074-7136. Доступно на: <https://bit.mephi.ru/index.php/bit/article/view/1306>. Дата доступа: 18 nov. 2020. DOI: http://dx.doi.org/10.26583/bit.2020.4.05.

***Благодарности.** Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 18-07-00088.*

Kirill V. Plaksiy[1], Lidiia L. Kulagina[2], Andrey A. Nikiforov[3], Natalia G. Miloslavskaya[4]
National Research Nuclear University MEPhI (Moscow Engineering Physics Institute),
Kashirskoe shosse, 31 Moscow, 115409, Russia
[1]e-mail: kirillplaksiy@mail.ru, http://orcid.org/0000-0002-8949-6772
[2]e-mail: lidusy_0104@mail.ru, https://orcid.org/0000-0003-1261-1119
[3]e-mail: andreinikiforov993@gmail.com, http://orcid.org/0000-0002-2726-0000
[4]e-mail: NGMiloslavskaya@mephi.ru, http://orcid.org/0000-0002-1231-1805

Investigation of information security issues for graph databases suitable for big data processing while detecting money laundering and terrorism financing cases*

DOI: http://dx.doi.org/10.26583/bit.2020.4.05

Abstract. The information security (IS) issues in the currently popular graph database management systems (DBMS), suitable for big data processing and storing information created during the generation of criminal cases on money laundering and financing of terrorism (ML/FT), are examined. This paper continues the previous authors' research and aims to analyze IS threats and vulnerabilities of graph DBMS. These DBMS differ from the relational ones in the type of stored data and the principle of their storage; therefore the compiling of a list of IS threats is urgent due to its absence on a global scale. The original IS threat list and methods for protecting against them, as well as some recommendations for eliminating vulnerabilities used by IS threats are proposed. The obtained results are based on the analysis of IS threats for conventional DBMS and taking into account the peculiarities of graph DBMS, their structure as well as vulnerabilities of specific graph DBMS.

Keywords: money laundering, terrorism financing, ML/FT, information security, typology, Big Data, Database Management System (DBMS), information security threats, vulnerabilities.

For citation: PLAKSIY, Kirill V. et al. Investigation of information security issues for graph databases suitable for big data processing while detecting money laundering and terrorism financing cases. IT Security (Russia), [S.l.], v. 27, n. 4, p. 53–64, 2020. ISSN 2074-7136. Available at: <https://bit.mephi.ru/index.php/bit/article/view/1306>. Date accessed: 18 nov. 2020. DOI: http://dx.doi.org/10.26583/bit.2020.4.05.

***Acknowledgement.** The research was carried out under the financial support of the RFBR in the framework of scientific project No. 18-07-00088.*

Введение

С каждым годом количество утечек информации неуклонно растет. Поскольку основным хранилищем данных выступает система управления базами данных (СУБД), то атаки на СУБД являются одними из самых опасных для организаций. Согласно отчету об утечках данных компании Verizon [1], СУБД – один из наиболее уязвимых активов различных организаций. Графовые СУБД – не исключение. Это приводит к необходимости обеспечения информационной безопасности (ИБ) всей инфраструктуры, связанной с использованием графовых СУБД. Обеспечение ИБ СУБД должно начинаться с устранения существующих уязвимостей. Сократить их количество возможно лишь при комплексном подходе к защите данных в графовых СУБД. Создание эффективной системы обеспечения ИБ СУБД требует оценки актуальных угроз ИБ с учетом ценность защищаемой информации и методов несанкционированного воздействия на нее. Среди основных угроз ИБ обычно выделяют угрозы несанкционированного использования информации в СУБД системными администраторами, пользователями, злоумышленниками, вирусных атак с различными последствиями, SQL-инъекций, технических проблем, снижения производительности, отказа в доступе, исключающего возможность использования информации, физического ущерба, нанесенного оборудованию или каналам связи, наличия ошибок и недоработок, несанкционированных возможностей программного обеспечения (ПО), управляющего СУБД и операционных систем (ОС) и т.п.

Во многом нереляционные СУБД более безопасны из-за отсутствия SQL-запросов и невозможности провести SQL-инъекцию [2]. Но отсутствие SQL-кода в запросе еще не означает, что система полностью защищена. Так, СУБД NoSQL состоит из приложения, интерфейса прикладного программирования (API) NoSQL и NoSQL-СУБД, каждый из которых имеет свои уязвимости. Например, сама СУБД, как и любое другое приложение, подвержена атакам переполнения буфера и имеет уязвимости в системе аутентификации. Для злоумышленника этот уровень сложно атаковать, так как его уязвимостями отслеживают разработчики и сообщество пользователей. На уровне API в большинство нереляционных СУБД находятся библиотеки, используемые для организации доступа к данным. Они часто имеют открытый исходный код, что помогает обнаружить их уязвимости злоумышленникам. Чаще всего атакуется верхний уровень, содержащий

уязвимости в проверке входных данных. При этом используется тот же подход, что и при SQL-инъекциях, но основанный на других языках запросов, характерных для нереляционных СУБД.

Данное исследование продолжает работу, начатую авторами в [3, 4], и ставит своей целью проанализировать типичные уязвимости и угрозы ИБ для графовых СУБД и выработать некоторые рекомендации по устранению уязвимостей, используемых этими угрозами ИБ.

1. Угрозы информационной безопасности графовых СУБД

В ходе исследования были рассмотрены различные графовые хранилища данных, которые могут быть применены при обнаружении преступных дел по отмыванию доходов, полученных преступным путем, и финансированию терроризма (ОД/ФТ). Для работы с данными в качестве средства визуализации хорошо подходят графы [5], что упрощает восприятие человеком информации и уменьшает её объемы за счет различных подходов, учитывающих специфику решаемой задачи.

В графовых СУБД информация хранится в виде узлов или объектов, а отношения между ними позволяют получать дополнительную информацию, имеющую большую ценность. Несмотря на то, что эта технология является относительно новой, наиболее значимые результаты получаются с 2013 г. и уже существуют СУБД, которые работают в реальных системах. Для хранения новых данных и работы с ними используются такие платформы как Neo4j [6], AllegroGraph [7], JanusGraph [8]. Для исследования были выбраны Neo4j, JanusGraph и Dgraph из-за их популярности (рис. 1) [9], а также открытого кода, языков реализации, мультиплатформенности и разнообразия средств обеспечения ИБ (табл. 1).

Rank			DBMS	Database Model	Score		
Jun 2020	May 2020	Jun 2019			Jun 2020	May 2020	Jun 2019
1.	1.	1.	Neo4j	Graph	48.27	-1.49	-1.28
2.	2.	2.	Microsoft Azure Cosmos DB	Multi-model	30.80	+0.13	+2.56
3.	3.	↑ 4.	ArangoDB	Multi-model	5.38	+0.70	+0.81
4.	4.	↓ 3.	OrientDB	Multi-model	4.82	+0.68	-0.77
5.	5.	5.	Virtuoso	Multi-model	2.28	-0.07	-0.83
6.	6.	↑ 7.	Amazon Neptune	Multi-model	2.17	+0.41	+0.93
7.	7.	↓ 6.	JanusGraph	Graph	2.01	+0.36	+0.46
8.	↑ 9.	↑ 11.	Dgraph	Graph	1.40	+0.31	+0.51
9.	↓ 8.	↓ 8.	GraphDB	Multi-model	1.25	+0.06	+0.16
10.	↑ 12.	↑ 18.	FaunaDB	Multi-model	1.19	+0.25	+0.84
11.	↓ 10.	↑ 13.	Stardog	Multi-model	1.15	+0.08	+0.44
12.	↓ 11.	↓ 9.	Giraph	Graph	0.97	+0.02	-0.11
13.	13.	↓ 12.	TigerGraph	Graph	0.90	+0.05	+0.18

Рис. 1. Популярные СУБД
(Fig. 1. Popular databases)

В ходе исследования было установлено, что специального перечня уязвимостей и угроз ИБ для графовых СУБД не создано при том, что их различия с реляционными СУБД существенны [3, 4]. Поэтому было решено провести анализ существующих перечней угроз ИБ для реляционных СУБД и, учитывая специфику графовых СУБД, сформулировать актуальные для последних.

Таблица 1. Графовые СУБД

	Neo4j	**JanusGraph**	**Dgraph**
Языки реализации	Java, Scala	Java	Go
Серверные ОС	Linux, OS X, Solaris, Windows Может использоваться и без сервера в качестве встроенной базы данных Java	Linux, OS X, Unix, Windows	Linux, OS X, Windows
Контроль доступа	Присутствует	Осуществляется через Rexter Graph Server	Нет (запланировано на будущие версии)
Аутенти-фикация	Подключаемая аутентификация с поддерживаемыми стандартами (LDAP, Active Directory, Kerberos)	Базовая и токен-аутентификация	Нет (запланировано на будущие версии)
Шифрование	Целенаправленного внутреннего шифрования нет, поддержка стороннего шифрования	Целенаправленного внутреннего шифрования нет, поддержка стороннего шифрования	Шифрование данных в состоянии покоя HDFS
Целостность данных	Использование ограничений	Контроль целостности данных при загрузке	Использование двойных баз
Резервные копии	Да, как для одного компьютера, так и для кластеров	Собственные копии, а также поддержка сторонних средств	Копии работающих кластеров

Сначала были рассмотрены типовые угрозы безопасности персональных данных (ПДн), которые обычно хранятся в различных базах данных (БД) и обрабатываются в информационных системах ПДн (ИСПДн) [10]. В целях формирования систематизированного перечня угроз в базовой модели ФСТЭК России они классифицируются в соответствии со следующими признаками:

- вид защищаемой информации, содержащей ПДн;
- вид возможных источников угроз;
- тип ИСПДн, на которые направлена реализация угроз;
- способ реализации угроз;
- вид нарушаемого свойства информации (вид несанкционированных действий, осуществляемых с ПДн);
- используемая уязвимость;
- объект воздействия.

Далее на основе базы угроз ФСТЭК России были определены наиболее типичные угрозы для БД и сформулированы угрозы ИБ для графовых СУБД, включая следующие:

- угроза использования механизмов авторизации для повышения привилегий (УБИ.031);
- угроза повышения привилегий (УБИ.122);
- угроза несанкционированного создания учётной записи пользователя (УБИ.090);
- угроза межсайтового скриптинга (УБИ.041);
- угроза межсайтовой подделки запроса (УБИ.042);
- угроза искажения XML-схемы (УБИ.026);
- угроза неконтролируемого копирования данных внутри хранилища больших данных (УБИ.057);
- угроза несанкционированного удаления защищаемой информации (УБИ.091);
- угроза несанкционированной модификации защищаемой информации (УБИ.179);
- угроза несанкционированного удаления защищаемой информации (УБИ.091);

- угроза неконтролируемого уничтожения информации хранилищем больших данных (УБИ.060);
- угроза, сбоя автоматического управления системой разграничения доступа хранилища больших данных (УБИ.148);
- угроза несогласованности правил доступа к большим данным (УБИ.097);
- угроза обхода некорректно настроенных механизмов аутентификации (УБИ.100);
- угроза приведения системы в состояние «отказ в обслуживании» (УБИ.140)

2. Уязвимости графовых СУБД

В ходе работы были выделены типичные уязвимости, характерные для СУБД и для графовых СУБД в частности:

1. Уязвимость в системе аутентификации. Многие графовые СУБД по умолчанию устанавливаются без пароля. Разработчики предполагают, что установка СУБД происходит в доверенном окружении.

2. Уязвимость в системе авторизации. Изначально в некоторых графовых СУБД любой новый пользователь имеет по умолчанию доступ к чтению всей БД. Также существует уязвимость в системе авторизации администратора – пользователь Admin имеет доступ к БД и права на чтение и запись. Если по умолчанию отсутствует пароль, то предоставляется полный доступ.

3. Уязвимость драйверов БД, приводящая к переполнению буфера и позволяющая нарушителю вызвать атаку «отказ в обслуживании».

4. Незашифрованный текст. Данные передаются в БД в открытом виде и могут быть перехвачены при любой атаке типа «человек по середине».

5. Уязвимость, опускающая инъекции в регулярных выражениях. Многие нереляционные СУБД позволяют осуществлять поиск с помощью регулярных выражений. Их неправильное использование может нанести вред. Аутентификация пользователя может осуществляться посредством запроса, указывающего в качестве пароля регулярное выражение. Переменная password не фильтруется, что позволяет злоумышленнику получить несанкционированный доступ (НСД) к БД.

6. Уязвимость, допускающая инъекции кода. Графовые СУБД не поддерживают SQL, но ни одна СУБД не может обойтись без использования языка запросов. Для каждой графовой СУБД существует свой язык запросов, с помощью которого можно осуществить разного рода атаки-инъекции, если отсутствует входная фильтрация данных.

7. Возможность манипуляции REST-интерфейсом. В ходе развития сервис-ориентированной архитектуры большую популярность получили REST-решения (REpresentational State Transfer, рис. 2). В некоторые графовые СУБД входит простой REST-интерфейс, позволяющий получать доступ к БД в режиме чтения.

Для каждой СУБД уязвимости уникальны [11-13]. Входящие в базу CVE (Common Vulnerabilities and Exposures) уязвимости графовых СУБД были сведены в табл. 2.

3. Распространённые методы защиты данных, применимые к графовым СУБД

При росте потребностей в операциях с информацией возрастала необходимость в средствах обеспечения ИБ данных в СУБД. Средства защиты в различных СУБД несколько отличаются, однако общим является многоуровневость защиты – чем больше барьеров-уровней, тем сложнее их преодолеть злоумышленнику. Подход к обеспечению защиты данных в графовых СУБД, как и в других системах, складывается из обеспечения конфиденциальности, целостности и доступности [14]. На нижних уровнях находятся стандартные способы защиты: пароли, шифрование данных, разграничение прав доступа к объектам БД, контрольные след выполняемых операций, резервное копирование.

Таблица 2. Уязвимости графовых СУБД

№	Neo4j	JanusGraph	Dgraph
1	CVE-2018-18389. Настройка LDAP-аутентификации с помощью STARTTLS и системной учетной записи авторизации позволяет злоумышленнику войти на сервер, отправив любое допустимое имя пользователя с произвольным паролем из-за неправильного контроля доступа Neo4j Enterprise Database Server 3.4.x до 3.4.9.	CVE-2018-1000632. Версия dom4j до версии 2.1.1 содержит уязвимость, которая может привести к изменению злоумышленником XML-документов с помощью XML-инъекции в класс «элементы» методами: addElement, addAttribute. Атака реализуется при указании атрибутов или элементов в XML-документе.	CVE-2019-5736. Уязвимость модуля Runc Go до версии 1.0-rcb, использованного в Docker до версии 18.09.2 и других продуктах, позволяет злоумышленнику перезаписать двоичный файл runc хоста (и получить root-доступ к хосту), используя возможность выполнять команду как root. Причина: неправильная обработка файлового дескриптора, связанного с /proc/self/exe.
2	CVE-2018-1000820. СУБД Neo4j-contrib neo4j-apoc-procedures версии содержит внешние XML-сущности (XXE) с уязвимостью XML-парсера, что может привести к раскрытию конфиденциальной информации, отказу в обслуживании, сканированию портов.	CVE-2015-5211. В некоторых ситуациях Spring Framework 4.2.0–4.2.1, 4.0.0–4.1.7, 3.2.0–3.2.14 и более ранние версии уязвимы для атаки Reflected File Download (RFD). Злоумышленник создает URL-адрес с расширением пакетного сценария, в результате чего ответ загружается (а не отображается) и включает некоторые входные данные, отраженные в ответе.	CVE-2020-10661. Инструменты безопасного хранения конфиденциальной информации в динамических облачных средах HashiCorp Vault и Vault Enterprise версий 0.11.0–1.3.3 Go модуля, чей API использует Dgraph, могут при определенных обстоятельствах иметь существующие политики вложенного пути к файлу, предоставляющие доступ к пространствам имен, созданным позднее.
3	CVE-2013-7259. Множественные уязвимости подделки межсайтовых запросов (CSRF) в Neo4J 1.9.2 позволяют удаленным злоумышленникам перехватывать аутентификацию администраторов для запросов, выполняющих произвольный код.	CVE-2013-1801. Httparty gem 0.9.0 и более ранние версии для Ruby не ограничивают должным образом приведение строковых значений, что может позволить удаленным злоумышленникам внедрять объекты и выполнять произвольный код или вызывать «отказ в обслуживании» с помощью действия «Поддержка пакетов для преобразования типа YAML».	CVE-2020-10660. Инструменты безопасного хранения конфиденциальной информации в динамических облачных средах HashiCorp Vault и Vault Enterprise версий 0.9.0–1.3.3 Go модуля, чей API использует Dgraph, могут при определенных обстоятельствах непреднамеренно включать в себя группы, к которым, в сущности, не имеют прав доступа.
4	CVE-2013-7220. js/ui/screenShield.js в GNOME Shell (gnome-shell) до версии 3.8. Допускалось, что злоумышленники, находящиеся недалеко физически, могли выполнять случайные команды путем использования рабочей станции, оставленной без присмотра, и поиском активности на клавиатуре.	CVE-2013-0156. active_support/ core_ext/hash/conversions.rb в Ruby до версий до 2.3.15, 3.0.x до 3.0.19, 3.1.x–3.1.10 и 3.2.x–3.2.11 неправильно ограничивает приведение строк values, что позволяет удаленным злоумышленникам внедрять объекты и выполнять произвольный код или вызывать «отказ в обслуживании» с использованием вложенных ссылок на объекты XML, используя поддержку Action Pack.	CVE-2020-7220. Инструменты безопасного хранения конфиденциальной информации в динамических облачных средах HashiCorp Vault и Vault Enterprise версий с 0.11.0 по 1.3.1 Go модуля, чей API использует Dgraph, при определенных обстоятельствах аннулируют динамические секреты для монтирования в удаленном пространстве имен.

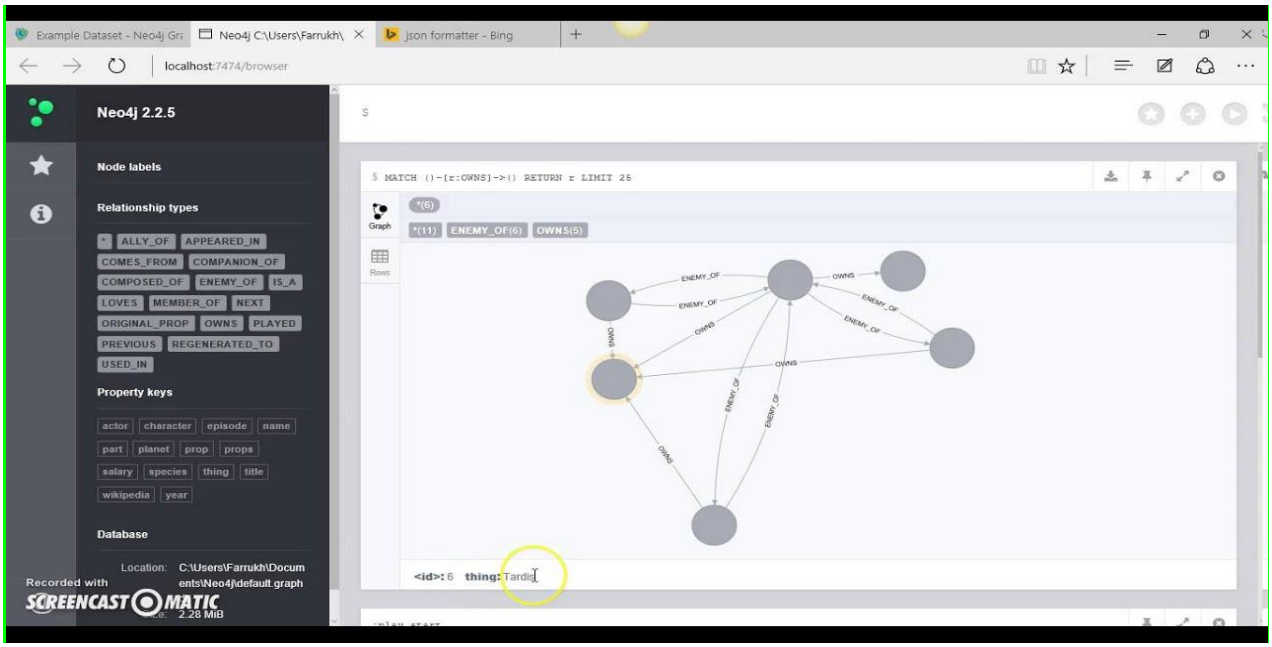

Рис. 2. REST-API Neo4j
(Fig. 2. REST-API Neo4j)

Перечислим наиболее распространенные методы защиты СУБД.

1. *Разграничение прав доступа и аутентификация*. Это гибкая система многопользовательской СУБД, при которой администратор предоставляет пользователям права доступа исходя из минимальных полномочий, необходимых для выполнения должностных обязанностей. Большинство СУБД имеют встроенный набор средств по разграничению прав доступа, с помощью которого пользователи и/или группы наделяются правами. В СУБД имеется возможность управления правами и действиями над определенными объектами, такими как чтение, добавление, изменение и удаление записей.

Одним из примеров применения разграничения доступа является ядро модели безопасности Neo4j с предопределенными ролями доступа пользователей, которые включают в себя набор разрешенных действий. Графовая СУБД Neo4j осуществляет аутентификацию пользователей с помощью собственного сервиса, который хранит данные о пользователях и ролях локально, на диске [15]. Другим вариантом авторизации пользователей в Neo4j является использование внешнего ПО, такого как Active Directory или OpenLDAP. Также реализовать аутентификацию и единый вход Neo4j можно с помощью дополнения Neo4j Kerberos.

В СУБД JanusGraph используются соединения, основанные на HTTP-запросах или WebSockets. Соединения выполняются по HTTPS (HyperText Transfer Protocol Secure), запрашивающему аутентификацию пользователей. HTTP-запросы реализуют базовую аутентификацию и аутентификацию с помощью токенов. В отличие от обычной аутентификации для СУБД (отправки имени пользователя и пароля вместе с запросом) однократная регистрация, WebSockets и аутентификация с помощью токенов повышают производительность.

2. *Шифрование данных*. Отсутствие шифрования позволяет получить доступ к СУБД с помощью коммуникационного канала, а использование средств криптографической защиты позволяет предотвратить данную угрозу. Шифрование должно использоваться при хранении и передаче конфиденциальных данных в зашифрованном

виде, а также при преобразовании исходных данных специальным алгоритмом для сокрытия их содержания. Новое представление зашифрованных данных с секретным ключом шифрования хранится в БД и передается по коммуникационному каналу [14]. Существует два режима работы зашифрованных БД:

а) расшифрование файла с конфиденциальной информацией на внешнем носителе и работа с ним, после чего файл вновь зашифровывается. Независимое взаимодействие СУБД и средств шифрования является очевидным достоинством данного режима, но при сбое или отказе часть СУБД может остаться незашифрованной;

б) расшифрование файла происходит в оперативной памяти перед выполнением действий с конфиденциальной информацией. Такие процедуры в большинстве случаев встроены в СУБД, что позволяет поддерживать высокий уровень защиты от НСД, но снижает производительность из-за усложнения процесса обработки данных.

В настоящее время графовая СУБД Neo4j [15] не шифрует данных. Дополнительная защита данных осуществляется с помощью шифрования файловой системы или самих данных из приложения. Шифрование файловой системы является простым способом усиления защиты данных на диске, но недостаточным для осуществления полной защиты данных. Neo4j использует архитектуру, основанную на REST и операторах Cypher в виде вызова веб-службы. Ответы на вызовы передаются по сети открытым текстом. Некоторые приложения кроме использования протокола HTTPS требуют введения дополнительных средств защиты. Одним из дополнительных способов является ограничение доступа к данным – только авторизованным для данной работы пользователям. В ходе процесса шифрования данных на уровне приложений они динамически изменяются, выполняется зашифрование и расшифрование до/после чтения или записи данных в БД. С помощью защиты на уровне приложения могут быть реализованы многие стандарты безопасности для областей здравоохранения и образования, такие как HIPPA (Health Insurance Portability and Accountability Act) и FERPA (Family Educational Rights and Privacy Act). Также для реализации защиты приложений на основе Java используется библиотека Neo4j Object Graph Mapping (OGM), которая позволяет конвертировать атрибуты, поскольку Neo4j может сохранять данные в формате, отличном от начального (например, дату в виде формата Long или String). Такие преобразования являются начальной точкой для шифрования на уровне приложений. Данный подход обеспечивает защиту данных как на диске, как и во время передачи от/на сервер Neo4j. Но для шифрования данных задействуются память и вычислительные мощности, что негативно сказывается на использовании ресурсов сети. Зашифрованные данные трудны для использования вне приложения и не пригодны для таких действий над данными в БД, как поиски, индексация и случайные запросы Cypher.

Графовая СУБД JanusGraph способна взаимодействовать со множеством продуктов, осуществляющих шифрование.

Графовая СУБД Dgraph [16] может шифровать данные в состоянии покоя HDFS (Hadoop Distributed File System), что позволяет хранить их в зашифрованных каталогах HDFS (зонах шифрования). Расшифрованные данные никогда не хранятся в каталогах HDFS, так как зашифрование и расшифрование данных происходит на стороне клиента.

3. *Защита полей, целостность данных*. Изменение данных в СУБД чаще всего происходит в результате действий пользователей, поэтому целесообразным является использовать защиту полей и записей в таблицах СУБД. Для защиты полей используются следующие уровни прав доступа: полный запрет доступа, только чтение и полный доступ для просмотра, ввода, изменения и удаления. Также применяется сокрытие от избранных

пользователей полей таблиц и запрет на вызов конструктора, чтобы пользователь не смог изменить приложение.

В графовой СУБД Neo4j защита полей осуществляется с использованием ограничений узлов и отношений, что позволяет обеспечить целостность данных. Создаются уникальные ограничения свойств, а также ограничения существования свойств узлов и отношений. Реализованы ключи узлов, гарантирующие уникальность значения свойств для всех узлов с данной меткой. Ограничение существования свойств узлов и отношений определяет их существование только с данной меткой/типом. При таком методе защиты запросы на создание новых узлов и отношений без свойств выполнены не будут. Ключи узлов позволяют гарантировать, что на всех узлах с данной меткой существуют определенные свойства, и их значения свойств уникальны.

В графовой СУБД JanusGraph целостность данных гарантируется сторонними продуктами, контроль целостности осуществляется при загрузке данных.

Графовая СУБД Dgraph представляет хранилище данных как верхний слой над другой БД SQL/NoSQL, и именно она отвечает за целостность данных.

4. *Контрольный след выполняемых операций* отображает детальные сведения в СУБД о действиях пользователя. Данная информация позволяет обнаруживать несанкционированные вмешательства, выявлять уязвимости в защите и предотвращать некорректные изменения данных в СУБД. При работе с важными данными или при выполнении критических операций всегда возникает необходимость регистрации контрольного следа выполняемых операций. Если, например, противоречивость данных приводит к подозрению, что совершено несанкционированное вмешательство в БД, то контрольный след используется для прояснения ситуации и подтверждения того, что все процессы находятся под контролем. Если это не так, то контрольный след поможет, по крайней мере, обнаружить нарушителя. Для сохранения контрольного следа обычно используется особый файл, в котором система автоматически записывает все выполненные пользователями операции при работе с БД.

5. *Резервное копирование* или «бэкап» (backup copy). Это создание копии файлов и папок на дополнительном носителе информации (внешнем жестком диске, CD/DVD-диске, флэш-памяти, в облачном хранилище и т.д.). Резервное копирование необходимо для восстановления данных, если они повредились или разрушились в основном месте их хранения (на внутреннем жестком диске компьютера или флэш-памяти мобильного устройства). Осуществляет восстановление данных на случай аппаратных или программных сбоев. Многие СУБД имеют инструменты, похожие по принципу создания резервных копий как одного компьютера, так и кластера. Существует режим копирования онлайн.

Графовая СУБД Neo4j имеет открытый исходный код и обеспечивает ACID-совместимый транзакционный сервер. Данные в Neo4j хранятся точно так же, как и на диске, а БД использует указатели для навигации и перемещения по графу. Для производственных сценариев Neo4j обеспечивает поддержку кластера и отказоустойчивость во время выполнения. При необходимости автоматизировать процесс резервного копирования и восстановления БД можно использовать возможности управления конфигурацией Ansible, который является открытым исходным кодом, и способен повышать масштабируемость, согласованность и надежность любой ИТ-среды. Ansible также можно использовать для автоматизации таких задач, как подготовка серверов, необходимых в инфраструктуре, и для управления конфигурацией или развертывания приложений.

Описанные методы обеспечения ИБ определяют основные средств защиты информации, которые необходимы для информационных систем в целом и в графовых СУБД в частности.

4. Средства, используемые для устранения уязвимостей графовых СУБД

В табл. 3 приведены уязвимости графовых СУБД и средства их устранения.

Таблица 3. Средства устранения уязвимостей графовых СУБД

Уязвимости	Средства устранения уязвимостей
Уязвимость в системе аутентификации Уязвимость в системе авторизации	Использование средств разграничения доступа: • Active Directory, OpenLDAP на основе сетевых протоколов аутентификации LDAP (Lightweight Directory Access Protocol) и Kerberos; • аутентификация с помощью токенов; • использование компонентов экосистемы Apache Hadoop.
Уязвимость, вызывающая переполнение буфера и отказ в обслуживании	Использование Apache Hadoop для хранения данных: • распределенная между узлами вычислительного кластера файловая система HDFS (Hadoop Distributed File System); • MapReduce для распределенных операций предварительной обработки.
Нешифрованный текст	Использование средств шифрования: • алгоритмы шифрования AES (Advanced Encryption Standard); • использование HTTPS для шифрования сетевого взаимодействия; • использование компонента экосистемы Apache Hadoop. Cloudera, обеспечивающего шифрование данных HDFS-файлов (Hadoop Distributed File System).
Уязвимость, допускающая инъекции в регулярных выражениях Уязвимость, допускающая инъекции кода Возможность манипуляции с REST-интерфейсом	Проверка входных данных: • использование компонента экосистемы Apache Hadoop Native Auditing, журналы аудита периметра на шлюзе Knox, мониторинг запросов доступа, операций обработки и изменения данных; • ограничение использования регулярных выражений и REST-интерфейса.

Заключение

В настоящее время безопасность СУБД является одним из значительных аспектов обеспечения ИБ организации. Количество данных продолжает увеличиваться экспоненциально, что непосредственно влияет на способы хранения данных и устаревание. Получение доступа к данным СУБД подразумевает полный контроль над ними и перехват управления внутренними сервисами ресурса злоумышленником. Здесь важно осознание и устранение способствующих этому уязвимостей СУБД. Обеспечение ИБ современных информационных систем, включая СУБД, требует комплексного подхода, что невозможно без широкого применения набора защитных средств, объединенных в единую инфраструктуру. Многие из таких средств получили распространение не только в мировом масштабе, но и в России. В данных условиях обеспечение ИБ должно быть динамичным и непрерывным, включая согласованную деятельность всех заинтересованных сторон – как пользователей, так и администраторов.

Графовые СУБД можно представить, как совокупность данных, совместно хранящихся и обрабатывающихся в соответствии с определенными правилами. Они широко используются организациями по всему миру. Их производители постоянно усовершенствуют и обновляют свои продукты, в связи с чем растет количество

потребителей. С появлением нового формата хранения данных в нереляционных графовых СУБД обычные атаки претерпевают изменения. Поэтому обеспечение полноценной и работоспособной системы обеспечения ИБ для графовых СУБД требует немало сил и финансовых вложений. Также необходимо своевременно и корректно устранять имеющиеся в них уязвимости.

В ходе данного исследования были рассмотрены популярные графовые СУБД по данным DB-Engine за июнь 2020 г. – Neo4j, JanusGraph и Dgraph, проведено сравнение их функциональных возможностей для определения преимуществ и недостатков. Описанные методы обеспечения ИБ определяют основные средств защиты информации, которые необходимы для информационных систем в целом и графовых СУБД в частности. Кроме это показано, какими средствами можно устранить типичные уязвимости графовых СУБД. Также было установлено, что не все из них могут быть устранены рассмотренными методами. Поэтому в последующей работе будет описано ПО, которое поможет закрыть оставшиеся уязвимости, связанные с инъекциями кода в графовых СУБД.

СПИСОК ЛИТЕРАТУРЫ:

1. Отчет Verizon об утечках данных. URL: https://www.dataarmor.ru/десятка-крупнейших-угроз-безопаснос/ (дата обращения: 15.10.2020).
2. NoSQL – Инъекции на примере нереляционной СУБД. URL: https://cyberleninka.ru/article/n/nosql-inektsii-na-primere-nerelyatsionnoy-subd-mongodb/viewer. (дата обращения: 15.10.2020).
3. Плаксий, Кирилл В.; Никифоров, Андрей А.; Милославская, Наталья Г. Исследование графовых субд, пригодных для работы с большими данными при обнаружении дел по отмыванию доходов, полученных преступным путем, и финансированию терроризма. Безопасность информационных технологий, [S.l.]. Т. 26, № 3. С. 103–116, 2019. ISSN 2074-7136. URL: https://bit.mephi.ru/index.php/bit/article/view/1222 Дата (дата обращения: 15.10.2020). DOI: http://dx.doi.org/10.26583/bit.2019.3.09.
4. Plaksiy K., Nikiforov A., Miloslavskaya N. Applying Big Data Technologies to Detect Cases of Money Laundering and Counter Financing of Terrorism. Proceedings of 2018 6th International Conference on Future Internet of Things and Cloud (FiCloud2018). Barcelona (Spain), 6-8 August 2018. P. 70–77. DOI: http://dx.doi.org/10.1109/W-FiCloud.2018.00017.
5. Харари Ф. Теория графов. М.: Мир. 1973. – 296 с.
6. Neo4j – Платформа для связанных данных. URL: https://neo4j.com/ (дата обращения: 15.10.2020).
7. Fernandes D., Bernardino J. Graph Databases Comparison: AllegroGraph, ArangoDB, InfiniteGraph, Neo4J, and OrientDB. DATA. 2018. P. 373–380. DOI: https://doi.org/10.5220/0006910203730380.
8. JanusGraph. URL: https://janusgraph.org/ (дата обращения: 15.10.2020).
9. DB-Engines Рейтинг графовых БД. URL: https://db-engines.com/en/ranking/graph+dbms. (дата обращения: 15.10.2020).
10. Базовая модель угроз безопасности персональных данных при их обработке в информационных системах персональных данных. URL: https://fstec.ru/component/attachments/download/289. (дата обращения: 15.10.2020).
11. Распространенные уязвимости Neo4j. URL: https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=neo4j (дата обращения: 15.10.2020).
12. National Vulnerability Database. URL: https://nvd.nist.gov/vuln/search/results?form_type=Basic&results_type=overview&query=Neo4j&search_type=all (дата обращения: 15.10.2020).
13. Go Module Vulnerabilities. URL: https://github.com/dgraph-io/dgraph/issues/5569 (дата обращения: 15.10.2020).
14. DataArmor. С чего начинается защита базы данных? URL: https://www.dataarmor.ru/%D1%81-%D1%87%D0%B5%D0%B3%D0%BE-%D0%BD%D0%B0%D1%87%D0%B8%D0%BD%D0%B0%D0%B5%D1%82%D1%81%D1%8F-%D0%B7%D0%B0%D1%89%D0%B8%D1%82%D0%B0-%D0%B1%D0%B0%D0%B7%D1%8B-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85/ (дата обращения: 15.10.2020).
15. Габриелян Г.А. Графовая база данных NEO4J для проектирования высоконагруженных систем. Студенческий электрон. научн. журн. 2018. № 11(31). URL: https://sibac.info/journal/student/31/111409 (дата обращения: 15.10.2020).
16. Dgraph. URL: https://dgraph.io/ (дата обращения: 15.10.2020).

17. Data & Insight. 6 категорий решений для защиты Big Data в Apache Hadoop URL: https://dis-group.ru/company-news/articles/6-kategorij-reshenij-dlya-zashhity-big-data-v-apache-hadoop/ (дата обращения: 15.10.2020).

REFERENCES:

[1] Verizon data breach report. URL: https://www.dataarmor.ru/десятка-крупнейших-угроз-безопаснос/ (accessed: 15.10.2020) (in Russian).
[2] Fremuchkov A.N. NoSQL — Injection on the example of a non-relational DBMS. URL: https://cyberleninka.ru/article/n/nosql-inektsii-na-primere-nerelyatsionnoy-subd-mongodb/viewer (accessed: 15.10.2020) (in Russian).
[3] Plaksiy, Kirill V.; Nikiforov, Andrey A.; Miloslavskaya, Natalia G. Investigation of graph databases suitable for work with big data while detecting money laundering and terrorism financing cases. IT Security (Russia), [S.l.]. Vol. 26, no. 3. P. 103–116, 2019. ISSN 2074-7136. URL: https://bit.mephi.ru/index.php/bit/article/view/1222 (accessed: 15.10.2020). 2020. DOI: http://dx.doi.org/10.26583/bit.2019.3.09 (in Russian).
[4] Plaksiy K., Nikiforov A., Miloslavskaya N. Applying Big Data Technologies to Detect Cases of Money Laundering and Counter Financing of Terrorism. Proceedings of 2018 6th International Conference on Future Internet of Things and Cloud (FiCloud2018). Barcelona (Spain), 6-8 August 2018. P. 70–77. DOI: http://dx.doi.org/10.1109/W-FiCloud.2018.00017.
[5] Harari F. Graph theory. M.: Mir, 1973. 296 p. (in Russian).
[6] Neo4j – Platform for connected data. URL: https://neo4j.com/ (accessed: 15.10.2020).
[7] Fernandes D., Bernardino J. Graph Databases Comparison: AllegroGraph, ArangoDB, InfiniteGraph, Neo4J, and OrientDB. DATA. 2018. P. 373–380. DOI: https://doi.org/10.5220/0006910203730380.
[8] JanusGraph. URL: https://janusgraph.org/ (accessed: 15.10.2020).
[9] DB-Engines Ranking of Graph DBMS. URL: https://db-engines.com/en/ranking/graph+dbms (accessed: 15.10.2020).
[10] The basic model of threats to the security of personal data during their processing in information systems personal data. URL: https://fstec.ru/component/attachments/download/289 (accessed: 15.10.2020) (in Russian).
[11] Common Vulnerabilities and Exposures Neo4j. URL: https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=neo4j (accessed: 15.10.2020).
[12] National Vulnerability Database. URL: https://nvd.nist.gov/vuln/search/results?form_type=Basic&results_type=overview&query=Neo4j&search_type=all (accessed: 15.10.2020).
[13] Go Module Vulnerabilities. URL: https://github.com/dgraph-io/dgraph/issues/5569 (accessed: 15.10.2020).
[14] DataArmor. Where does database protection begin? URL: https://www.dataarmor.ru/%D1%81-%D1%87%D0%B5%D0%B3%D0%BE-%D0%BD%D0%B0%D1%87%D0%B8%D0%BD%D0%B0%D0%B5% D1%82%D1%81%D1%8F-%D0%B7%D0%B0%D1%89%D0%B8%D1%82%D0%B0-%D0%B1%D0%B0% D0%B7%D1%8B-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85/ (accessed: 15.10.2020).
[15] Gabrielan G.A. Graph database NEO4J for the design of high-load systems. Student electron. scientific. journal 2018. № 11(31). URL: https://sibac.info/journal/student/31/111409 (accessed: 15.10.2020) (in Russian).
[16] Dgraph. URL: https://dgraph.io/ (accessed: 15.10.2020).
[17] Data & Insight. 6 categories of security solutions Big Data in Apache Hadoop. URL: https://dis-group.ru/company-news/articles/6-kategorij-reshenij-dlya-zashhity-big-data-v-apache-hadoop/ (accessed: 15.10.2020) (in Russian).

Поступила в редакцию – 15 октября 2020 г. Окончательный вариант – 05 ноября 2020 г.
Received – October 15, 2020. The final version – November 05, 2020.