

ПРОГРАММНЫЕ СРЕДСТВА АНАЛИЗА ОТКАЗОУСТОЙЧИВОСТИ БЕСПРОВОДНЫХ СЕНСОРНЫХ СЕТЕЙ

В. Е. Стрельников

Новосибирский государственный технический университет,
630073, Новосибирск, Россия

УДК 004.056

В настоящее время беспроводным сенсорным сетям уделяется повышенное внимание в исследовательских центрах и коммерческих компаниях. Область применения технологий беспроводных сенсорных сетей достаточно обширна. Например, они могут применяться для мониторинга качества воздуха, контроля трафика, в новых методах медицинской диагностики, в системах „умный дом“ и др. Однако вопрос безопасности беспроводных сенсорных сетей до сих пор является открытым. При их разработке и внедрении необходимо принимать во внимание как известные, так и потенциальные атаки. В данной статье рассматриваются вопросы разработки программных средств для анализа отказоустойчивости беспроводных сенсорных сетей.

Ключевые слова: беспроводные сенсорные сети, отказоустойчивость, программное обеспечение.

Nowadays wireless sensor networks have got a lot of attention from the scientific centers and profit companies. The application area of this technology is vast. It can be potentially applied for air pollution monitoring, traffic control, advanced medical diagnostics, smart home systems etc. However, the problem of wireless sensor networks security is still open. It needs to take into account both known and potential attacks under design and deployment of wireless sensor networks. In this paper software tools for analysis of wireless sensor networks fault tolerance is offered.

Key words: wireless sensor networks, fault tolerance, software.

Введение. В настоящее время технологиям, основанным на использовании беспроводных сенсорных сетей, уделяется повышенное внимание в отечественных и иностранных исследовательских центрах. Аналитики заявляют об огромном рыночном потенциале указанных сетей. Интернет вещей [1, 2], составной частью которого являются беспроводные сенсорные сети, является стратегическим приоритетным направлением для крупнейших игроков IT-рынка, об этом говорилось на ряде международных научно-технических конференций представителями Intel, IBM, AMD, Google, Yahoo, Cisco, Samsung и др. По данным корпорации Cisco экономический эффект от соединения в сеть вещей, людей, процессов и данных оценивается в 19 трлн \$, из них 14 трлн придется на проекты частного сектора и 5 трлн — на госсектор.

Беспроводные сенсорные сети (БСС) основаны на совместной работе большого числа крошечных узлов, каждый из которых состоит из модуля сбора и обработки данных, источника питания, а также приемопередатчика. Некоторые узлы располагаются близко к

наблюдаемому явлению, осуществляют его мониторинг, другие узлы могут и не участвовать в мониторинге, но служить для передачи наблюдаемых сведений в центр сбора информации. Наиболее важной особенностью сенсорных сетей является способность самоорганизации, т. е. кооперация и совместная работа сетевых узлов для сбора, хранения, передачи и обработки информации. Узлы оснащены микропроцессором, поэтому вместо передачи исходных данных они могут их обрабатывать, выполняя простые вычисления, и передавать далее только необходимые и частично обработанные данные. Недавние достижения в области микроэлектроники, развитие платформ SmartMesh и SensiNet, разработка стандарта IEEE 802.15.4 для персональных беспроводных сетей, ратификация стандарта ZigBee позволили сделать шаг от теоретических изысканий к реальным приложениям беспроводных сенсорных сетей в самых разных сферах человеческой деятельности. В настоящее время беспроводные сенсорные сети находят применение в системах „умный дом“, системах контроля загрязнения окружающей среды, в системах оптимизации транспортных потоков, в медицинской диагностике, в системах военного назначения, в автоматических производственных линиях и т. д.

Вместе с распространением и внедрением БСС в критические приложения повышаются требования к их безопасности. Согласно требованиям рынка, сенсоры должны обладать низкой стоимостью и ограниченными ресурсами. Сенсорные узлы часто должны быть случайным образом распределены в неконтролируемой среде для сбора информации, требующей защиты. Передача данных от сенсора к базовой станции осуществляется через цепочку ненадежных узлов по беспроводному каналу. Все это значительно затрудняет обеспечение безопасности БСС. Тем не менее, сеть должна обладать отказоустойчивостью к преднамеренным и непреднамеренным разрушающим воздействиям. Необходимо не только надлежащим образом реагировать на известные атаки, но и своевременно выявлять потенциальные угрозы, чтобы избежать серьезных проблем в будущем.

На данный момент не существует какого-либо публично доступного программного обеспечения для анализа широкого спектра разрушающих воздействий в беспроводных сенсорных сетях. Ситуация осложняется тем, что, несмотря на интенсивные исследования, концепция построения беспроводной сенсорной сети окончательно не оформилась, регулярно подвергается пересмотру в зависимости от конкретных требований прикладной задачи. В условиях, когда системная архитектура и программно-аппаратная реализация БСС находятся на этапе интенсивного формирования, говорить о наличии устоявшихся средств анализа отказоустойчивости не приходится.

В данной работе предлагается инструментарий, позволяющий проводить анализ DoS атак „Blackhole“, „Jamming“ и ряда других, оценивать степень отказоустойчивости системы, анализировать эффективность защитных мер. Кроме того, уделяется внимание вопросам производительности предлагаемого программного обеспечения.

1. Проблемы безопасности беспроводных сенсорных сетей. Сенсорные сети могут содержать различные типы датчиков, например, сейсмические, датчики определения магнитного поля, тепловые, инфракрасные, акустические, которые в состоянии осуществлять самые разнообразные измерения условий окружающей среды. Например, такие как [3] температура, влажность, наличие движения, давление, состав почвы, уровень шума, присутствие или отсутствие некоторых объектов, механическая нагрузка, динамические характеристики, такие как скорость, направление и размер объекта. Возможности непрерывного зондирования и беспроводное соединение делают возможным использование БСС во многих областях, в том числе и в критических технологиях. Сюда можно отнести си-

стемы военного назначения (системы управления военными объектами, связи и разведки; мониторинг вооружения и боеприпасов дружественных сил; наблюдение за боем; ориентация на местности; обнаружение ядерных, биологических и химических атак), исследование окружающей среды (отслеживание движения птиц, животных и насекомых; мониторинг состояния окружающей среды с целью контроля качества сельскохозяйственной продукции; обнаружение химического или биологического загрязнения; обнаружение лесных пожаров; метеорологические или геофизические исследования; предсказание наводнений), здравоохранение (устройства для инвалидов; мониторинг пациентов; оперативная диагностика; мониторинг использования медикаментов в больницах), в перспективе исследование космического пространства, химическая обработка и ликвидация последствий стихийных бедствий с использованием робототехнических систем.

Использование БСС в указанных жизненно важных системах делает данные сети привлекательным объектом для атак. В подобных разрушающих воздействиях могут быть потенциально заинтересованы спецслужбы враждующих государств. Важные системы, которые у всех на виду, могут атаковаться и ради развлечения или сомнительной известности. Причиной атаки может быть недобросовестная конкуренция. Не секрет, что существует спрос на услуги проведения атак на системы конкурентов с целью их дискредитации. Также причиной атак может быть вымогательство. В СМИ упоминались случаи, когда злоумышленники блокировали работу информационных систем и потом вымогали деньги за разблокировку. Некоторые пользователи соглашались на условия злоумышленников, объясняя это тем, что перебои в работе приводят к еще большим убыткам. Подобный сценарий не исключен и применительно к приложениям БСС. Поэтому необходимо знать уязвимости БСС и предпринимать меры по их устранению.

Важнейшим фактором, влияющим на безопасность БСС, является ограниченная емкость батарей, устанавливаемых в узлах (мотах). Следует учитывать, что заменить батареи чаще всего невозможно, рациональное энергопитание является серьезной проблемой. Исследования показали, что каждый бит, передаваемый в БСС, потребляет примерно столько же энергии, как исполнение от 800 до 1000 инструкций [4]. Таким образом, связь является самым энергоемким процессом. В связи с этим необходимо минимизировать число циклов приема и передачи данных. Любое расширение сообщений для обеспечения механизмов безопасности приводит к значительным затратам. Кроме того, обеспечение более высокого уровня безопасности в БСС, например, за счет криптографии, потребует большего потребления энергии [5]. Отсюда также следуют ограниченные вычислительные возможности данных устройств.

Узел БСС имеет небольшое количество памяти и дискового пространства. Память узла, как правило, включает в себя флэш-память и RAM. Флэш-память используется для хранения загруженного кода приложения, а RAM используется для хранения прикладных программ, данных датчика и промежуточных результатов вычислений. Для запуска сложных алгоритмов не хватает места уже после загрузки ОС [4]. Так, датчик TelosB имеет 16-битный RISC-процессор (8 МГц), только 10К RAM памяти, 48К программной памяти и 1024К внешней флэш-памяти.

Из-за перечисленных ограничений трудно непосредственно использовать обычные механизмы безопасности [6], поэтому БСС весьма уязвимы для различного рода вредоносных воздействий. Кроме того, часто узлы в БСС развернуты в удаленных регионах и оставлены без присмотра. Следовательно, вероятность того, что датчик будет под воздействием какой-либо атаки, в такой среде очень высока. Дистанционное управление БСС делает

практически невозможным превентивное обнаружение физического воздействия, что еще больше усложняет задачу обеспечения безопасности БСС.

Опубликовано множество статей по проблеме безопасности БСС. Обзор основных атак, таких как глушение, ручное вмешательство, создание коллизии, разрядка, манипулирование маршрутной информацией, выборочная передача пакетов, „Sybil attack“, „Blackhole attack“, „Wormhole attack“, „Hello Flood“, флудинг, „Clone attack“ приведен, например, в работах [7, 8].

В исследованиях, связанных с безопасностью БСС, предложены различные схемы обеспечения безопасности, которые учитывают ресурсные ограничения этих сетей. В данных исследованиях были предложены несколько безопасных протоколов маршрутизации [9], защищенные протоколы агрегации данных [10] и др. В дополнение к традиционным способам обеспечения безопасности, таким как безопасный маршрут и безопасная агрегация данных, механизмы безопасности, реализованные в БСС, также должны включать безопасное сотрудничество между узлами из-за децентрализованного характера сетей и отсутствия надежной инфраструктуры. В реальных БСС узлы не могут считаться надежными априори. Поэтому в настоящее время популярны исследования, ориентированные на построение доверительной модели датчиков, чтобы решить проблемы, которые выходят за рамки возможностей традиционных криптографических механизмов [11–18]. Так как в большинстве случаев узлы датчика находятся без присмотра, уязвимость к физической атаке является также важным вопросом в БСС. Существует несколько предложений в литературе для защиты от физического нападения на сенсорные узлы [19], для защиты от атак, результирующее действие которых подобно физическому воздействию [20].

Математические методы позволяют найти эффективное решение ряда узких задач безопасности БСС, однако в общем случае они, как правило, непригодны для оптимизации БСС, т.к. требуют существенных допущений. Приходится использовать имитационное моделирование для сравнения альтернативных стратегий организации отказоустойчивых БСС.

2. Моделирование беспроводных сенсорных сетей.

2.1. *Модели топологий беспроводных сенсорных сетей.* Модель беспроводной сенсорной сети можно представить как конечный набор сенсорных узлов $N = \{N_1, \dots, N_n\}$, где $|N| = n$. Сеть также включает в себя базовую станцию или „сток“ в дополнение к сенсорным узлам. Каждый из n узлов сети состоит из датчиков с дополнительными возможностями, средствами управления и контроля решения задач сети, к которым можно отнести задачи создания и управления кластером, создание точек агрегации данных.

Пораженные узлы определяются как набор узлов $T = \{T_1, \dots, T_p\}$, где $T \subset N$, такой, что каждый целевой узел r от множества T является критическим узлом сети и $|T| = r \ll n$. Воздействие определяется как набор вредоносных узлов в сети (источники атаки), которые обозначены как $A = \{A_0, \dots, A_{k-1}\}$, где $|A| = k \leq n$.

Отсюда видно, что в качестве модели структуры БСС удобно использовать граф с дополнительными свойствами вершин и ребер, которые можно изменять в процессе выполнения имитационной модели.

Узлы типичной беспроводной сенсорной сети работают с целью мониторинга и обнаружения представляющего интерес события в области их покрытия для последующей доставки данных наблюдений к централизованной базовой станции. Данные могут быть доставлены к базовой станции непосредственно сенсорными узлами, либо через цепочку определенных промежуточных узлов.

Модель доставки сообщений формирует определенный класс топологий. Например, к наиболее распространенным классам организации сенсорных сетей относятся модели: плоская, кластерная и агрегация данных.

В плоской топологии каждый узел в сети непосредственно посылает свои показания датчика на базовую станцию, используя один „хоп“, без промежуточных узлов передачи сообщений. В таких сетях каждый узел имеет равный приоритет. Предполагается, что для успешной работы плоской топологии все узлы должны иметь достаточно большой радиус передачи, чтобы облегчить их общение с централизованной базовой станцией.

В кластерной топологии набор сенсорных узлов с дополнительными возможностями определяется как множество „руководители“. Эти узлы выступают в качестве контроллера и администратора для некоторого набора (кластера) предопределенных узлов в сети. Руководители несут ответственность за управление их кластерами, агрегацию данных от узлов кластера, и пересылку данных на базовую станцию. Кроме того, руководители отвечают также за мониторинг состояния узлов в их кластерах, информирование об ошибках и потерях базовой станции. Руководители обычно используют маршрут в два „хопа“. Для формализации потока данных требуются два объекта: поток от узла до руководителя и поток от руководителя к базовой станции.

Некоторые специальные кластеры используют несколько „хопов“ для передачи данных между руководителями до переадресации на базовую станцию. Их классифицируют как сетевые топологии агрегирования данных.

В топологии типа агрегации данных показания отдельных узлов проходят через сеть от исходного узла к базовой станции с помощью определенного дерева взаимосвязанных узлов. Данные по пути агрегируются на конкретных узлах в сети, которые называются точками агрегации и имеют множество входящих ребер в узлы, число которых обычно превышает исходящие. Цель данной модели топологии — уменьшение общего потока трафика в сети и минимизация потребления энергии, связанного с частыми операциями передачи данных на базовую станцию по отдельным узлам сети. Типичная топология агрегации данных состоит из соединенных между собой деревьев, определяющих поток сетевого трафика от отдельного узла к базовой станции. Поток трафика может быть формализован как путь $\{f, f_2, \dots, f_{L(f)}\}$, где $L(f)$ представляет собой длину пути от узла f к базовой станции.

При моделировании топологий основным вопросом является связанность датчиков, т. е. сеть должна быть связной. Учитывая множество узлов, размещенных в пространстве, мы должны указать, какие узлы могут принимать передачу от другого узла, а какие нет. Если узел u находится в области передачи узла v , мы говорим, что u смежный к v или, что равносильно выражению, u является соседом v . Это соотношение, как правило, симметрично (или неориентированное); то есть, если узел u может передавать данные узлу v , также v может передавать данные узлу u . Топология сети описывается с помощью графа $G = (V, E)$, где V (вершины) — набор сенсоров (матов, узлов) и множество E (ребра, каналы) описывает отношения смежности между узлами. То есть для двух узлов $u, v \in V$, $(u, v) \in E$, если v смежная к u . В неориентированных графах если $u, v \in V$, $(u, v) \in E$, то и $(v, u) \in E$.

Проверку на существование ребра между вершинами графа, т. е. прямого канала между двумя сенсорами, делают с помощью моделей, аналогичных моделям чувствительности сенсоров, описанных в [21]. Например, часто используют бинарную модель

$$\begin{cases} (u,v) \in E, d(u,v) \leq R, \\ (u,v) \notin E, d(u,v) > R. \end{cases}$$

Здесь $d(u,v)$ — Евклидово расстояние, R — радиус передачи сенсора.

Важной составляющей процесса моделирования БСС является случайный выбор координат узлов сети. Для тестирования предлагаемых решений для БСС с помощью имитационного моделирования практически во всех публикациях используют плоскую квадратную поверхность и равномерное распределение датчиков в квадрате. Таким образом, возникает очень важная проблема выбора качественного генератора псевдослучайных чисел с длинным периодом.

2.2. *Генерация псевдослучайных чисел.* В основе любого генератора псевдослучайных чисел лежит базовый генератор, т. е. генератор непрерывной равномерно распределенной случайной величины на интервале $[0, 1]$. С целью повышения эффективности разрабатываемых средств анализа отказоустойчивости БСС было рассмотрено несколько способов генерации. Наиболее популярными генераторами случайных чисел являются широко известные линейный конгруэнтный метод и мультипликативный метод.

Желаемая последовательность случайных чисел $\{X_n\}$ может быть получена исходя из некоторого начального значения с помощью следующего отношения:

$$X_{n+1} = (aX_n + c) \bmod m, \quad n \geq 0.$$

Эта последовательность называется линейной конгруэнтной последовательностью. Для выбора констант генератора a , c и m необходимо использовать определенные правила [22].

Конгруэнтная последовательность всегда образует петли, т. е. обязательно существует цикл, повторяющийся бесконечное число раз. Это свойство является общим для всех последовательностей вида $X_{n+1} = f(X_n)$, где f преобразует конечное множество само в себя. Повторяющиеся циклы называются периодами. Последовательности, которые необходимо использовать для моделирования БСС, требуют относительно длинного периода.

Данный генератор входит в большинство стандартных библиотек компиляторов. Для разрабатываемой системы рассматривались две функции, одна из них функция `rand()` из BSD LIBC, другая `rand()` из Microsoft C Runtime (MSCVRT.DLL). Выяснилось, что генератор обладает не таким уж большим периодом и для исследования больших сетей не подойдет.

Для реализации проекта выбрана платформа .NET Framework 4.5, где существует класс `Random`, который является генератором псевдослучайных чисел. Текущая реализация класса `Random` основана на модифицированном алгоритме Дж. Ж. Митчела и Д. Ф. Мура:

$$X_n = (X_{n-24} + X_{n-55}) \bmod m, \quad m \geq 55,$$

где m — четное число, а X_0, \dots, X_{54} — произвольные целые не все четные числа [22].

Данный алгоритм обладает весьма большим периодом, в расчетах не требуются умножение и деление, а, следовательно, он не требует больших вычислительных мощностей и его можно выбрать для моделирования.

Генерация псевдослучайных чисел начинается с некоторого начального числа. При повторном использовании того же начального числа создается та же самая последовательность чисел. Для получения различных последовательностей можно сделать выбор

начального числа зависимым от времени. Тогда для каждого нового экземпляра `Random` будут создаваться различные последовательности. По умолчанию в лишенном параметров конструкторе класса `Random` для получения начального значения используются системные часы, а параметризованный конструктор данного класса может принимать значение типа `Int32`, зависящее от количества тактов в текущем времени. Также имеется возможность реализации собственного генератора псевдослучайных чисел с помощью наследования класса `System.Random`.

Кроме организации генератора псевдослучайных чисел, для моделирования БСС необходимы генераторы псевдослучайных топологий сети. В большинстве исследований по беспроводным сенсорным сетям, когда оценивается эффективность того или иного алгоритма, в качестве топологии сети обычно используются UDG-графы [23]. При разработке программных средств, рассматриваемых в данной работе, были реализованы подходы, предлагаемые в указанной статье.

2.3. Модели атакующих воздействий. Для исследования последствий атак в самых общих предположениях применяют теорию случайных графов [24]. Атакующее воздействие определяет ненадежность элементов графа, отображающего топологию сети. В качестве метрики атак выступают теоретико-графовые характеристики, например, связность. Ввиду огромной сложности подобных вычислений (вычисление вероятности связности случайного графа является NP полной задачей) аналитические выкладки возможны только для графов малой размерности.

При разработке математических моделей атакующего воздействия на конкретный узел используются методы теории систем массового обслуживания [20]. При использовании предположений о простейшем потоке (процесс Пуассона) событий в системе можно получить аналитические результаты. Хотя в общем случае исследование отказоустойчивости БСС требует применения имитационного моделирования, указанные результаты могут быть реализованы в соответствующей системе с целью получения быстрого результата в частных случаях.

3. Программные средства анализа отказоустойчивости БСС. Практической реализацией генератора топологий является динамическая библиотека `UdgSharp`. Данная библиотека позволяет осуществлять генерацию UDG-графов по всем алгоритмам, указанным в статье [23]. `UdgSharp` имеет два класса — `Node` и `Udg`. Класс `Node` описывает структуру вершины графов.

```
public class Node
{
    public Node() {...}
    public Node(int id, int x, int y) {...}
    public int Id { get; set; }
    public int X { get; set; }
    public int Y { get; set; }
    protected bool Equals(Node other) {...}
    public override bool Equals(object obj) {...}
    public override int GetHashCode() {...}
    public override string ToString() {...}
}
```

Данный класс имеет следующие свойства:

- `int Id` — уникальный идентификатор вершины;
- `int X` — координата X на плоскости;
- `int Y` — координата Y на плоскости.

И методы:

- `Node()` — конструктор объекта по умолчанию, создает вершину с координатами $(0, 0)$ и нулевым идентификатором;
- `Node(int id, int x, int y)` — конструктор объекта, создает вершину с координатами (x, y) и идентификатором `id`;
- `bool Equals(Node other)` — метод сравнивает объект с объектом типа `Node`;
- `bool Equals(Object obj)` — метод сравнивает объект с объектом типа `Object`;
- `int GetHashCode()` — метод генерирует хэш-код объекта;
- `String ToString()` — метод возвращает объект в виде строки.

Класс `Udg` описывает структуру и реализует генерацию UDG-графов.

```
public class Udg
{
    public Udg(int nodesCount, int radius, Algorithm algorithm,
              int maxX, int maxY) {...}

    public int NodesCount { get; private set; }
    public int EdgesCount { get; private set; }
    public bool[,] AdjacencyMatrix { get; private set; }
    public List<Node> Nodes { get; set; }
    public int Radius { get; set; }
    public void Reinitialize() {...}
    protected bool Equals(Udg other) {...}
    public override bool Equals(object obj) {...}
    public override int GetHashCode() {...}
    public override string ToString() {...}
}
```

Данный класс имеет следующие свойства:

- `List<Node> Nodes` — список вершин графа;
- `bool[,] AdjacencyMatrix` — матрица смежности;
- `int NodesCount` — количество вершин графа;
- `int EdgesCount` — количество ребер графа;
- `int Radius` — радиус каждой вершины.

А также методы:

- `Udg(int nodesCount, int radius, Algorithm algorithm, int maxX, int maxY)` — конструктор объекта, создает UDG-граф с заданным количеством вершин `nodesCount` по заданному алгоритму `algorithm` и в заданной области $(0, \text{maxX})$, $(0, \text{maxY})$;
- `void Reinitialize()` — позволяет реорганизовать граф в соответствии с его новыми параметрами;
- `bool Equals(Udg other)` — метод сравнивает объект с объектом типа `Udg`;
- `bool Equals(Object obj)` — метод сравнивает объект с объектом типа `Object`;
- `int GetHashCode()` — метод генерирует хэш-код объекта;

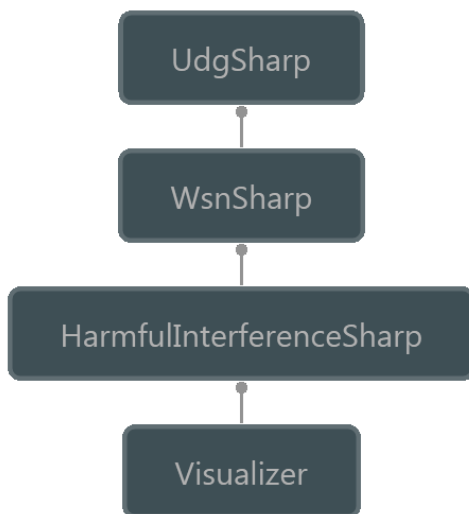


Рис. 1. Диаграмма зависимостей

— `String ToString()` — метод возвращает объект в виде строки.

За эмуляцию сети в данной работе отвечает библиотека `WsnSharp`. Классы библиотеки осуществляют такие задачи как маршрутизация, передача пакетов и прочее. `WsnSharp` содержит классы `Wsn` и `SensorNode`. `SensorNode` является дочерним классом `Node`. Класс `SensorNode` описывает структуру сенсорного узла. Класс `Wsn` описывает структуру и реализует эмуляции сети.

Эмуляцией вредоносных воздействий занимается еще одна библиотека — `HarmfulInterferenceSharp`, содержащая абстрактный класс `HarmfulInterference`, который является родителем для всех классов вредоносных воздействий. Наследуя данный класс и перегружая его методы, можно изменять сценарии атаки, что весьма удобно для программиста, т. к. нет необходимости менять код других библиотек и частей приложения.

Приложение „Визуализатор сети“ состоит из четырех файлов: исполняемый файл (`Visualizer.exe`), библиотека генерации UDG графов (`UdgSharp.dll`), библиотека эмуляции сети (`WsnSharp.dll`), библиотека эмуляции вредоносных воздействий (`HarmfulInterferenceSharp.dll`). Данное приложение позволяет генерировать и визуализировать графы с заданными настройками, создавать сеть из получившегося графа, добавляя к нему „сток“, а также накладывать на сеть какие-либо вредоносные воздействия.

Диаграмма зависимостей изображена на рис. 1.

Основное окно `Visualizer.exe` имеет вид, показанный на рис. 2.

Данное окно состоит из трех элементов — экрана вывода (в центре), панели управления (справа) и экрана лога (снизу). На экране вывода визуализируются сгенерированный граф, сеть, вредоносные воздействия, а также осуществляется взаимодействие с пользователем при добавлении вредоносных атак и „стока“ сети.

С помощью панели управления визуализатора происходит работа пользователя с приложением, данная панель состоит из трех групп: настройки графа (`graph settings`), настройки рисования (`draw settings`) и настройки вредоносных воздействий (`interference settings`).

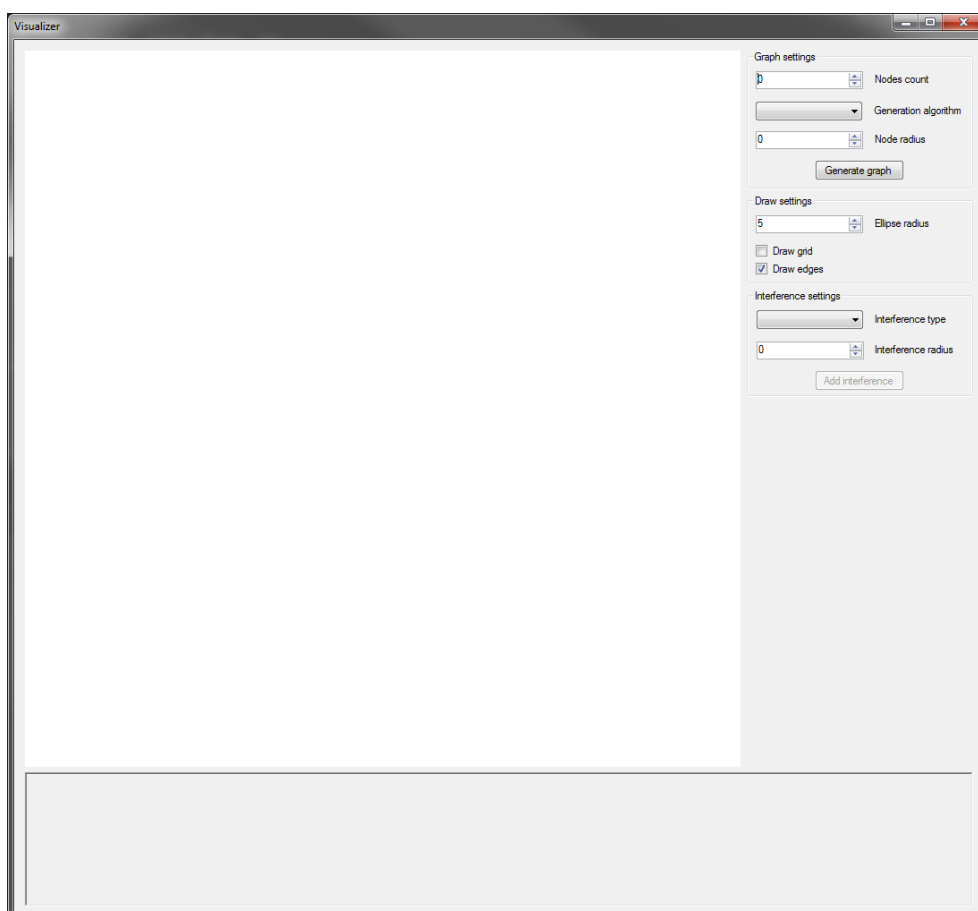


Рис. 2. Основной экран приложения

На экране лога выводятся различные события приложения с подробной информацией, также лог дублируется в файл, который можно посмотреть позже.

Для начала работы с основными функциями приложения необходимо сгенерировать граф, для этого необходимо на панели управления в группе Graph settings в поле ввода Nodes count ввести количество вершин графа, выбрать из выпадающего списка Generation algorithm один из трех алгоритмов генерации и в поле ввода Radius ввести радиус вершин.

В группе Draw settings можно выбрать настройки отображения, в поле ввода Ellipse radius вводится размер вершин, который будет отображаться на экране вывода, опция Draw grid определяет, будут ли на экране вывода отображаться сетка, опция Draw edges определяет, будут ли на экране вывода отображаться графы. После необходимо нажать на кнопку Generate graph, если все поля заполнены верно, то на экране вывода отобразится соответствующий граф, а на экране лога выведется соответствующая запись о его генерации.

Далее для создания сети необходимо создать „сток“, для этого на экране вывода необходимо нажать левой кнопкой мышки в область, где должен находиться данный узел. При правильном выполнении действия сгенерируется сеть и приведется первичная маршрутизация, в логе появится запись о создании сети, а на экране вывода красным цветом отобразится „сток“.

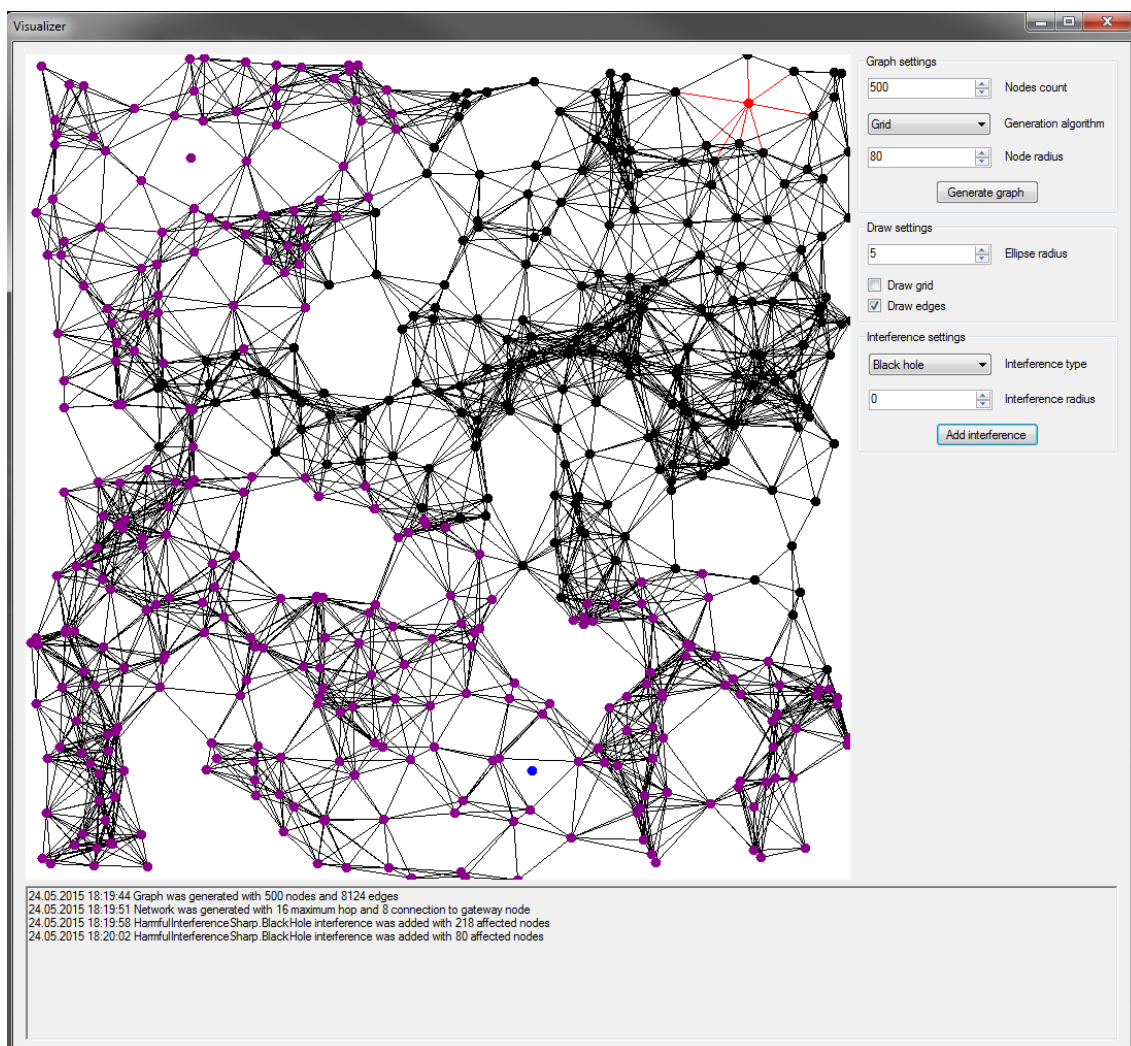


Рис. 3. Результат работы программы

Теперь на данную сеть можно накладывать различные вредоносные воздействия. Для этого в панели управления в группе *Interference settings*, необходимо выбрать в выпадающем списке *Interference type* соответствующее воздействие и ввести радиус воздействия в поле *Radius*. Далее необходимо нажать на кнопку *Add interference* и после нажать левой кнопкой мыши на экран вывода; если все выполнено правильно, то создается воздействие, произойдет обработка его сценария, также оно отобразится на экране вывода синим; все вершины, которые будут подвержены ему, подсвечиваются фиолетовым.

На рис. 3 показано окно программы, где на сеть наложено сразу несколько атак.

Разработанный продукт применялся для исследования, в частности, атаки *Black Hole*. Использовались 100 сгенерированных сетей с параметрами: 1000 узлов, радиус 50, область 800x800 с целью найти оптимальное число „стоков“, а также влияние количества вредоносных узлов на среднее количество вовлеченных в атаку узлов. Стоки размещали, используя распространенную практику размещения такого узла, а именно — в один из углов сети, т. е. с координатами (0; 0), (0; 800), (800; 0), (800; 800). Далее постепенно отодвигали источник атаки, при этом подсчитывалось количество возможных вовлеченных

узлов. Очевидно, наиболее опасной ситуацией для сети является нахождение взломанного узла в непосредственной близости к „стоку“, в худшем случае на линии радиуса приема. Результаты моделирования это показали. Далее был добавлен второй сток. Количество вовлеченных узлов уменьшилось почти в два раза, как в случае расположения источника атаки вблизи „стока“, так и в случае нахождения вредоносного узла в середине области покрытия. Однако чем дальше мы от одного „стока“, тем ближе мы к другому такому же узлу, а, следовательно, при смещении атакующего узла относительно середины расстояния между стоками будут наблюдаться негативная тенденция и увеличение количества вовлеченных узлов.

Результаты имитационного моделирования также показали, что при дальнейшем добавлении стоков на границы области покрытия количество вовлеченных узлов уменьшается незначительно, а, значит, добавление относительно дорогих базовых станций не будет иметь экономической выгоды. Таким образом, в данном сценарии противодействия атаке Black Hole рекомендуемое количество стоков равно двум.

Заключение. В рамках проекта по разработке средств анализа отказоустойчивости БСС проанализированы наиболее опасные для данных сетей атаки: глушение, ручное вмешательство, создание коллизии, разрядка, манипулирование маршрутной информацией, выборочная передача пакетов, „Sybil attack“, „Blackhole attack“, „Wormhole attack“, „Hello Flood“, флудинг, „Clone attack“ и проч. Также проанализированы математические методы моделирования атак в БСС. Сделан вывод о том, что они пригодны для анализа некоторых частных атак. Но в общем случае требуется инструментарий на основе имитационного моделирования. Разработан программный продукт, предназначенный для решения задачи анализа отказоустойчивости беспроводных сенсорных сетей и обладающий возможностью визуализации результатов. Продукт содержит метод включения в анализ новых атак. Применение программного продукта для исследования некоторых атак позволило получить новые результаты, разработать рекомендации для защиты БСС.

Список литературы

1. ATZORI L., IERA A., MORABITO G. The internet of things: A survey // *Computer Networks*. 2010. N 54 (15). P. 2787–2805.
2. PASCHOU M., SAKKOPOULOS E., SOURLA E. Health Internet of Things: metrics and methods for efficient data transfer. *Simulation Modelling Practice and Theory*, 2012.
3. ESTRIN D., GOVINDAN R., HEIDEMANN J., KUMAR S. Next century challenges: scalable coordination in sensor networks / *Proc. of the ACM MobiCom'99*. 1999, USA: Washington. P. 263–270.
4. HILL J., SZEWCZYK R., WOO A., HOLLAR S., CULLER D. E., PISTER K. System architecture directions for networked sensors / *Proc. of the 9th International Conf. on Architectural Support for Programming Languages and Operating Systems*. 2000. N. Y.: ACM Press. P. 93–104.
5. YUAN L., QU G. Design space exploration for energy-efficient secure sensor networks / *Proc. of IEEE International Conference on Application-Specific Systems, Architectures, and Processors*. July 2002. P. 88–100.
6. PERRIG A., SZEWCZYK R., WEN V., CULLER D. E., TYGAR J. D. SPINS: Security protocols for sensor networks // *Wireless Networks*. 2002. V.8. N 5. P. 521–534.
7. ШАХОВ В. В., СТРЕЛЬНИКОВ В. Е., НГУЕН В. Д. К вопросу об эффективности беспроводных сенсорных сетей // *Проблемы информатики*. 2014. № 2. С. 28–38.
8. WOOD A. D., STANKOVIC J. A. Denial of service in sensor networks // *IEEE Computer*. 2002. V. 35. N 10. P. 54–62.

9. DENG J., HAN R., MISHRA S. INSENS: Intrusion-tolerant routing in wireless sensor networks / Technical Report CU-CS-939-02. Department of Computer Science, University of Colorado at Boulder. Nov. 2002.
10. KUMAR V., MADRIA S. Secure Hierarchical Data Aggregation in Wireless Sensor Networks: Performance Evaluation and Analysis / IEEE 13th International Conf. on Mobile Data Management MDM. 2012. P. 196–201.
11. LI X., LI Z., STOJMENOVIC M., NARASIMHAN V., NAYAK A. Autoregressive Trust Management in Wireless Ad Hoc Networks // Ad Hoc & Sensor Wireless Networks. 2012. N 16 (1–3). P. 229–242.
12. KIMAND T., SEO H. A Trust Model using Fuzzy Logic in Wireless Sensor Network / Proc. of World Academy of Science, Engineering and Technology. 2008. P. 63–66.
13. FENG R., XU X., ZHOU X. AND WAN J. A Trust Evaluation Algorithm for Wireless Sensor Networks Based on Node Behaviors and D-S Evidence Theory Sensors. 2011. P.1345–1360.
14. CHEN H., WU H., ZHOU X. AND GAO C. Agent-based Trust Model in Wireless Sensor Networks / 8th ACIS International Conf. on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing. 2007. P. 119–124.
15. CHEN H., GU G., WU H. AND GAO C. Reputation and Trust Mathematical Approach for Wireless Sensor Networks // International J. of Multimedia and Ubiquitous Engineering. 2007. N 2 (3). P. 23–32.
16. CHEN H. Task-based Trust Management for Wireless Sensor Networks // International J. of Security and Its Applications. 2009. N 3 (2). P. 21–26.
17. ZHOU X., TANG L. Design and Evaluation of Black list aided False Data Filtering Scheme for Wireless Sensor Networks. Master degree thesis, Peking University, 2007.
18. HUR J., LEE Y., YOON H., CHOI D. AND JIN S. Trust Evaluation Model for Wireless Sensor Networks / The 7th International Conf. on Advanced Communication Technology. 2005. P. 491–496.
19. WANG X., GU W., CHELLAPPAN S., SCHOSECK K., XUAN D. Lifetime optimization of sensor networks under physical attacks / Proc. of IEEE International Conf. on Communications. May 2005.
20. ШАКHOV V. V. Protecting Wireless Sensor Networks from Energy Exhausting Attacks // Springer LNCS. 2013. P. 184–193.
21. ШАКHOV V. V. Experiment Design for Parameter Estimation in Sensing Models // Springer LNCS. 2013. V. 8072. P. 151–158.
22. КНУТ Д. Э. Искусство программирования. Т. 2. Получисленные алгоритмы. М.: „Вильямс“, 2007.
23. ШАКHOV V. V., SOKOLOVA O., YURGENSON N. A Fast Method for Network Topology Generating. Springer LNCS. 2014. V. 8715. P. 96–101.
24. МИГОВ Д. А., ШАКHOV V. V. Reliability of Ad Hoc Networks with Imperfect Nodes. Springer LNCS. 2014. V. 8715. P. 49–58.

*Стрельников Василий Евгеньевич — магистрант
Новосибирского государственного
технического университета,
e-mail: kafedra_vt@vt.cs.nstu.ru*

Дата поступления — 01.11.2015