Всероссийский конкурс учебно-исследовательских работ
старшеклассников
по политехническим, естественным, математическим дисциплинам
для учащихся 9-11 классов

Направление:
Информатика**.**

Название работы:

Исследование процессов распространения вредоносного программного обеспечения в сетях различных конфигураций.

Выполнила:
Волкова Ксения Алексеевна, 11 Н класс, МБОУ “Лицей №1”, г. Пермь.

Руководитель:
Батин Сергей Евгеньевич.

Пермь - 2018

Оглавление.

Введение …… 2
Глава 1. Вредоносное ПО ……... 3
§ 1. Классификация вредоносного ПО ……..……..……..... 3
§ 2. Модели распространения вредоносного ПО ……. 5
§ 3. Конфигурация сети ..…… 7
Глава 2. Создание имитационной модели ……. 8
§ 1. Гипотезы ……..……. 8
§ 2. Создание имитационной модели ……. 9
§ 3. Анализ полученных результатов ……. 12
Заключение ……... 15
Список литературы …….. 16
Аннотация

Введение.

У каждого человека есть ПК, смартфон или планшетный компьютер, а с них, в свою очередь, несложно выйти в глобальную сеть - Интернет. Пользователи часто скачивают информацию, и некоторые даже не догадываются об угрозе заражения различными вирусами ровно до тех пор, пока вредоносное ПО не начинает вносить существенные изменения в работу операционной системы. Вирусы способны нанести огромный ущерб компьютеру, похитить конфиденциальную информацию и использовать её в дальнейшем против самих пользователей. Именно поэтому в современности возникла проблема создания различных способов борьбы с компьютерными вирусами. Чтобы эффективно с ними бороться, необходимо знать разновидности, особенности и способы распространения вирусов, т.е. исследовать процессы распространения вредоносного ПО. В этом и заключается актуальность данной работы.

Цель: исследование влияния конфигурации сети на скорость распространения вредоносного программного обеспечения.

Задачи:

➔ Изучение темы;
➔ Изучение различных подходов к моделированию;
➔ Разработка и реализация имитационной модели на ЭВМ.

В результате должна быть получена программа, случайно генерирующая конфигурацию, а также с разной вероятностью заражающая узлы, с помощью которой можно сделать выводы о влиянии конфигурации на распространение вредоносного ПО.

Глава 1. Вредоносное программное обеспечение.

§ 1. Классификация вредоносного ПО.

Вредоносное ПО можно разделить по принципу действия на несколько основных групп:

1.Вирус - это самовоспроизводящийся программный код, который внедряется в установленные программы без согласия пользователя. Заразиться можно разными способами: от нажатия вредоносной ссылки или файла в неизвестном письме до заражения на вредоносном сайте. При этом вирус может выполнять множество разных задач, направленных в первую очередь на принесение вреда ОС.

2.Черви являются в некотором роде вирусами, так как созданы на основе саморазмножающихся программ. Однако черви не могут заражать существующие файлы. Вместо этого червь поселяется в компьютер отдельным файлом и ищет уязвимости в Сети или системе для дальнейшего распространения себя. Некоторые черви существуют в виде сохраненных на жестком диске файлов, а некоторые поселяются лишь в оперативной памяти компьютера.

3.Троян. По своему действию является противоположностью вирусам и червям. Его предлагают загрузить под видом законного приложения, однако вместо заявленной функциональности он делает то, что нужно злоумышленникам. Трояны не самовоспроизводятся и не распространяются сами по себе. Однако с увеличением вала информации и файлов в Интернете трояна стало довольно легко подцепить. Нынешние трояны эволюционировали до таких сложных форм, как, например, бэкдор (троян, пытающийся взять на себя администрирование компьютера) и троян-загрузчик (устанавливает на компьютер жертвы вредоносный код).

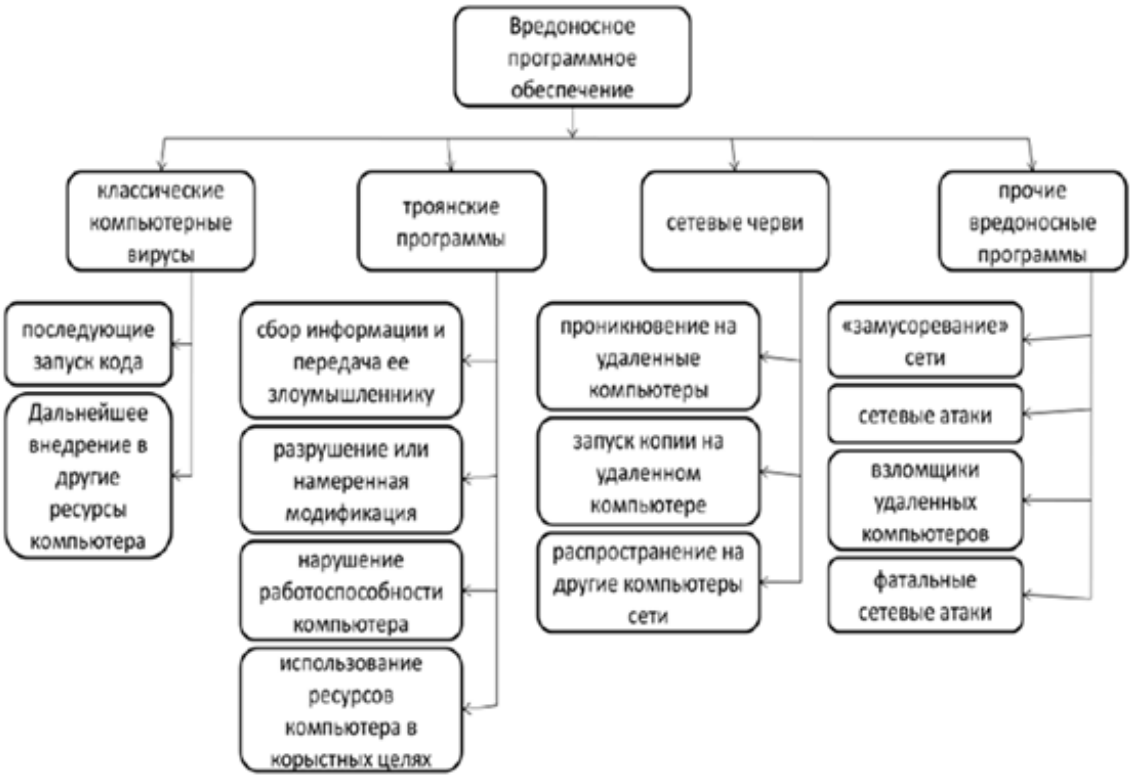


рис. 1

Сетевые черви в наше время являются самыми опасными среди вирусов, в виду того, что этот тип вирусов может быстро распространяться, а также на основе данных, полученных от червя, может быть создано другое вредоносное ПО. В данной работе будет использована модель сетевого червя, т.е. программы, способной к самостоятельному поиску новых узлов для заражения и использующей для распространения коммуникационную сеть.

§ 2. Модели распространения вредоносного ПО.

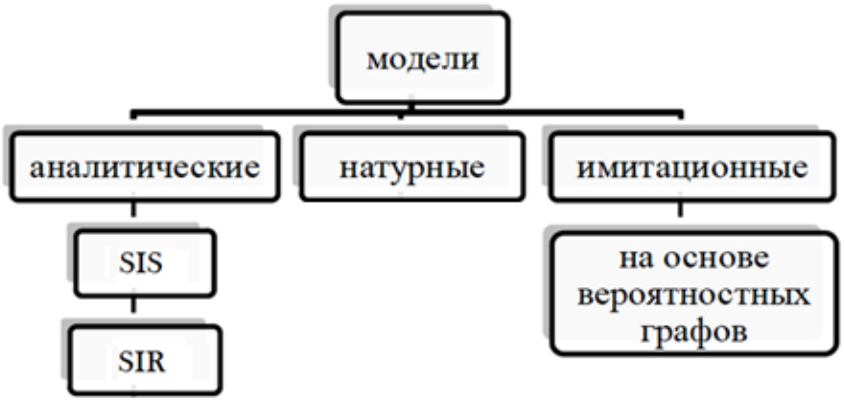


рис. 2

1. *Аналитические модели*. Процесс описывается системой дифференциальных уравнений.
    a. Простая эпидемиологическая модель - *SIS* (“Susceptible-Infected-Susceptible”) Произвольный хост сети, состоящей из постоянного количества N хостов, может находиться в двух состояниях – уязвимом (S) и инфицированном (I), т.е. $S+I=N$;
    b. SIR - модель. (“Susceptible–Infected–Removed model”) В ней хосты существуют в трех состояниях: уязвимом (S), зараженном (I) и невосприимчивом (R). Таким образом, $S+I+R=N$.

Недостатком аналитического моделирования является возможность исследования распространения только для известных и простых моделей размножения вредоносного ПО, а также не позволяет учитывать конфигурацию сети.

Преимуществом аналитического моделирования считается получение решения «в общем виде», а также высокая скорость моделирования конкретных сценариев для различных начальных условий.

2. *Натурные модели.* Разработка вредоносного ПО и проведение эксперимента на конкретной сети.

Недостатком является сложность в проведении эксперимента.

3. *Имитационные модели.* На основе вероятностных графов.

Этот подход позволяет избавиться от недостатков, присущих аналитическим моделям, учесть большинство факторов и исследовать степень их влияния на процесс распространения вредоносного ПО в сети.

§3. Конфигурация сети.

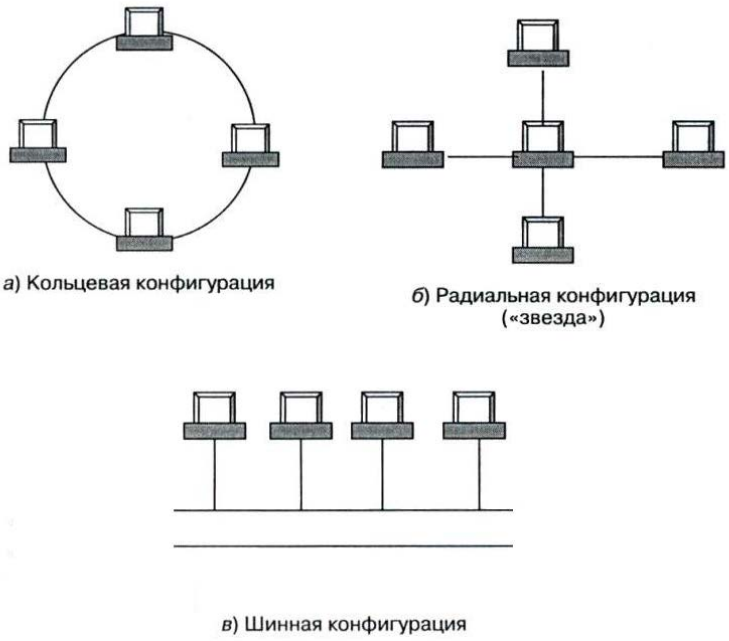


рис. 3

Шина — все компьютеры как бы построены в одну линию, т. е. от одного кабеля имеются отводы к каждому из компьютеров сети, причем концы кабеля являются незамкнутыми. Чаще всего такая схема применяется для соединения нескольких компьютеров, установленных в одном помещении. Наиболее яркий недостаток этой конфигурации— при любом обрыве в кабеле теряется связь между всеми компьютерами.

Кольцо — все компьютеры, как и в предыдущем случае, соединены между собой при помощи одного кабеля, концы которого соединены между собой. Теперь любой разрыв кабеля уже не приводит к потере связи между компьютерами. Для соединения нескольких колец используются специальные устройства.

Звезда — каждый компьютер сети отдельным кабелем подключен к одному ПК, который играет роль файлового сервера. Наиболее распространенная схема — локальная сеть. Обрыв в кабеле приводит к потере контакта только с одним компьютером или же сегментом сети.

Глава 2. Создание имитационной модели.

§ 1. Гипотезы.

В данной работе будет использоваться имитационная модель, поскольку она позволяет избавиться от недостатков, присущих аналитическим моделям, которые не позволяют учесть конфигурацию сети, и является упрощенной, в отличие от натурной модели.

Для создания модели были приняты следующие *предположения*:

1. Модель описывается с помощью вероятностных графов.
2. Узлы могут быть в двух состояниях: уязвимый и зараженный.
3. Нет возможности перехода из зараженного состояния в уязвимое.
4. Изначально заражен только один узел.
5. У узлов есть степень уязвимости, т.е. чем ближе число к 1, тем менее уязвим узел.
6. У каждого узла есть минимум две связи с другими узлами.
7. Вероятность заражения уязвимого узла от разных зараженных узлов, связанных с ним, разная.
8. *Один узел не может быть связан со всеми узлами сразу (не обязательно).

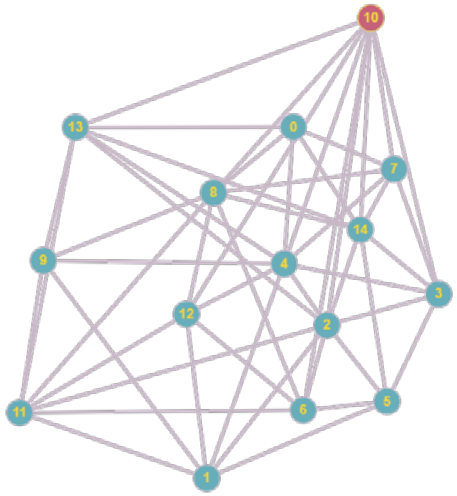


рис. 4

§ 2. Создание имитационной модели.

Для создания модели был использован язык C++. Программа создавалась в интегрированной среде разработки - IDE.

Данные о связях между компьютерами хранятся в двумерном массиве.

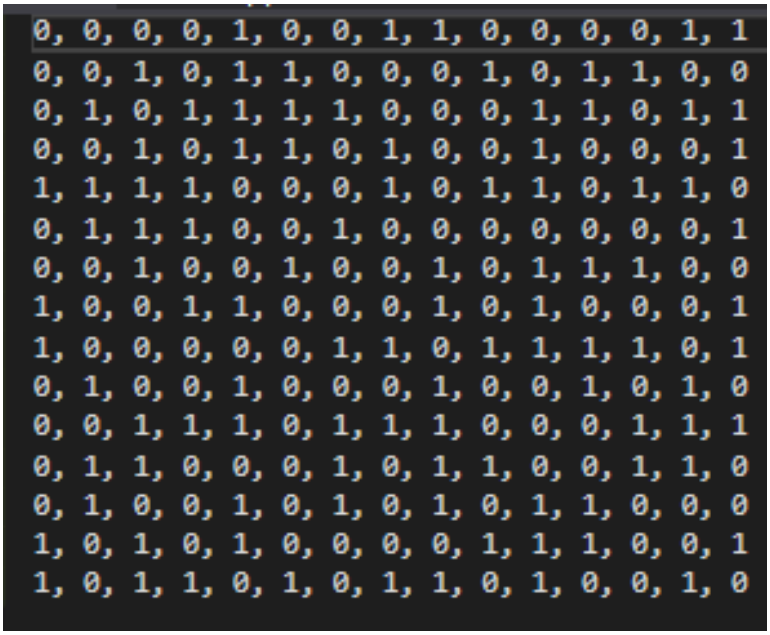


рис. 5

Изначально массив случайным образом заполняется либо 1, либо 0, после чего необходимо этот массив проанализировать, чтобы сеть была замкнутой:

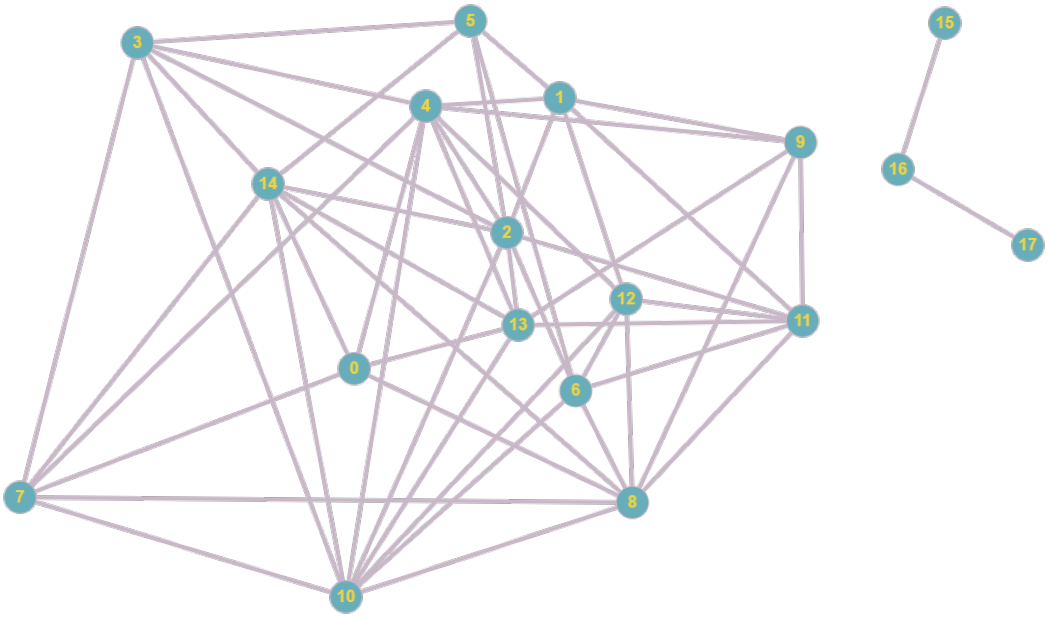


рис. 6

Чем и занимается функция *Check*:

```
void Check(unsigned int size, float *arr[])
{
    unsigned int i, j, n, x;
    for (i = 0; i<size; i++)
    {
        n = 0;
        for (j = 0; (j<size) && (n<2); j++)
        {
            if (arr[i][j] == 1)
                n++;
        }
        if (n == 0)
        {
            while (n != 2)
            {
                x = rand() % size;
                cout << "n, x \n" << n << x;
                if ((arr[i][x] == 0) && (x>i))
                {
                    arr[i][x] = 1;
                    arr[x][i] = arr[i][x];
                    n++;
                }
            }
        }

        if (n == 1)
        {
            while (n != 2)
            {
                x = rand() % size;

                if ((arr[i][x] == 0) && (x>i))
                {
                    arr[i][x] = 1;
                    arr[x][i] = arr[i][x];
                    n++;
                }
            }
        }
```

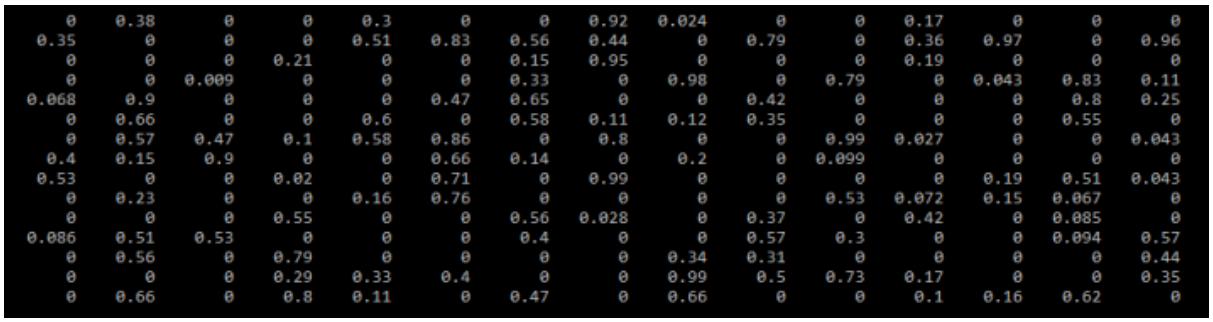
рис. 7

Чтобы внести в заражение элемент случайности, значения “1” в массиве заменяются на значения с плавающей точкой (0;1). С помощью функции *AntiV:*

```
void AntiV(unsigned int size, float *arr[])
{
    unsigned int i, j;
    for (i = 0; i<size; i++)
    {
        for (j = 0; j<size; j++)
        {
            if (arr[i][j] == 1)
            {
                arr[i][j] = (rand() % 1000) / 1000.0;
                if (arr[i][j] == 0)
                    arr[i][j] = 0.001;
            }
        }
    }
}
```

рис. 8

массив приобретает вид:



0	0.38	0	0	0.3	0	0	0.92	0.024	0	0	0.17	0	0	0
0.35	0	0	0	0.51	0.83	0.56	0.44	0	0.79	0	0.36	0.97	0	0.96
0	0	0	0.21	0	0	0.15	0.95	0	0	0	0.19	0	0	0
0	0	0.009	0	0	0	0.33	0	0.98	0	0.79	0	0.043	0.83	0.11
0.068	0.9	0	0	0	0.47	0.65	0	0	0.42	0	0	0	0.8	0.25
0	0.66	0	0	0.6	0	0.58	0.11	0.12	0.35	0	0	0	0.55	0
0	0.57	0.47	0.1	0.58	0.86	0	0.8	0	0	0.99	0.027	0	0	0.043
0.4	0.15	0.9	0	0	0.66	0.14	0	0.2	0	0.099	0	0	0	0
0.53	0	0	0.02	0	0.71	0	0.99	0	0	0	0	0.19	0.51	0.043
0	0.23	0	0	0.16	0.76	0	0	0	0	0.53	0.072	0.15	0.067	0
0	0	0	0.55	0	0	0.56	0.028	0	0.37	0	0.42	0	0.085	0
0.086	0.51	0.53	0	0	0	0.4	0	0	0.57	0.3	0	0	0.094	0.57
0	0.56	0	0.79	0	0	0	0	0.34	0.31	0	0	0	0	0.44
0	0	0	0.29	0.33	0.4	0	0	0.99	0.5	0.73	0.17	0	0	0.35
0	0.66	0	0.8	0.11	0	0.47	0	0.66	0	0	0.1	0.16	0.62	0

рис. 9

Далее эти значения сравниваются со случайным дробным числом. Если табличное значение меньше случайного, то происходит заражение.

Шаги заражения условны. С их помощью можно оценить скорость заражения сети.

§ 3. Анализ полученных результатов.

В первую очередь я рассмотрела два самых простых случая - это “шинная” конфигурация и конфигурация типа “Звезда”:

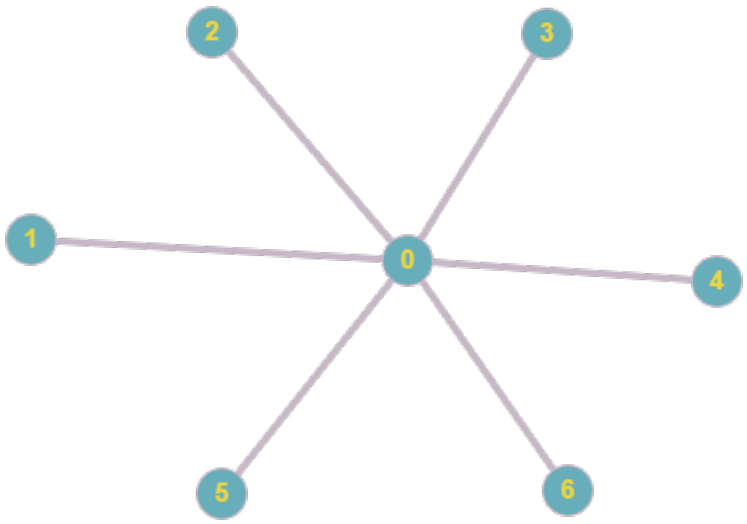


рис. 10 а)

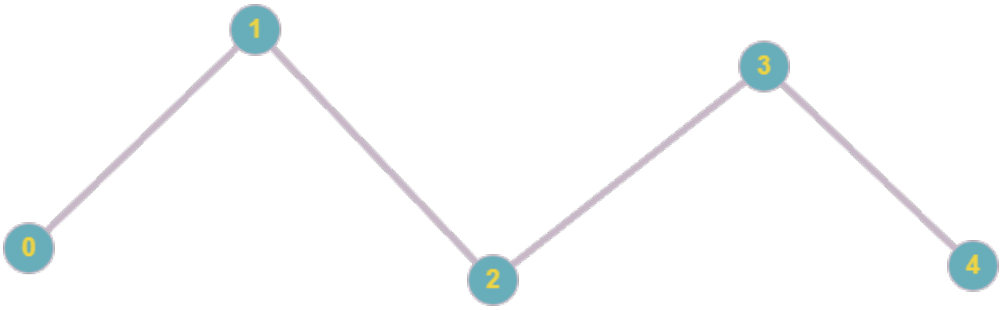
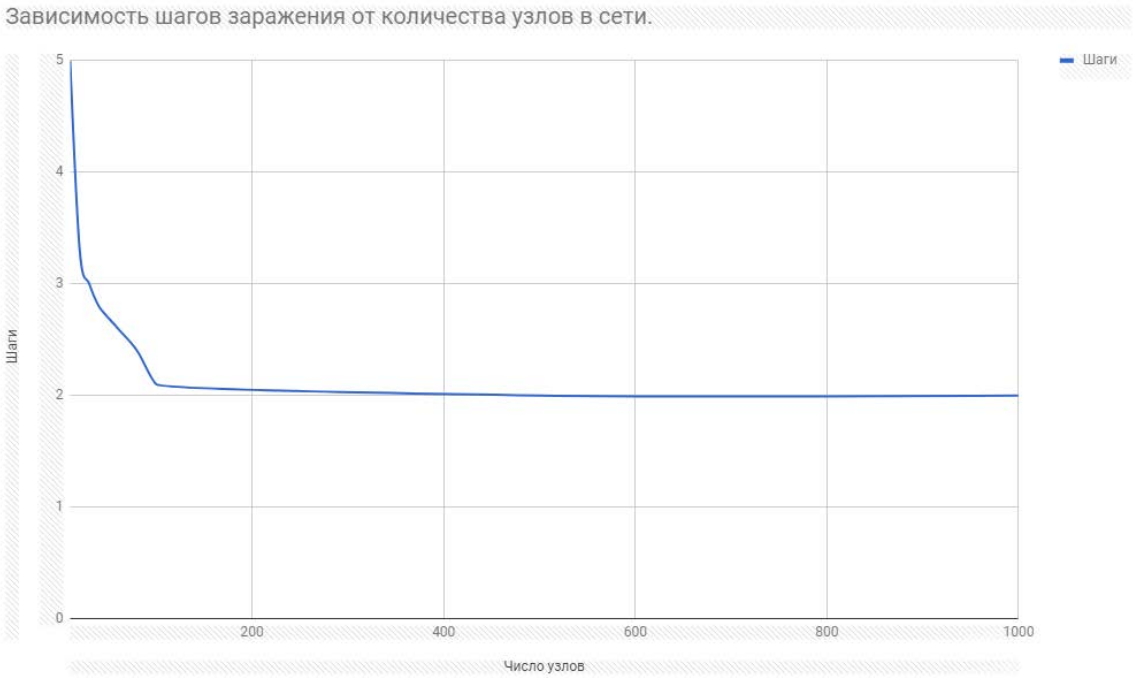


рис. 10 б)

Результаты были вполне ожидаемые: на рис. а заражение происходило гораздо быстрее, чем на рис. б

Однако такие модели далеки от реальных сетей ввиду того, что, например, во всем известной сети Интернет у каждого узла бессчетное количество связей. Для того, чтобы приблизить модель к реальности, необходимо было увеличить как число узлов в сети, так и возможные связи между ними.



При увеличении числа узлов и связей между ними встала проблема уменьшения скорости заражения. Несмотря на вероятность заражения, узлы переходили из уязвимого состояния в зараженное обратно пропорционально их количеству. Это можно объяснить тем, что чем больше связей, тем больше вероятность того, что узел заразится. Чтобы проще понять масштабы, достаточно взглянуть на граф:

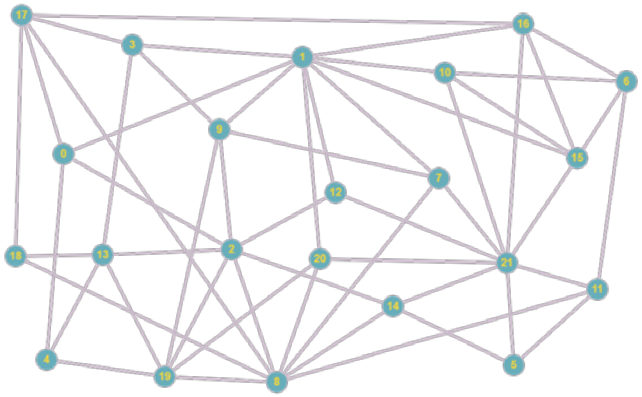


рис. 11[1]

[1] Для визуализации графов был использован интернет ресурс: http://graphonline.ru/ Граф создавался на основе матрицы смежности.

Созданная имитационная модель позволяет создать сеть, максимально приближенную к конфигурации на рис. 10 а) , т.е. один узел может быть соединен со всеми узлами в сети, а это приводит к практически мгновенному заражению.

Наиболее устойчивой к заражениям конфигурацией оказалась конфигурация на рис. 10 б).

Заключение.

В ходе работы были исследованы разновидности вредоносного ПО (вирусы, черви, трояны), а также существующие модели его распространения - аналитические (SIS и SIR модели), натурные и имитационные (на основе вероятностных графов). Аналитические модели имеют свои плюсы и минусы, как и имитационные. Оба этих способа удобны, в отличие от натурных моделей. Был произведен выбор метода моделирования сети, а также типа вредоносного ПО, используемого в модели. Была разработана имитационная модель, реализована на ЭВМ и проанализирована. Для разработки модели был выбран язык программирования C++ и программа была написана с помощью IDE.

В результате была получена программа, случайно генерирующая конфигурацию, а также с разной вероятностью заражающая узлы, с помощью которой были сделаны выводы о влиянии конфигурации на распространение вредоносного ПО.

Список литературы:

1. Котенко И. В., Воронцов В. В. Аналитические модели

распространения сетевых червей // Труды СПИИРАН. Вып. 4. — СПб.: Наука, 2007
2. Маленкович С. Классификация вредоносных программ. URL: https://blog.kaspersky.ru/klassifikaciya-vredonosnyx-programm/2200/ (дата обращения: 10.06.2017)
3. Задорина Н.А., Мурашова И.Ю. “Моделирование процесса распространения вредоносных программ в локальной сети” // Вопросы современных технических наук: свежий взгляд и новые решения. - 2017. - Т. IV. - С. 78-81.

Annotation.

The main subject of it is the study of the spread of viruses.

The aim of the research is to study the problem of creation of a simulation model and it’s analyze.

To accomplish the task it was necessary to study literature, study the ways of modeling, develop a simulation model and it's implement on a computer.
During the research a model was developed and implemented on a computer; a choice of a method of modeling a network is made; introduced an element of randomness of infection, the generation of networks; the model was applied to select the most resistant to infection patterns of networks.
In conclusion it should be noted that the object set in the beginning of the research have been accomplished.
This work is up-to-date because viruses are often found in the most popular network - Internet.