

Санкт-Петербургский Политехнический университет Петра Великого
Институт прикладной математики и механики
Высшая школа прикладной математики и вычислительной физики

Курсовая работа
по дисциплине «Компьютерные сети»

Выполнил студент гр. 5040102/00201

Денисов П.П.

Преподаватель

Баженов А.Н.

Санкт-Петербург

2022 г

Оглавление

Постановка задачи.....	3
Ход работы.....	3
Описание программы.....	3
Демонстрация работы программы.....	3
Заключение.....	5
Список литературы	5

Постановка задачи

Требуется смоделировать систему для слежения за солнечным зайчиком, движущимся по экрану. Система состоит из четырех камер (отдельных узлов сети) и связанного с ними роутера.

Для устранения угрозы подмены данных с камеры злоумышленником, реализовать протокол PAXOS.

Ход работы

Описание программы

Программа была выполнена на языке программирования Python 3.7 в среде разработки JetBrains PyCharm CE.

Программа содержит следующие основные классы:

Sender – класс, отвечающий за отправку пакетов. Может работать как по протоколу Go-Back-N, так и по протоколу Selective Repeat. Выбор протокола происходит при инициализации класса. Также передаваемыми параметрами являются: размер окна, количество данных для передачи, вероятность потери передаваемого пакета.

Receiver – класс, отвечающий за прием пакетов. Отправляет сообщение ACK, если пакет был успешно получен, и сообщение NAK в противном случае.

Screen – класс, представляющий собой экран, по которому движется солнечный зайчик. Размеры и координаты экрана, а также координаты и диаметр солнечного зайчика могут быть заданы произвольно. В данном классе реализован расчет точек пересечения с матрицами камер.

Sensor – класс, представляющий собой узел сети с камерой (сенсором). Имеет возможность проведения массовой рассылки по протоколу Go-Back-N либо Selective Repeat всем остальным узлам с камерами. Данная рассылка используется для реализации алгоритма Лэмпорта, используемого для борьбы со злоумышленниками.

DesignatedRouter – класс, представляющий собой специально выделенный роутер. Он является смежным со всеми остальными узлами (с камерами) в сети и хранит её топологию. После получения от камер координат их пересечения с солнечным зайчиком рассчитывает его центр.

Демонстрация работы программы

На рис. 1 представлен экран с солнечным зайчиком, реализуемый классом *Screen*. Солнечный зайчик движется случайно в пределах экрана с некоторым заданным шагом. При каждом новом движении рассчитываются точки пересечения окружности с матрицами камер. На рисунке границы видимости камер выделены черными линиями. Каждая камера видит ровно 1/4 экрана.

Для размера солнечного зайчика пришлось ввести ограничение. Необходимо, чтобы его диаметр был не меньше ширины видимости камеры. В противном случае зайчик может оказаться в поле зрения только одной камеры, что исключает его пересечения с матрицами камер.

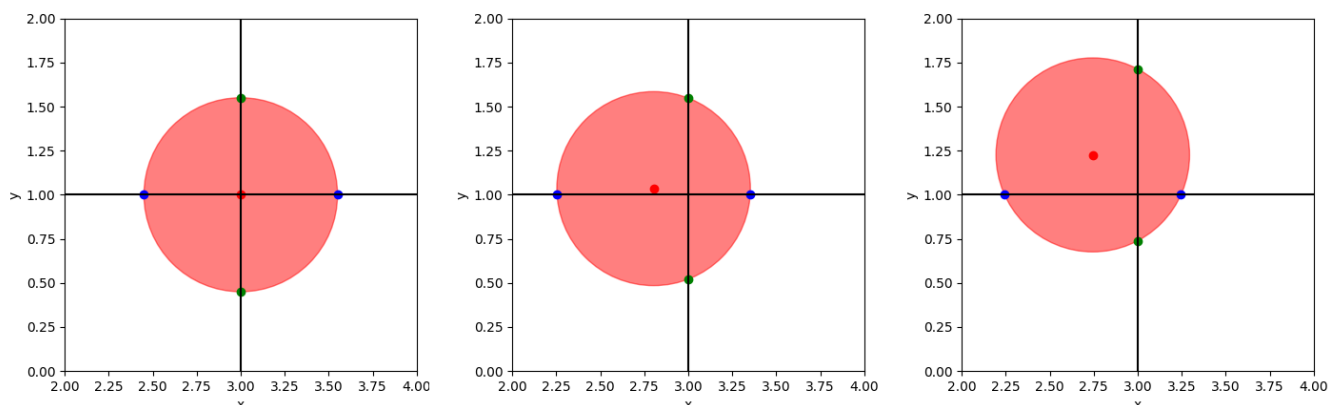


Рис. 1 Движение солнечного зайчика по экрану

Узел с камерой (класс *Sensor*), получив видимые ей координаты пересечения с солнечным зайчиком, начинает обмениваться координатами с другими узлами сети, реализуя алгоритм византийских генералов. Передача данных осуществляется по протоколам Go-Back-N либо Selective Repeat (можно выбрать один из них).

Для устранения угрозы подмены данных с камеры злоумышленником, был реализован алгоритм Лэмпорта. Рассмотрим его реализацию в данной программе.

В нашем случае имеется четыре камеры, одна из которых взломана нарушителем и передает неверные данные. Пусть взломанной камерой будет камера под номером 3. В подобном случае алгоритм осуществляется в четыре шага.

1-й шаг. Каждый узел с камерой посылает всем остальным сообщение со своими координатами пересечения. Нетронутые злоумышленником камеры указывают действительные координаты, в то время как взломанная камера посылает случайные значения.

2-й шаг. Каждый узел формирует свой вектор из имеющейся информации:

- Вектор камеры №1: (1, 2, x , 4);
- Вектор камеры №2: (1, 2, y , 4);
- Вектор камеры №3: (1, 2, 3, 4);
- Вектор камеры №4: (1, 2, z , 4).

Здесь: 1 – координаты, полученные от первой камеры, 2 – от второй, 4 –

от четвертой. А x , y , z — случайные координаты, отправленные злоумышленником.

3-й шаг. Каждый посылает свой вектор всем остальным (узел 3 посылает опять произвольные значения). После этого у каждого узла с камерой есть по четыре вектора:

cam1	cam2	cam3	cam4
(1,2, x, 4)	(1,2, x, 4)	(1,2, x, 4)	(1,2, x, 4)
(1,2, y, 4)	(1,2, y, 4)	(1,2, y, 4)	(1,2, y, 4)
(a, b, c, d)	(e, f, g, h)	(1,2,3,4)	(i, j, k, l)
(1,2, z, 4)	(1,2, z, 4)	(1,2, z, 4)	(1,2, z, 4)

4-й шаг. Каждый узел с камерой определяет для себя координаты пересечений всех остальных камер. Чтобы определить координаты i -й камеры, каждый узел сети берёт по три числа — координаты этой камеры, пришедшие от всех узлов, кроме узла под номером i . Если какое-то значение повторяется среди этих трех чисел как минимум два раза, то оно помещается в результирующий вектор, иначе соответствующий элемент результирующего вектора остается пустым. Таким образом все исправные камеры получают один и тот же вектор, согласие достигнуто.

При запуске программы для положений зайчика, представленных на рис. 1, программа определила центр зайчика как [3.00, 1.00], [2.80, 1.05], [2.75, 1.23].

Заключение

На языке программирования Python 3.7 была реализована программа, моделирующая систему слежения за солнечным зайчиком, которая состоит из четырех камер (отдельных узлов сети) и связанного с ними роутера.

Был реализован алгоритм Лэмпорта для устранения угрозы подмены данных с камеры злоумышленником.

Список литературы

1. А.Н. Баженов, Компьютерные сети, курс лекций
2. Программная реализация <https://github.com/Ppasha9/computer-network-coursework>