Теоретические основы информатики

Theoretical foundations of computer science

УДК 004.942

ПРИМЕНЕНИЕ ФОРМАЛЬНЫХ МЕТОДОВ ПРИ ПРОЕКТИРОВАНИИ КОЛЛАБОРАЦИОННОЙ СИСТЕМЫ ПРОТИВОВИРУСНОЙ ЗАЩИТЫ

Р. Е. ШАРЫКИН[1)]***, А. Н. КУРБАЦКИЙ***[1)]

[1)]*Белорусский государственный университет, пр. Независимости, 4, 220030, г. Минск, Беларусь*

Предлагается подход, позволяющий исследовать математическую модель системы защиты от вирусов на этапе ее проектирования с помощью статистического анализа исполняемой спецификации модели, основанной на формализме распределенных объектно ориентированных стохастических гибридных систем (РООСГС). Важными аспектами модели являются ее распределенный и вероятностный характер. Эти аспекты делают модель более сложной для проведения атак, но в то же время значительно усложняют понимание ее свойств разработчиком. На данном примере мы показываем, как с помощью использования спецификации системы в качестве модели РООСГС вкупе со статистическим анализом можно исследовать ее свойства на раннем этапе проектирования и как с применением данного подхода можно обнаружить «дефекты» модели и исправить их еще в процессе ее создания.

Ключевые слова: математическое моделирование; гибридные системы; стохастические системы; спецификация моделей; совместная антивирусная защита.

Образец цитирования:
Шарыкин РЕ, Курбацкий АН. Применение формальных методов при проектировании коллаборационной системы противовирусной защиты. *Журнал Белорусского государственного университета. Математика. Информатика.* 2020;1:59–69.
https://doi.org/10.33581/2520-6508-2020-1-59-69

For citation:
Sharykin RE, Kourbatski AN. Application of formal methods in the design of a collaborative virus defense system. *Journal of the Belarusian State University. Mathematics and Informatics.* 2020;1:59–69. Russian.
https://doi.org/10.33581/2520-6508-2020-1-59-69

Авторы:
Роман Евгеньевич Шарыкин – соискатель кафедры технологий программирования факультета прикладной математики и информатики. Научный руководитель – А. Н. Курбацкий.
Александр Николаевич Курбацкий – доктор технических наук, профессор; заведующий кафедрой технологий программирования факультета прикладной математики и информатики.

Authors:
Raman E. Sharykin, competitor at the department of software engineering, faculty of applied mathematics and computer science.
sharykin@gmail.com
Alexander N. Kourbatski, doctor of science (engineering), full professor; head of the department of software engineering, faculty of applied mathematics and computer science.
kurb@unibel.by

Журнал Белорусского государственного университета. Математика. Информатика. 2020;1:59–69
Journal of the Belarusian State University. Mathematics and Informatics. 2020;1:59–69

APPLICATION OF FORMAL METHODS IN THE DESIGN OF A COLLABORATIVE VIRUS DEFENSE SYSTEM

***R. E. SHARYKIN*[a], *A. N. KOURBATSKI*[a]**

[a]*Belarusian State University, 4 Niezaliežnasci Avenue, Minsk 220030, Belarus*
Corresponding author: R. E. Sharykin (sharykin@gmail.com)

This article proposes an approach that allows us to study a mathematical model of a virus protection system at the stage of its design using statistical analysis of an executable model specification based on the formalism of Distributed Object Based Stochastic Hybrid Systems (DOBSHS). Important aspects of the model are its distributed and probabilistic nature. These aspects make the model more difficult to carry out attacks, but at the same time significantly complicate the understanding of its properties by the developer. In this example, we show how, using the specification of the system as a DOBSHS model, coupled with its statistical analysis, we can investigate its properties at an early stage of design and how, using this approach, we can detect «defects» of the model and correct them during the process of creating the model.

Keywords: mathematical modeling; hybrid systems; stochastic systems; model specification; collaborative virus defense.

Введение

Ввиду растущей важности и сложности распределенных систем вирусной защиты использование формальных методов для понимания динамики таких систем на ранних этапах их разработки представляется важной задачей. Одной из трудностей применения формальных методов в данной области является определение подходящей формальной модели, такой как сети Петри или дискретная марковская цепь. После того как выбор формальной модели был сделан, может быть очень затратным ее изменение впоследствии в случае, если обнаружено, что другая формальная модель лучше подходит для описания рассматриваемой системы.

Цель настоящей статьи – изложение подхода к разработке модели сложных динамических распределенных стохастических систем с коммуникацией на ранних этапах до проведения ее аналитического анализа и его иллюстрация на примере предварительной модели стохастической групповой системы защиты от вирусов и экспериментов с ней.

Мы представляем подход к применению формальных методов на ранних этапах дизайна системы защиты от вирусов, основанный на использовании формальной модели распределенных объектно ориентированных стохастических гибридных систем (РООСГС) [1], и подход к ее верификации, изложенный в [2]. Данная модель базируется на переписывающей логике [3]. Использование таковой позволяет применять формальные методы в начале проектирования, в то же время не ограничивая разработчика определенным формализмом. Показано, что различные математические модели могут быть естественным образом представлены с помощью переписывающей логики [3; 4], а основанный на ней формализм может быть сжатым, интуитивно понятным и хорошо подходит для спецификации распределенных параллельных систем с асинхронной коммуникацией. Спецификации, выполненные в переписывающей логике, могут выполняться с помощью системы Maude [5], которая позволяет симулировать и улучшать спецификацию системы защиты для различных сценариев атак в самом начале разработки системы защиты. Как только разработчик остается удовлетворенным полученной спецификацией, он может выбрать наиболее подходящую математическую модель с известным представлением в переписывающей логике для формального доказательства критически важных свойств системы либо нахождения более сложных недостатков, которые невозможно выявить с помощью техники симуляций. Maude также представляет средства, облегчающие проведение доказательств в формализме переписывающей логики [6].

После того как формализм выбран, важным вопросом является анализ системы на ранних этапах ее разработки. Желаемый метод должен быть «легким» в том смысле, что он может давать возможность разработчику концентрироваться на алгоритме защиты и его параметрах. Мы предлагаем способ, который предоставляет возможность получать быструю оценку основных поведенческих свойств системы и испробовать модель защиты на различных предположениях относительно окружения, позволяя потратить усилия для проведения более сложных формальных процедур валидации на поздних этапах, когда пространство возможностей для дизайна уже сокращено.

Для демонстрации нашей методологии рассматривается частный случай вероятностного алгоритма защиты от вирусов. В примере системы защиты от вирусных атак предлагается подход [2], основанный на

симуляциях модели и использующий язык QuaTEx [7] для задания свойств, подлежащих изучению. Данный подход эффективен для количественного исследования стохастических систем, таких как протоколы защиты от нежелательного проникновения при распределенных атаках. Представлен формальный подход к анализу поведенческих свойств системы, распределенной по узлам в контексте защиты от вирусных атак, и иллюстрируются типы свойств систем защиты, которые применимы как к нашей, так и к другим системам и для которых формальный анализ представляется важным аналитическим методом.

Пример коллаборационной стохастической системы защиты от вирусных атак

Рассматриваемый пример системы защиты от вирусных атак базируется на коллаборационной системе, не являющейся стохастической, которая описана в [8], где детально представлен изначальный алгоритм, обсуждается его эффективность и т. п.

В случае совместной защиты от вирусных атак локальные сети сотрудничают, объединяясь в группы, в которых исходящие маршрутизаторы информируют своих партнеров посредством локальных оповещений о возможной вирусной активности. Если узел получает подобные коллаборационные извещения от N или более источников, он переходит в защитный фильтрующий режим. Алгоритм имеет схожие черты с алгоритмами одноранговых оповещений, представленными в [9; 10], однако в нашем случае распространением оповещений управляет система ограничения скорости соединений, аналогичная описанной в [11]. Эта система распространяет оповещение, когда обнаруживает, что некоторый внутренний узел создает соединения с частотой, превышающей некоторое пороговое значение. Образованные с превышением данного порога соединения с новыми узлами сбрасываются исходящим маршрутизатором до следующего временно́го периода. Одним из достоинств данной комбинированной стратегии защиты является то, что системы ограничения скорости, распространяющие оповещения, эффективно снижают скорость распространения вируса, позволяя стратегии коллаборации узлов начать действовать, в то время как алгоритмы, основанные на простом обмене оповещениями между узлами, уязвимы для быстро распространяющихся вирусов.

При получении достаточного количества коллаборационных оповещений о вирусной атаке исходящий маршрутизатор имеет возможность переходить в режим защиты, в котором он сбрасывает все пакеты, которые, как предполагается, имеют отношение к большинству оповещений. Для создания достаточно абстрактной модели мы не уточняем, как именно организуется фильтрация, а полагаем, что алгоритм содержит фильтрацию и узлы могут находиться в данном режиме. Существуют основанные на генерации сигнатур исследования по разработке автоматических систем, таких как EarlyBird [12] и Autograph [13], которые могут быть использованы для реализации фильтрации в рассматриваемой нами системе.

Размер групп оповещения существенно влияет на свойства изучаемой модели. Его увеличение приводит к большему подавлению распространения вируса. Однако с ростом размера групп также повышается «цена» обработки оповещений, отправленных в результате ложноположительного обнаружения вируса (более подробное исследование влияния размера групп см. в [10]).

Мы вводим в систему защиты из [8] вероятности обнаружения вируса и ложноположительного обнаружения вируса. Также фазы системы мы заменяем на процессы, которые протекают асинхронно с интервалами, имеющими экспоненциальное распределение. Это делает систему более реалистичной, но в то же время стохастической по своей природе. Основным же дополнением к указанной системе является формирование групп оповещения в соответствии со случайным распределением, в отличие от фиксированных групп в [8]. Структура групп в [8] позволяла провести атаку с полным заражением системы, основываясь на алгоритме формирования групп [14]. При их случайном формировании атаки подобного типа становятся невозможными, так как идея построения атаки критически зависит от способности атакующего предсказать точный состав групп оповещения.

Мы моделируем локальную сеть как граф локальных узлов с одним исходящим маршрутизатором. Несколько локальных сетей объединяются друг с другом посредством их исходящих маршрутизаторов. В целом система групповой защиты моделируется как параллельное, асинхронное развертывание алгоритма защиты локальной сети, установленного на всех исходящих маршрутизаторах. В каждом экземпляре системы защиты локальной сети независимо от других выполняются три процесса (с интервалами запуска, распределенными экспоненциально): детектирование инфекции, формирование группы оповещения, оповещение. Также на всех временны́х интервалах происходит обновление состояния защиты для каждого экземпляра системы.

При выполнении детектирования исходящий маршрутизатор может наблюдать локальное превышение заданных порогов на скорость установки соединений в некоторых узлах, генерируя локальное оповещение для каждого превышения.

Для формирования групп оповещения исходящий маршрутизатор создает группы размера G из множества участвующих локальных сетей в соответствии с определенным вероятностным распределением.

При выполнении оповещения исходящий маршрутизатор направляет оповещение в свою группу оповещения в случае, если он сгенерировал как минимум одно локальное оповещение к данному моменту. Исходящий маршрутизатор также получает оповещения от других локальных сетей и увеличивает уровень локальной метрики оповещений alert, основываясь на количестве полученных локальных оповещений и оповещений от других сетей. Каждое оповещение увеличивает значение alert с учетом параметра severity, который задает критичность оповещений. Параметр alert уменьшается со временем с заранее данной скоростью. Когда alert превышает пороговое значение alpha, исходящий маршрутизатор применяет фильтрацию ко всем входящим пакетам, которые подпадают под критерий фильтрации. Параметр alpha вычисляется как severity * r, где r обозначает меру коллаборации, необходимую для включения фильтров. Данное фильтрующее состояние сохраняется до тех пор, пока параметр alert не достигнет нулевого значения вследствие его уменьшения с течением времени.

Спецификация системы защиты в переписывающей логике

В данном разделе представим спецификацию с помощью переписывающей логики вероятностной групповой системы защиты от компьютерных вирусов, описанной в разделе «Пример коллаборационной стохастической системы защиты от вирусных атак». Для моделирования системы защиты мы используем модель спецификации SHYMaude [2]. Опишем ключевые элементы SHYMaude, имеющие отношение к данному моделированию.

Распределенная конфигурация системы моделируется как множество объектов и сообщений, действующих параллельно в соответствии с множеством правил перезаписи, описывающих поведение отдельных элементов. Объекты в SHYMaude представляются с помощью следующего синтаксиса:

```
< id : C | attr_1: value_1, …, attr_N: value_N >
```

Здесь id – идентификатор объекта; C – его класс; attr_1, …, attr_N – имена атрибутов объекта; value_1, …, value_N – значения атрибутов. Пример объекта, представляющего собой узел сети (к примеру, компьютер в сети), выглядит таким образом:

```
< 7 : Node | infected: true, alerts: 5, alert: 2 >
```

Данный узел является инфицированным и имеет уровень alert, равный 2.

Активное сообщение для объекта с идентификатором id выглядит так:

```
< id <- message >
```

Здесь message состоит из множества атрибутов, один из которых может быть выделенным и задавать «команду» для получающего узла (необязательный в случае, если значение сообщения уникально в его отсутствии). Пример сообщения, передающего группу оповещения узлу 7:

```
< 7 <- group : [1, 4, 7] >
```

В этом сообщении [1, 4, 7] – множество идентификаторов объектов, составляющих группу оповещения.

Запланированное сообщение задается в виде

```
[time, msg]
```

Здесь time – оставшееся время до активации сообщения; msg – запланированное сообщение.

Полная спецификация системы с помощью языка SHYMaude может быть найдена в [15]. Для наглядности приведем некоторые переписывающие правила.

Динамика инфицирования системы вирусом описывается следующими переписывающими правилами.

1. Распространение вируса узлом, не находящимся в состоянии фильтрации:

```
rl < id : Node | infected : true, filtering : false, attrSet >
   < id <- propagate >
   =>
   < id : Node | infected : true, filtering : false, attrSet >
   < (id + 1 + sampleNat(N - 1)) rem N <- infect >
   [D, < id <- propagate >] with probability D := Exponential(propagateRate).
```

При получении сообщения propagate узел производит попытку инфицирования случайно выбранного узла и планирует следующую попытку через случайный интервал времени, выбранный в соответствии

с экспоненциальным распределением, определяемым параметром propagateRate. Атрибут attrSet является переменной типа AttributeSet, которая для данного переписывающего правила может быть равна любому набору пар атрибутов и их значений объекта.

2. Получение команды propagate узлом в состоянии фильтрации:

```
rl < id : Node | infected : true, filtering : true, attrSet >
   < id <- propagate >
   =>
   < id : Node | infected : true, filtering : true, attrSet >
   [D, < id <- propagate >] with probability D := Exponential(propagateRate).
```

Указанный узел не может заражать другие узлы и лишь планирует следующую попытку.

3. Получение команды инфицирования (попытки инфицирования данного узла):

```
rl < id : Node | infected : false, filtering : false, attrSet >
   < id <- infect > =>
   < id : Node | infected : true, filtering : false, attrSet >
   [D, < id <- propagate >] with probability D := Exponential(propagateRate).
```

Не находящийся в состоянии фильтрации узел, получивший свидетельствующее о попытке инфицирования сообщение infect, инфицируется и планирует попытку заражения.

4. Получение команды инфицирования инфицированным узлом:

```
rl < id : Node | infected : true, filtering : false, attrSet > < id <- infect >
   =>
   < id : Node | infected : true, filtering : false, attrSet >.
```

Если узел уже инфицирован, то попытка его инфицирования не изменяет его состояния.

5. Получение команды инфицирования узлом, находящимся в состоянии фильтрации:

```
rl < id : Node | filtering : true, attrSet > < id <- infect >
   =>
   < id : Node | filtering : true, attrSet >.
```

Указанный узел не может быть инфицирован.

Динамика системы защиты описывается следующими правилами.

6. Детектирование вируса:

```
rl < id : Node | infected : infected, detected : detected, attrSet >
   < id <- detect >
   =>
   < id : Node | infected : infected, detected : if infected then D else F fi, attrSet >
with probability D := Bernoulli(detectionProb), F := Bernoulli(fpRate)
   [D, <id <- detect >] with probability D := Exponential(detectRate).
```

В этом правиле происходит детектирование вируса с заданной вероятностью, также учтена ложноположительная вероятность обнаружения вируса. Детектирование производится через случайные промежутки времени, распределенные экспоненциально.

7. Оповещение групп:

```
rl < id : Node | infected : infected, detected : detected, group : group, attrSet >
   < id <- alertGroup >
   =>
   if(detected) then sendGroupAlerts(group) else none fi
   < id : Node | infected : infected, detected : detected, group : group, attrSet >
   [D, < id <- alertGroup >] with probability D := Exponential(alertRate).
```

Оповещение производится только узлами, которые детектировали вирус.

8. Получение оповещения:

```
rl < id : Node | alerts : alerts, attrSet >
   < id <- alert >
   =>
   < id : Node | alerts : alerts + 1, attrSet >
```

При получении оповещения узел увеличивает счетчик оповещений на единицу.

Метрики оценки систем защиты

Оценка и сравнение большого количества подходов к защите от вирусов являются задачей современных исследований. В обзоре [16], анализирующем текущее состояние разработок в области формализации оценок и сравнительных критериев, отмечено, что оценка противовирусных стратегий часто основывается исключительно на том, как влияет защита на темп роста инфекции. Однако во многих случаях это лишь одна из метрик, и она не позволяет различать схемы защиты, похожие на конкретно выбранной метрике.

В [16] предлагаются дополнительные метрики, которые дают более глубокое понимание различия поведения систем защиты и предоставляют возможность сравнивать эти системы в измерениях, отличных от темпов роста инфекции. Такие метрики позволяют оценивать системы защиты, учитывая их особенности, например возможность своевременной интервенции человека или нежелательность нахождения большей части системы в режиме фильтрации, так как последняя может оказать серьезный эффект на производительность. В качестве простого примера метрики, иллюстрирующей вторую особенность, приведем процент незараженных узлов, находящихся в режиме без фильтрации после достижения верхнего предела заражения системы (далее будем говорить также, что система достигла насыщения). Эта метрика может быть очень важным аспектом при сравнении различных систем, так как она дает количественную оценку части системы, которая сохранила максимальную производительность после того, как дальнейшее прогрессирование распространения вируса более не ожидается, иными словами, вирус достиг своего полного насыщения.

Представим более подробно четыре метрики – три из них связаны с кривой заражения и одна не связана с ней.

Рассмотрим тип кривой распространения вируса, который реализуется при симуляциях системы защиты, описанной в данной статье. Форма кривой является сигмовидной, с резким ростом вначале и последующими достижением равновесия и остановкой эпидемии. Визуально можно выделить три величины, которые характеризуют этот тип кривых: процент зараженных узлов при насыщении, максимальная скорость распространения инфекции и время, за которое достигается насыщение.

Метрика 1. Процент зараженных хостов после того, как вирус достиг полного насыщения.

Метрика 2. Максимальная скорость распространения вируса, измеренная как процент заразившихся узлов в период, соответствующий пику заражения.

Метрика 3. Временна́я точка, в которой вирус достигает своего полного насыщения.

Большинство оценок, используемых при проектировании систем защиты от вирусов, основываются на влиянии этих систем на распространение вируса. Это дает прямое представление об общей эффективности системы, но не учитывает цену применяемых элементов защиты. Такая цена может выражаться в падении общей производительности из-за необходимости фильтрации, усложнении коммуникации, эффектах на локальную защиту для участвующих и не участвующих сетей. Приведем пример метрики, дающей количественное представление о части системы, не затронутой эффектами системы защиты.

Метрика 4. Количество неинфицированных узлов, находящихся в режиме без защиты, после того как вирус достиг своего полного насыщения.

Данную метрику дополнительно можно использовать для оценки эффективности системы.

Статистический анализ

Спецификацию в SHYMaude можно легко транслировать в спецификацию Maude [2], являющуюся выполняемой логической спецификацией [5]. Для анализа системы используем инструмент MultiVeStA [17], который, в свою очередь, основан на инструментах VeStA [7] и PVeStA [18]. MultiVeStA предлагает простую и понятную интеграцию существующих дискретно-событийных симуляторов, язык MultiQuaTEx (расширение языка QuaTEx [7]), который позволяет более компактное задание свойств систем, и клиент-серверную архитектуру для воспроизведения распределенных симуляций. Для анализа MultiVeStA использует статистический метод Монте-Карло. С помощью инструмента можно достигать предустановленного уровня точности результата. Также в системе возможны распределенные вычисления, основанные на клиент-серверном взаимодействии.

Точность результата в MultiVeStA задается с помощью параметров α и δ. Для полученного значения v при использовании данного алгоритма доверительный интервал имеет вид

$$\left[v(1-\delta), \frac{v}{1-\delta} \right].$$

Мы использовали величины $\alpha = 0{,}01$ и $\delta = 0{,}05$. Выбор параметра α гарантирует, что повторно вычисленные значения будут находиться в доверительном интервале с вероятностью 99 %.

К параметрам системы защиты также относятся количество исходящих узлов, размер группы, метрики строгости и сотрудничества. Эти параметры, а также параметры ложноположительного и ложноотрицательного темпов обнаружения вируса за интервал времени в использованном распределении Бернулли приведены в таблице.

Параметры модели
Model parameters

Параметры	Описание	Значение
N	Количество узлов	20
G	Размер группы	8
severity	Критичность	3
r	Сотрудничество	2
alpha = severity * r	Порог количества оповещений	6
p_{fp}	Ложноотрицательный темп обнаружения	0,1
p_{fn}	Ложноположительный темп обнаружения	0,05
t_s	Количество временных шагов до объявления насыщения	10

Насыщение распространения вируса было определено как стабильность процента зараженных узлов на протяжении 10 временны́х шагов.

Распространение вируса с течением времени. Начнем исследование системы с построения графика распространения вируса с течением времени. Используем следующее MultiQuaTEx-выражение для расчета процента зараженных узлов после определенного числа временны́х интервалов x:

```
MyProperty(x) = if {s.rval(1) >= x} then {s.rval(11)} else # MyProperty({x}) fi;
eval parametric(E[MyProperty(x)], x, 0.0, 1.0, 50.0);
```

где x варьируется от 0 до 50 с шагом 1. По вычисленным для каждого x значениям MultiVeStA строит график.

Здесь и далее QuaTEx-выражения будут приводиться так, как они представляются в MultiVeStA. Выражение s.rval(n) обозначает функцию на состоянии с параметром n, т. е. для каждого n можно задать некоторую функцию на состоянии. Принято использовать s.rval(1) как функцию, возвращающую текущее время. В примере выше s.rval(11) возвращает процент зараженных узлов.

По окончании вычислений, когда достигнут заданный уровень точности, MultiVeStA выводит численные результаты и построенный по ним график (рис. 1).

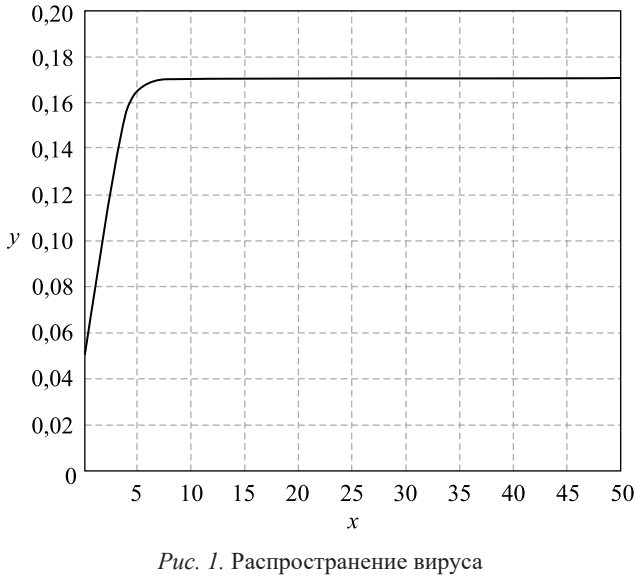

Рис. 1. Распространение вируса
Fig. 1. Virus propagation

Как видно из графика, распространение вируса быстро растет в течение 5 временны́х интервалов и достигает насыщения примерно после 10 таких интервалов. Пораженной оказывается менее 20 % системы, поэтому можно утверждать, что на заданных параметрах система статистически устойчива к этому типу вируса, если под статистической устойчивостью понимать факт того, что определенная существенная часть системы остается незараженной после достижения насыщения распространения вируса.

Процент зараженных хостов по достижении насыщения. Получим оценку значения метрики 1. MultiQuaTEx-выражение для вычисления этой оценки выглядит следующим образом:

```
percentInfectedAfterSaturation(percentage, count) =
   if {count <= 0.0} then {s.rval(11)} else
    if {s.rval(11) >= percentage}
     then # percentInfectedAfterSaturation({s.rval(11)}, {10.0})
     else # percentInfectedAfterSaturation({percentage}, {count - 1.0})
   fi fi;
eval E[percentInfectedAfterSaturation({0.0},{10.0})];
```

В данном примере мы инициируем расчет командой eval, вызывая рекурсивную по времени функцию percentInfectedAfterSaturation, в которой изначальный процент инфицированных хостов положен равным 0, а время до насыщения – 10 временны́м интервалам. Система рассчитывает математическое ожидание, дисперсию и доверительный интервал. В результате имеем, что процент зараженных узлов при достижении полного насыщения лежит в доверительном интервале [0,15; 0,2].

Максимальная скорость распространения вируса. Получим оценку значения метрики 2 с помощью следующего MultiQuaTEx-выражения:

```
MaxInfectionSpeed(speed, percentage, count) =
   if {count <= 0.0} then {speed} else
    if {s.rval(11) percentage > speed}
     then # MaxInfectionSpeed ({s.rval(11) percentage}, {s.rval(11)},{10.0})
     else # MaxInfectionSpeed ({speed}, {percentage}, {count 1.0})
   fi fi;
eval E[MaxInfectionSpeed ({0.0}, {0.0}, {10.0})];
```

Здесь мы инициируем расчет командой eval, вызывая рекурсивную по времени функцию MaxInfectionSpeed, в которой изначальная скорость заражения и процент инфицированных хостов положены равными 0, а время до насыщения – 10 временны́м шагам. В итоге имеем, что максимальная скорость заражения узлов лежит в доверительном интервале [0,07; 0,11].

Время до объявления насыщения вируса. Для вычисления оценки значения метрики 3 используем следующее MultiQuaTEx-выражение:

```
SaturationTime(percentage, count) =
   if {count <= 0.0} then {s.rval(1)} else
    if {s.rval(11) > percentage}
     then # SaturationTime({s.rval(11)}, {10.0})
     else # SaturationTime({percentage}, {count - 1.0})
   fi fi;
eval E[SaturationTime({0.0}, {10.0})];
```

В данном случае мы инициируем расчет командой eval, вызывая рекурсивную по времени функцию SaturationTime, где процент инфицированных хостов в начальный момент времени положен равным 0, а время до насыщения – 10 временны́м шагам. Полученное время до насыщения лежит в доверительном интервале [13,3; 13,9].

Анализ результатов статистического анализа

Важность обновления групп оповещений отдельных узлов продемонстрирована в [14], где были найдены контркарантинные стратегии вирусного распространения, использующие факт статической структуры групп. Для изучения значения алгоритма выбора групп мы исследовали модификацию алгоритма защиты с «неудачным» алгоритмом выбора групп, не обеспечивающим равномерное покрытие узлов группами в силу того, что каждый узел выбирает членов своей группы случайным образом не из всего множества узлов, а лишь из узлов, идентификаторы id которых расположены близко от него. Таким образом, некоторые узлы не получают достаточного количества оповещений, чтобы вовремя перейти в режим защиты.

После изменения алгоритма выбора членов групп мы повторили статистический анализ метрик 1–3 и получили следующие результаты:

• эффективность системы защиты понизилась, о чем свидетельствует распространение вируса с течением времени (рис. 2);

• процент зараженных хостов по достижении насыщения стал оцениваться доверительным интервалом [0,54; 0,63] вместо ранее полученного [0,15; 0,20];

• максимальная скорость заражения узлов оказалась в интервале [0,16; 0,21] вместо ранее оцененного [0,07; 0,11];

• время до насыщения оказалось в интервале [24,6; 27,2] вместо ранее оцененного [13,3; 13,9].

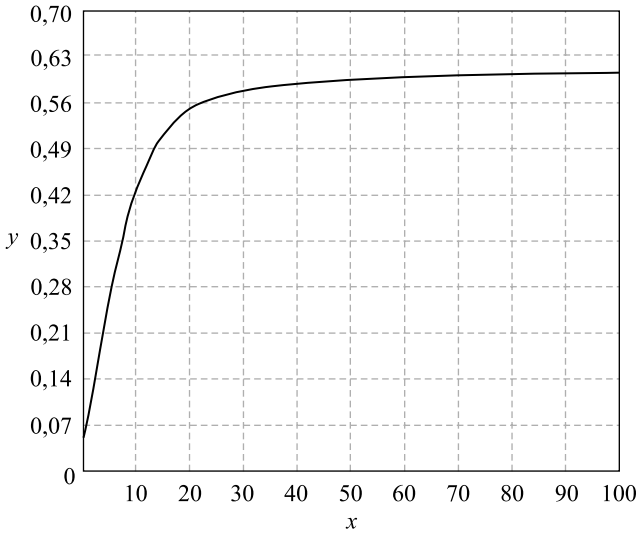

Рис. 2. Распространение вируса в случае неравномерного покрытия узлов группами оповещения

Fig. 2. Virus propagation in case of uneven coverage of nodes by alert groups

Из результатов видно, что «дефект» системы оказывает существенное влияние на ее эффективность и наглядно демонстрирует, как статистическая верификация помогает обнаружить неявные проблемы дизайна систем защиты. Данный «дефект» вполне может быть выявлен на этапе проектирования системы, и на этом этапе можно произвести ее доработку до последующего аналитического исследования, что иллюстрирует преимущество излагаемого подхода к разработке подобных систем.

Мы также изучили возможность «упрощения» системы путем исключения алгоритмического шага формирования групп. В данной модификации алгоритма узлы оповещают случайно выбранные узлы из всего множества узлов. Для такой модификации мы не обнаружили ухудшения эффективности системы защиты при выбранных параметрах. Поэтому при последующих исследованиях необходимо обратить внимание на указанное сокращение алгоритма, которое также иллюстрирует возможность потенциальных существенных «упрощений» системы на ранних этапах ее разработки.

Заключение

В статье проиллюстрирован подход к разработке модели сложных динамических распределенных стохастических систем с коммуникацией на ранних этапах, до проведения ее аналитического анализа, на примере предварительной модели стохастической групповой системы защиты от вирусов и экспериментов с ней. Данный подход базируется на модели РООСГС [1] и статистическом анализе, описанном в [2]. Продемонстрировано, как получение оценки основных численных параметров системы, обычно используемых для оценки систем подобного типа, дает возможность оценить систему до фиксации формальной модели и проведения полноценного анализа. В условиях предварительного эксперимента установлено, что система защиты позволяет сохранить значительный процент узлов незараженными при атаке вируса со случайным выбором заражаемого узла.

Изучено влияние алгоритма выбора групп оповещения на эффективность системы защиты. Анализ показывает, что важным условием этой эффективности является равномерное покрытие всего множества узлов группами оповещения, что обусловливает необходимость более тщательного изучения данного факта впоследствии. В нашем эксперименте также обнаружено, что существенное «упрощение»

алгоритма выбора групп оповещения не повлияло на эффективность системы. Это указывает на потенциальную возможность такого «упрощения» с сохранением свойств системы, что может быть проверено на более поздних этапах разработки. Можно утверждать, что применение изложенного статистического исследования систем на этапе их проектирования позволяет изучить важные аспекты системы до ее анализа аналитическими методами и на этом этапе исправить «дефекты» и определить некоторые потенциальные «упрощения» отдельных компонент, что может дать преимущество на более поздних стадиях разработки.

Библиографические ссылки

1. Шарыкин РЕ, Курбацкий АН. Модель распределенных объектно ориентированных стохастических гибридных систем. *Журнал Белорусского государственного университета. Математика. Информатика.* 2019;2:52–61. DOI: 10.33581/2520-6508-2019-2-52-61.

2. Шарыкин РЕ, Курбацкий АН. Верификация распределенных объектно ориентированных стохастических гибридных систем. *Вестник Гродненского государственного университета имени Янки Купалы. Серия 2. Математика. Физика. Информатика, вычислительная техника и управление.* 2019;9(3):123–132.

3. Meseguer J. Conditional rewriting logic as a unified model of concurrency. *Theoretical Computer Science.* 1992;96(1):73–155. DOI: 10.1016/0304-3975(92)90182-F.

4. Martí-Oliet N, Meseguer J. Rewriting logic: roadmap and bibliography. *Theoretical Computer Science.* 2002;285(2):121–154. DOI: 10.1016/S0304-3975(01)00357-7.

5. Clavel M, Durán F, Eker S, Lincoln P, Martí-Oliet N, Meseguer J, et al. Maude: Specification and programming in rewriting logic. *Theoretical Computer Science.* 2002;285(2):187–243. DOI: 10.1016/S0304-3975(01)00359-0.

6. Clavel M, Durán F, Eker S, Meseguer J. Chapter 1. Building equational proving tools by reflection in rewriting logic. In: Futatsugi K, Nakagawa AT, Tamai T, editors. *CAFE: an industrial-strength algebraic formal method.* [S. l.]: Elsevier; 2000. p. 1–31. DOI: 10.1016/B978-044450556-9/50061-7.

7. Sen K, Viswanathan M, Agha G. On statistical model checking of stochastic systems. In: Etessami K, Rajamani SK, editors. *Computer aided verification. Proceedings of the 17th International conference on computer aided verification.* Berlin: Springer; 2005. p. 266–280. (Lecture notes in computer science; volume 3576). DOI: 10.1007/11513988_26.

8. Briesmeister L, Porras P. Microscopic simulation of a group defense strategy. In: Nicol D, editor. *Proceedings of the 19th Workshop on principles of advanced and distributed simulation; 2005 June 1–3; Monterey, USA.* Los Alamitos: IEEE Computer Society; 2005. p. 254–261. DOI: 10.1109/PADS.2005.13.

9. Anagnostakis KG, Greenwald MB, Ioannidis S, Keromytis AD, Li D. A cooperative immunization system for untrusting Internet. In: Moreton N, editor. *The 11th IEEE International conference on networks; 2003 September 28 – October 1; Sydney, Australia.* Los Alamitos: IEEE Computer Society; 2003. p. 403–408. DOI: 10.1109/ICON.2003.1266224.

10. Nojiri D, Rowe J, Levitt K. Cooperative response strategies for large scale attack mitigation. In: Werner B, editor. *Proceedings DARPA information survivability conference and exposition; 2003 April 22–24; Washington, USA.* Los Alamitos: IEEE Computer Society; 2003. p. 293–302. DOI: 10.1109/DISCEX.2003.1194893.

11. Twycross J, Williamson MM. Implementing and testing a virus throttle. In: Paxson V, editors. *Proceedings of the 12th conference on USENIX security symposium; 2003 August 4–8; Washington, USA.* Berkeley: USENIX Association; 2003. p. 285–294.

12. Singh S, Estan C, Varghese G, Savage S. Automated worm fingerprinting. In: Brewer E, editor. *Proceedings of the 6th conference on symposium on operating systems design and implementation; 2004 December 6–8; San Francisco, USA.* Berkeley: USENIX Association; 2004. p. 45–60.

13. Kim H-A, Karp B. Autograph: Toward automated, distributed worm signature detection. In: Blaze M, editor. *Proceedings of the 13th conference on USENIX security symposium; 2004 August 9–13; San Diego, USA.* Berkeley: USENIX, The Advanced Computing Systems Association; 2004. p. 271–286.

14. Briesmeister L, Porras P. Automatically deducing propagation sequences that circumvent a collaborative worm defense. In: Hassanein H, editor. *Proceedings of the International performance computing and communications conference; 2006 April 10–12; Phoenix, USA.* Los Alamitos: IEEE Computer Society; 2006. p. 587–592. DOI: 10.1109/.2006.1629456.

15. Шарыкин РЕ. Спецификация коллаборационной системы защиты от вирусных атак в среде Maude [Интернет]. GitHub [процитировано 2 января 2020 г.]. Доступно по: https://github.com/shymaude/virusDefense/blob/master/defense.shymaude.

16. Briesemeister L, Porras PA, Tiwari A (Computer Science Laboratory). Model checking of worm quarantine and counter-quarantine under a group defense. Technical Report. Menlo Park: SRI International; 2005. Technical Report Number: SRI-CSL-05-03, SRI Project 13738.

17. Sebastio S, Vandin A. MultiVeStA: statistical model checking for discrete event simulators. In: Horvath A, editor. *Proceedings of the 7th International conference on performance evaluation methodologies and tools; 2013 December 10–12; Torino, Italy.* Brussels: Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering; 2013. p. 310–315. DOI: 10.4108/icst.valuetools.2013.254377.

18. AlTurki M, Meseguer J. PVeStA: a parallel statistical model checking and quantitative analysis tool. In: Corradini A, Klin B, Cirstea C, editors. *Algebra and coalgebra in computer science. International conference on algebra and coalgebra in computer science.* Berlin: Springer; 2011. p. 386–392. (Lecture notes in computer science; volume 6859).

References

1. Sharykin RE, Kourbatski AN. A model of distributed object-based stochastic hybrid systems. *Journal of the Belarusian State University. Mathematics and Informatics.* 2019;2:52–61. Russian. DOI: 10.33581/2520-6508-2019-2-52-61.

2. Sharykin RE, Kourbatski AN. Verification of distributed object-oriented stochastic hybrid systems. *Vestnik Grodnenskogo gosudarstvennogo universiteta imeni Yanki Kupaly. Seriya 2. Matematika. Fizika. Informatika, vychislitel'naya tekhnika i upravlenie.* 2019;9(3):123–132. Russian.

3. Meseguer J. Conditional rewriting logic as a unified model of concurrency. *Theoretical Computer Science.* 1992;96(1):73–155. DOI: 10.1016/0304-3975(92)90182-F.

4. Martí-Oliet N, Meseguer J. Rewriting logic: roadmap and bibliography. *Theoretical Computer Science.* 2002;285(2):121–154. DOI: 10.1016/S0304-3975(01)00357-7.

5. Clavel M, Durán F, Eker S, Lincoln P, Martí-Oliet N, Meseguer J, et al. Maude: Specification and programming in rewriting logic. *Theoretical Computer Science.* 2002;285(2):187–243. DOI: 10.1016/S0304-3975(01)00359-0.

6. Clavel M, Durán F, Eker S, Meseguer J. Chapter 1. Building equational proving tools by reflection in rewriting logic. In: Futatsugi K, Nakagawa AT, Tamai T, editors. *CAFE: an industrial-strength algebraic formal method.* [S. l.]: Elsevier; 2000. p. 1–31. DOI: 10.1016/B978-044450556-9/50061-7.

7. Sen K, Viswanathan M, Agha G. On statistical model checking of stochastic systems. In: Etessami K, Rajamani SK, editors. *Computer aided verification. Proceedings of the 17th International conference on computer aided verification.* Berlin: Springer; 2005. p. 266–280. (Lecture notes in computer science; volume 3576). DOI: 10.1007/11513988_26.

8. Briesmeister L, Porras P. Microscopic simulation of a group defense strategy. In: Nicol D, editor. *Proceedings of the 19th Workshop on principles of advanced and distributed simulation; 2005 June 1–3; Monterey, USA.* Los Alamitos: IEEE Computer Society; 2005. p. 254–261. DOI: 10.1109/PADS.2005.13.

9. Anagnostakis KG, Greenwald MB, Ioannidis S, Keromytis AD, Li D. A cooperative immunization system for untrusting Internet. In: Moreton N, editor. *The 11th IEEE International conference on networks; 2003 September 28 – October 1; Sydney, Australia.* Los Alamitos: IEEE Computer Society; 2003. p. 403–408. DOI: 10.1109/ICON.2003.1266224.

10. Nojiri D, Rowe J, Levitt K. Cooperative response strategies for large scale attack mitigation. In: Werner B, editor. *Proceedings DARPA information survivability conference and exposition; 2003 April 22–24; Washington, USA.* Los Alamitos: IEEE Computer Society; 2003. p. 293–302. DOI: 10.1109/DISCEX.2003.1194893.

11. Twycross J, Williamson MM. Implementing and testing a virus throttle. In: Paxson V, editors. *Proceedings of the 12th conference on USENIX security symposium; 2003 August 4–8; Washington, USA.* Berkeley: USENIX Association; 2003. p. 285–294.

12. Singh S, Estan C, Varghese G, Savage S. Automated worm fingerprinting. In: Brewer E, editor. *Proceedings of the 6th conference on symposium on operating systems design and implementation; 2004 December 6–8; San Francisco, USA.* Berkeley: USENIX Association; 2004. p. 45–60.

13. Kim H-A, Karp B. Autograph: Toward automated, distributed worm signature detection. In: Blaze M, editor. *Proceedings of the 13th conference on USENIX security symposium; 2004 August 9–13; San Diego, USA.* Berkeley: USENIX, The Advanced Computing Systems Association; 2004. p. 271–286.

14. Briesmeister L, Porras P. Automatically deducing propagation sequences that circumvent a collaborative worm defense. In: Hassanein H, editor. *Proceedings of the International performance computing and communications conference; 2006 April 10–12; Phoenix, USA.* Los Alamitos: IEEE Computer Society; 2006. p. 587–592. DOI: 10.1109/.2006.1629456.

15. Sharykin RE. Maude specification of the stochastic collaborative virus defense system [Internet]. GitHub [cited 2020 January 2]. Available from: https://github.com/shymaude/virusDefense/blob/master/defense.shymaude.

16. Briesemeister L, Porras PA, Tiwari A (Computer Science Laboratory). Model checking of worm quarantine and counter-quarantine under a group defense. Technical Report. Menlo Park: SRI International; 2005. Technical Report Number: SRI-CSL-05-03, SRI Project 13738.

17. Sebastio S, Vandin A. MultiVeStA: statistical model checking for discrete event simulators. In: Horvath A, editor. *Proceedings of the 7th International conference on performance evaluation methodologies and tools; 2013 December 10–12; Torino, Italy.* Brussels: Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering; 2013. p. 310–315. DOI: 10.4108/icst.valuetools.2013.254377.

18. AlTurki M, Meseguer J. PVeStA: a parallel statistical model checking and quantitative analysis tool. In: Corradini A, Klin B, Cirstea C, editors. *Algebra and coalgebra in computer science. International conference on algebra and coalgebra in computer science.* Berlin: Springer; 2011. p. 386–392. (Lecture notes in computer science; volume 6859).

Статья поступила в редколлегию 22.01.2020.
Received by editorial board 22.01.2020.