DOI: https://doi.org/10.32836/2521-6643-2018-1-56-2
УДК 004.056.5

В. О. Шапорин, кандидат технических наук, доцент кафедры компьютерных интеллектуальных систем и сетей Одесского национального политехнического университета
П. М. Тишин, кандидат физико-математических наук, доцент кафедры компьютерных интеллектуальных систем и сетей Одесского национального политехнического университета
Е. Л. Шапорина, старший преподаватель кафедры компьютерных интеллектуальных систем и сетей Одесского национального политехнического университета

ОНТОЛОГИЯ УЯЗВИМОСТИ В СИСТЕМАХ SCADA

Статья представляет исследование в области построения онтологических моделей уязвимости систем SCADA. Метод проектирования онтологии базируется на языке описания онтологии OWL и на использовании базы данных известных типов уязвимостей. При построении онтологии уязвимости учитывается иерархичность влияния параметров модели, что позволяет описывать унифицированные модели уязвимости с повышением детализации общего описания уязвимости. Предложенная онтологическая модель реализована в системе моделирования Protégé.

Ключевые слова: *управление уязвимостями; SCADA системы; Common Weakness Enumeration; дескриптивная логика; Web Ontology Language.*

Стаття подає дослідження у сфері побудови онтологічних моделей вразливості систем SCADA. Метод проектування онтології базується на мові опису онтології OWL і на використанні бази даних відомих типів вразливостей. У побудові онтології вразливості враховується ієрархічність впливу параметрів моделі, що дає можливість описувати уніфіковані моделі вразливості з підвищенням деталізації загального опису вразливості. Запропонована онтологічна модель реалізована в системі моделювання Protégé.

Ключові слова: *управління вразливостями; SCADA системи; Common Weakness Enumeration; дескриптивна логіка; Web Ontology Language.*

The main feature of information security is that it is not a state, but a continuous process of analyzing the security of computer systems and networks and their components. Modern information systems are often is the part of public services, what make a large number of challenges for system and security administrators. The main problem is that the all hardware and software of this systems has a many vulnerabilities, because it's a complex, modular and multi-vendor tools and devices. So it's important to have an opportunity to manage the weakness of this tools and decrease the probability of network attack, using this vulnerabilities. This article presents a study in the field of building ontological models of vulnerability of SCADA systems. The ontology design method is based on the OWL ontology description language and on the use of a database of known types of vulnerabilities. The paper proposes a system of axioms and related classes describing the vulnerability of hardware and software. This system is allowed security engineers to have a tool for flexible and effective representation of systems weakness and vulnerabilities. When constructing an ontology of vulnerability, the hierarchy of the influence of model parameters is taken into account, which makes it possible to describe unified vulnerability models with increased detail in the general description of vulnerability. The proposed ontological model is implemented in the Protégé modeling system.

Testing of this method was carried out in the university laboratory. Testing was perform during regular inventory of the classroom and troubleshooting current network infrastructure. The implementing of this method is make possible to decrease a time of analyzing the state of the target system. Also this method provide a decreasing of the risk of injection types of attacks in networks. The proposed method allows the use of not only external sources of information about vulnerabilities, but also the knowledge and experience gained during the operation of the target system by administrators and analysts in the processes of analyzing systems and networks.

Key words*: vulnerability management; SCADA systems; Common Weakness Enumeration; descriptive logic; Web Ontology Language.*

Постановка проблемы. В условиях интеграции автоматизированных систем управления технологическим процессом с корпоративными сетями и Internet все чаще возникают вопросы обеспечения сетевой безопасности, устойчивости сети предприятия к внутренним и внешним угрозам, а также к обеспечению антивирусной защиты.

Независимо от направления деятельности организации, от простой автоматизации рутинных механизмов к системам критического применения (энергетика, транспорт, медицина), большинство процессов управления свя-

заны с системами SCADA, которые реализуют диспетчеризацию, сбор и обработку данных.

С ростом популярности технологий Интернета вещей такие системы все чаще подвергаются вирусным и целенаправленным атакам, целью которых является как замедление или остановка производственных процессов, так и разрушение систем или же нанесение вреда людям и окружающей среде. Вместе с этим до сих пор такие атаки редко диагностируются в связи с отсутствием или ограниченностью механизмов определения уязвимости и вредоносных процессов в системах. Поэтому сбои, вызванные вирусами и злоумышленниками, расцениваются как ошибки оборудования или программного обеспечения.

Цели атак на промышленные информационно-управляющие системы, как правило, схожи с атаками на корпоративные системы – доказательство возможности проникновения, кража информации или разрушение системы. Однако существуют некоторые отличия в методах атак на такие системы. Например, простая задержка в передаче информации, которая не приводит к существенным потерям в корпоративных сетях, может оказаться фатальной в промышленных системах в связи с потерей актуальности информации или ее несвоевременностью.

Также могут отличаться и уязвимости промышленных систем. На волне популярности многие производители промышленных систем, стремясь вывести продукт раньше конкурентов, не уделяют должного внимания безопасности программного или аппаратного обеспечения. Это приводит к тому, что простую атаку типа “отказ в обслуживании” осуществляют с использованием интеллектуальных лампочек, чайников и т. п.

Беря во внимание рассмотренные проблемы, в данной работе было поставлено решение следующих задач:

– с помощью семантических технологий формально и точно описать структуру внешних и внутренних уязвимостей;

– введение совокупности понятий, отношений и аксиом предметной области управления уязвимостями;

– построение онтологии уязвимости в системах SCADA, которая определяет ключевые концепции управления уязвимостью и их отношение.

Анализ последних исследований и публикаций. Основная особенность информационной безопасности заключается в том, что это не состояние, а непрерывный процесс анализа защищенности компьютерных систем, сетей, и их компонентов. Базовым семейством стандартов в области информационной безопасности является ISO 27000 [1] и его дополнения, в которых описываются терминология, методы защиты, методы контроля безопасности и т. д. Одним из основных элементов, которому уделяется внимание

в данных документах, является управление уязвимостью. Национальным институтом стандартов и технологий США предложен общий подход к описанию уязвимости системы [2] с использованием инструментов баз знаний [3], который дает общее понимание того, как идентифицировать уязвимость и какие элементы могут выступать в качестве определяющих параметров данного описания. При этом документ не регламентирует, какие именно инструментальные средства следует использовать и какие именно параметры уязвимости выделять как ключевые. Вместе с тем широкую популярность приобретают подходы к информационной безопасности с использованием различных методов и разделов искусственного интеллекта [4–6]. Учитывая такую тенденцию, авторами было проведено исследование, которое позволяет оценить использование OWL онтологии как способа хранения знаний [7; 8] об уязвимости, что позволит использовать накопленный опыт анализа защищенности систем для сокращения временных затрат на повторный анализ или портирования знаний для анализа других систем.

Цель статьи – развитие методов анализа уязвимостей в информационных системах и компьютерных сетях за счет построения онтологий, использующихся в области информационной безопасности. Ожидается, что применение онтологий уязвимости позволит сократить время, необходимое для анализа защищенности систем и сетей, а также поддержания актуальности их программного и аппаратного обеспечения.

Изложение основного материала. Для создания онтологии уязвимости в системах SCADA необходимо ввести совокупность понятий, отношений и аксиом предметной области. За основу для представления в общем виде множества понятий и отношений между классами взята работа [2].

Общий вид описания уязвимости и взаимосвязей ее элементов представлен на рис. 1.

Для построения онтологии уязвимости в статье использовались базы данных известных уязвимостей Common Vulnerability and Exposures и Common Weakness Enumeration. Данные источники позволяют описывать как основные характеристики уязвимости, так и способствуют определению специфических понятий, введенных в [1]. В качестве инструментального средства построения онтологии использовался прикладной пакет Protege [9].

Учитывая, что одна уязвимость может порождаться использованием иной уязвимости, необходимо также определить структуру взаимодействия между ними. Таким образом можно устанавливать родственные уязвимости и отношения между ними, как правило, типа Child Of, Parent Of или Member Of.

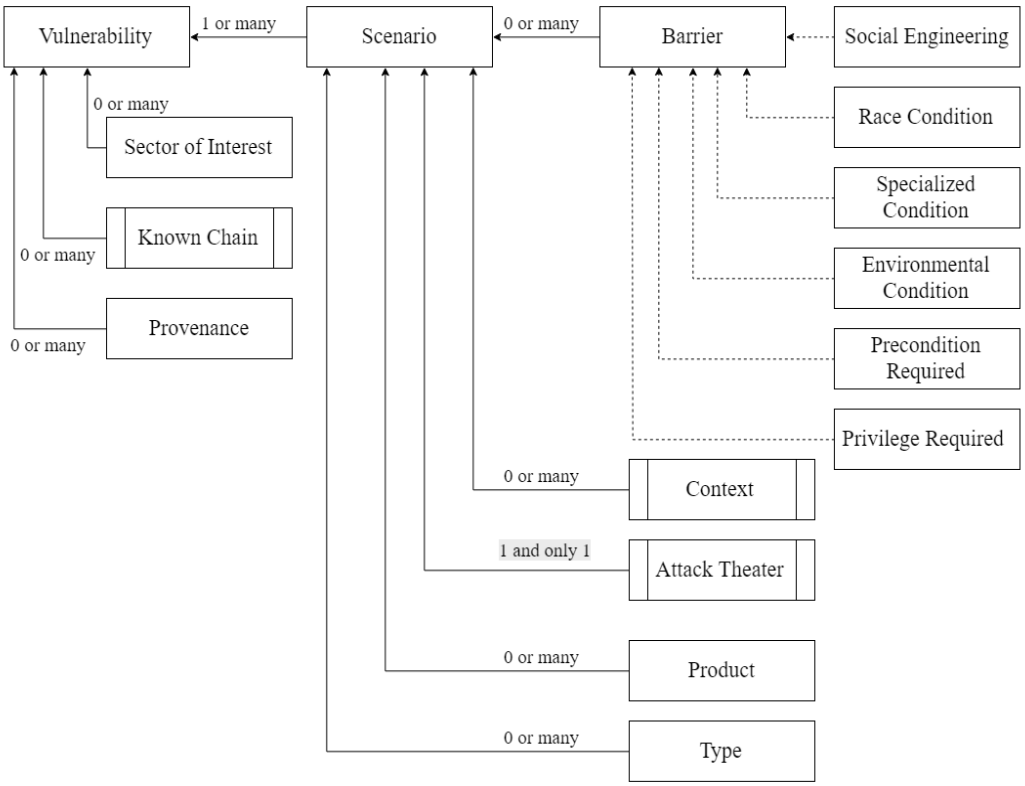

Рис. 1. Онтология уязвимости в системах SCADA

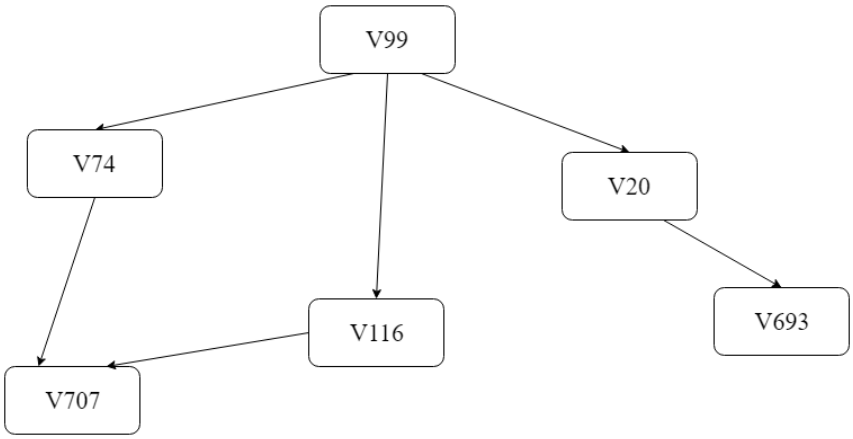

Рис. 2. Структурная сеть взаимосвязей уязвимости

Для описания классов и отношений между ними в качестве примера использовалась уязвимость CWE-99 Improper Controlof Resource Identifiers ('Resource Injection').

Класс VULNERABILITY. Класс “уязвимость” вводится для идентификации выявленной уязвимости. Каждой уязвимости должен быть присвоен свой уникальный идентификатор, который имеет функцию упрощения системы поиска и препятствование неправильному восприятию информации. Поскольку собственный идентификатор задается численным значением, это препятствует пониманию того, что имеется в виду под идентификатором. Для этого необходимо ввести название уязвимости. Название уязвимости должно содержать общую краткую информацию об уязвимости. Название должно быть согласовано на международном уровне для исключения возможных неточных или ошибочных утверждений, которые могут привести к тому, что эксперты не смогут в полном объеме, или вообще, правильно понять содержание уязвимости. Необходимо отметить дату создания и дату модификации информации для отображения актуальности информации о данной уязвимости. Также необходимо ввести поле, которое предоставит дополнительную информацию об уязвимости, если в этом может возникнуть необходимость. Дополнительная информация может дать эксперту больше понимания об уязвимости. Исследовав необходимые параметры к этому классу, можно выделить следующие свойства у класса “уязвимость”:

– has_vulnerability_id – это уникальный идентификатор, определяющий уязвимость;

– has_vulnerability_name – это атрибут, который является текстовым описанием уязвимости;

– has_vulnerability_version – определяет номер версии уязвимости;

– has_vulnerability_creation_date – определяет дату, когда была создана уязвимость (для этой версии);

– has_vulnerability_modification_date – определяет дату, когда уязвимость была последний раз модифицирована (для этой версии);

– has_vulnerability_documentation – может быть добавлена дополнительная информация об уязвимости.

Относительно этого класса и его свойств можно сформулировать следующие аксиомы. Данные аксиомы являются утверждениями, справедливыми для данной предметной области:

Аксиома 1. Областью значений (range) свойства has_vulnerability_id является xsd: int.

Это означает, что данное свойство будет определяться численным значением, которое будет задавать эксперт при создании уязвимости.

Аксиома 2. Областью определения (domain) свойства has_vulnerability_id является класс VULNERABILITY.

Это означает, что вся полученная информация, которая была введена экспертом, будет храниться в классе VULNERABILITY (уязвимость).

Для свойства has_vulnerability_id выполняется следующее ограничение:

Аксиома 3.

$$\text{VULNERABILITY} \subseteq (= 1)\text{has\_vulnerability\_id}, \quad (1)$$

где VULNERABILITY – класс уязвимости;

has_vulnerability_id – имя свойства класса “уязвимость”.

Ограничение (1) утверждает, что каждой уязвимости, которая задается в базу знаний, должен быть поставлен в соотношение один и только один объект, предоставляющий уязвимости уникальный идентификатор.

Для уязвимости “V99” уникальным идентификатором в рамках онтологии уязвимости, согласно базе данных CWE, является численное значение “99”. Численное значение “99” не может быть предоставлено более ни одной другой уязвимости в рамках данной онтологии и делает этот идентификатор уникальным.

Аксиома 4. Областью значений (range) свойства has_vulnerability_name является xsd: string.

Это означает, что данное свойство определяется строчным значением, которое предоставляет эксперт при создании уязвимости.

Аксиома 5. Областью определения (domain) свойства has_vulnerability_name является класс VULNERABILITY.

Из этого следует, что значение, которое предоставил эксперт, будет храниться в классе VULNERABILITY (уязвимость).

Для свойства has_vulnerability_name выполняется следующее ограничение:

Аксиома 6.

$$\text{VULNERABILITY} \subseteq (= 1)\text{has\_vulnerability\_ name}, \quad (2)$$

где VULNERABILITY – это класс уязвимости;

has_vulnerability_name – это имя свойства класса “уязвимость”.

Ограничение (2) утверждает, что в каждой уязвимости, которая задается в базу знаний, должен быть поставлен в соотношение один и только один объект, предоставляющий уязвимости имя.

Для уязвимости “V99” названием в рамках онтологии уязвимости, согласно базе данных CWE, является строчное значение “неправильный кон-

троль идентификаторов ресурсов (“Вливание ресурсов”)”. Строчное значение является кратким описанием уязвимости в рамках данной онтологии, которое было согласовано между экспертами этой предметной области, что позволит избежать вероятных неточных или ошибочных утверждений, могущих привести к тому, что эксперты не смогут в полном объеме, или вообще, правильно понять содержание уязвимости.

Последующие классы и уязвимости описываются по тому же принципу, и общая схема примет вид, как на рис. 3.

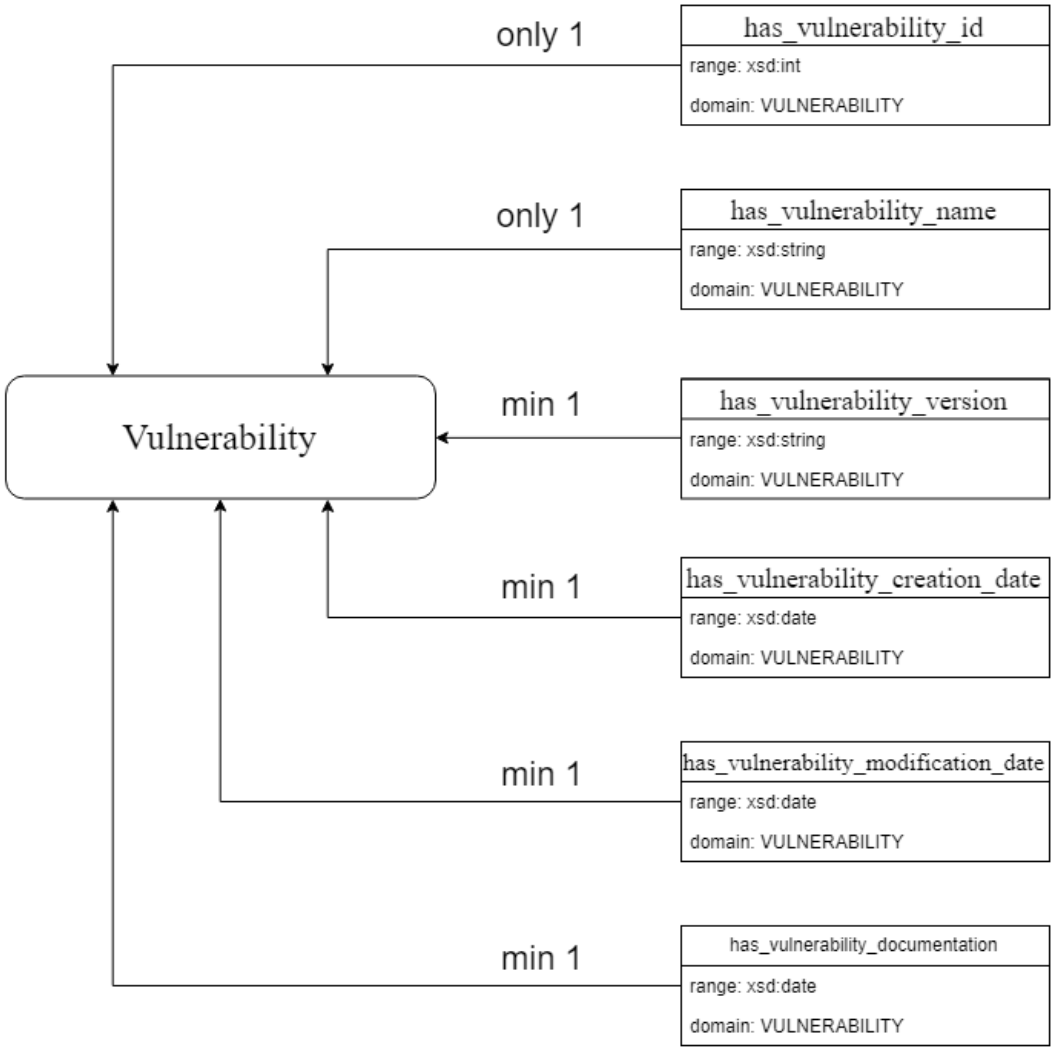

Рис. 3. Класс уязвимости с его свойствами

Отдельное внимание стоит уделить связям между классами. После создания классов онтологии уязвимости в системах SCADA необходимо установить отношение между классами. В рамках данной предметной области отношение между классами строится по особой процедуре.

Было решено ввести специализированные классы-отношения, которые предоставили бы возможность увеличить эффективность онтологии уязвимости.

Если между двумя классами необходимо задать отношение, то вводится дополнительный класс, имеющий свойства, которые связывают его с исходными классами. Схема данного строительства отображена на рис. 4.

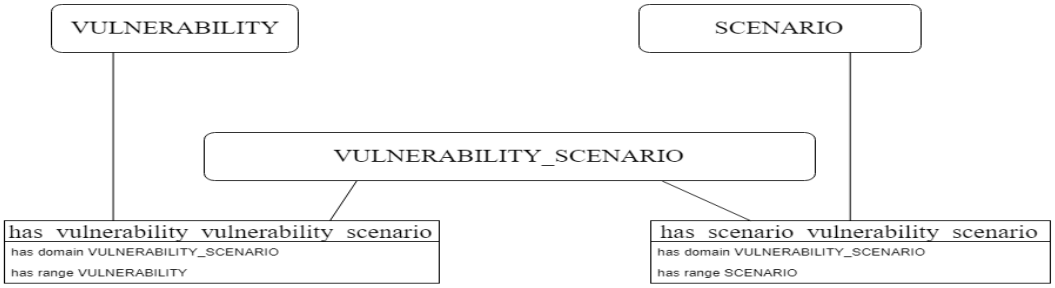

Рис. 4. Схема класса-отношения “уязвимость-сценарий”

Ниже приведена структура создания классов-отношений для онтологии уязвимости в системах SCADA.

Класс-отношение VULNERABILITY_SCENARIO. Класс-отношение “уязвимость-сценарий” введено для того, чтобы установить отношение между классом “уязвимость” и классом “сценарий”. Этому классу необходимо установить минимум два свойства, каждое из которых должно иметь способность установить связь между этими двумя классами. Проанализировав все требования к этому классу, можно установить следующие свойства:

– has_vulnerability_vulnerability_scenario – свойство, которое связывает с классом уязвимость;

– has_scenario_vulnerability_scenario – свойство, которое связывает с классом сценарий.

Относительно этого класса и его свойств можно сформулировать следующие аксиомы.

Аксиома 7. Областью значений (range) свойства has_vulnerability_ vulnerability_scenario является класс VULNERABILITY.

Это означает, что связь будет установлена от класса VULNERABILITY (уязвимость).

Аксиома 8. Областью определения (domain) свойства has_vulnerability_ vulnerability_scenario является класс VULNERABILITY_SCENARIO.

Это означает, что связь от класса VULNERABILITY (уязвимость) будет храниться в классе VULNERABILITY_SCENARIO (уязвимость-сценарий).

Для свойства has_vulnerability_vulnerability_scenario выполняется следующее ограничение:

Аксиома 9

$$\text{VULNERABILITY}_{\text{SCENARIO}} \subseteq (\geq = 1)\text{has\_vulnerability\_vulnerability\_scenario}, \quad (3)$$

где VULNERABILITY_SCENARIO – класс-отношение;

has_vulnerability_vulnerability_scenario – имя свойства класса "уязвимость-сценарий".

Ограничение (3) утверждает, что к каждому объекту класса-отношение "уязвимость-сценарий", задаваемый в базу знаний, должен быть поставлен в соотношение один или более объектов, который предоставляет отношение к классу "уязвимость".

Этим было изображено, как устанавливается связь между классом "уязвимость" и классом-отношением "уязвимость–сценарий".

Аксиома 10. Областью значений (range) свойства has_scenario_ vulnerability_scenario является класс SCENARIO.

Это означает, что связь будет установлена от класса SCENARIO (сценарий).

Аксиома 11. Областью определения (domain) свойства has_scenario_ vulnerability_scenario является класс VULNERABILITY_SCENARIO.

Это означает, что связь от класса SCENARIO (сценарий) будет храниться в классе VULNERABILITY_SCENARIO (уязвимость-сценарий).

Для свойства has_scenario_vulnerability_scenario выполняется следующее ограничение:

Аксиома 12.

$$\text{VULNERABILITY}_{\text{SCENARIO}} \subseteq \subseteq (> = 1)\text{has\_ scenario\_vulnerability\_scenario}, \quad (4)$$

где VULNERABILITY_SCENARIO – класс-отношение;

has_scenario_vulnerability_scenario – имя свойства класса "уязвимость-сценарий".

Ограничение (4) утверждает, что к каждому объекту класса-отношения "уязвимость-сценарий", который задается в базу знаний, должен быть по-

ставлен в соотношение один или более объектов, предоставляющих отношение к классу “сценарий”.

Этим было продемонстрировано, как устанавливается связь между классом “сценарий” и классом-отношением “уязвимость-сценарий”.

Таким образом была установлена связь между двумя уязвимостями, что, как ожидается, обеспечит преимущество перед другими способами, с помощью оптимизации онтологии. Этот класс будет не столько информативным, сколько инструментом оптимизации. Для установления связи необходимо в класс занести объект с названием, созданным от первых букв названий классов, которые необходимо соединить, и порядкового номера. В уязвимости “V99” объектом связи является “vs1”.

Выводы из данного исследования и перспективы дальнейших исследований в данном направлении. Скорость реакции на киберугрозы и ошибки в промышленных системах на сегодняшний день играют ключевую роль в стабильности работы систем критического применения. Поэтому существует необходимость в разработке технологий, методов и подходов, которые реализуют автоматизированный или автоматический поиск проблемных и уязвимых мест в сетях и системах, с целью снижения вероятности проникновения в систему или утечек информации.

Предложенный в работе метод позволяет использовать в процессах анализа систем и сетей не только внешние источники информации об уязвимостях, такие как mitre.org или другие ресурсы, но и знания, опыт, накопленные в процессе эксплуатации целевой системы администраторами и аналитиками.

Применение данного метода при анализе состояния системы в испытательной лаборатории позволили сократить время, затраченное на поиск уязвимости в компьютерной сети и ее элементах, приблизительно на 8 %. Также использование данного подхода снизило риск внедрения посторонних программных объектов в рассматриваемой сети приблизительно на 14 %.

Список использованных источников:

1. ISO/IEC 27000:2016. Information technology. Security techniques. Information security management systems. Overview and vocabulary. URL: https:// www.iso.org/standard/66435.html. ISO/IEC. Geneva, Switzerland, 2016. 41 p.

2. *Booth H., Turner Chr.* Vulnerability Description Ontology. A Framework for Characterizing Vulnerabilities. Gaithersburg: NIST, 2016. 45 p.

3. *Antoniou G., Harmelen Frank van.* OWL Web Ontology Language Overview // Handbook on Ontologies. Springer, Berlin, Heidelberg, 2004. P. 67–92. DOI.ORG/10.1007/978-3-540-24750-0_4

4. *Uschold M.* Building Ontologies: Towards a Unified Methodology // 16th Annual Conference of the British Computer Society Specialist Group on Expert Systems 16–18 December 1996. Cambridge, UK. 20 p.

5. *Tuzovsky A. F., Chirikov S. V., Yampolsky V. Z.* Knowledge management systems (methods and technologies) // NTL, Tomsk, 2005. 260 p.

6. *Brown K.* Encyclopedia of Language & Linguistics, Second Edition // Elsevier Science. Oxford, United Kingdom, 2006. P. 665–670.

7. *Konstantinova N. S., Mitrofanov O. A.* Ontologies as a knowledge storage system // SPDU, Saint Petersburg, 1990. 54 p.

8. *Thomas R. Gruber.* Toward Principles for the Design of Ontologies Used for Knowledge Sharing SKSL. Padowa, Italy, 1993. 602 p.

9. *Dosin D. G., Darevich R. R., Shkutyak N. V.* The opening of the ontology of materialism in Protégé-OWL artefacts // Artificial Intelligence, Lviv, Ukraine, 2008. 70–77 pp.

References:

1. ISO/IEC 27000:2016 (2016), Information technology. Security techniques. Information security management systems. Overview and vocabulary. Geneva, Switzerland, 41 p., available at: https://www.iso.org/standard/66435.html. ISO/IEC.

2. Booth H. and Turner Chr. (2016), Vulnerability Description Ontology. A Framework for Characterizing Vulnerabilities. Gaithersburg: NIST, 45 p.

3. Antoniou G. and Harmelen Frank van (2004), OWL Web Ontology Language Overview // Handbook on Ontologies. Springer, Berlin, Heidelberg, pp. 67–92. DOI.org/10.1007/978-3-540-24750-0_4

4. Uschold M. (1996) "Building Ontologies: Towards a Unified Methodology" // 16th Annual Conference of the British Computer Society Specialist Group on Expert Systems 16-18 December, Cambridge, UK, 20 p.

5. Tuzovsky A. F., Chirikov S. V. and Yampolsky V. Z. (2005), Knowledge management systems (methods and technologies) // NTL, Tomsk, 260 p.

6. Brown K. (2006), Encyclopedia of Language & Linguistics, Second Edition // Elsevier Science. Oxford, United Kingdom, 665–670 p.

7. Konstantinova N. S. and Mitrofanov O. A. (1990), Ontologies as a knowledge storage system // SPDU, Saint-Petersburg, 54 p.

8. Thomas R. Gruber (1993), Toward Principles for the Design of Ontologies Used for Knowledge Sharing SKSL. Padowa, Italy, 602 p.

9. Dosin D. G., Darevich R. R. and Shkutyak N. V. (2008), The opening of the ontology of materialism in Protégé-OWL artefacts // Artificial Intelligence. Lviv, Ukraine, pp. 70–77.