K. Borisenko, Ya. Bekeneva, N. Shipilov, A. Shorov

The simulation system for developing and testing protection methods against DDoS-attacks with the ability to connect the real nodes

Keywords: DDoS attack, virtual network, real network, simulation, scenario of clients' behavior, defense methods, OMNeT++, PlanetLab.

Abstract. This paper is devoted to simulation system for security process modeling in computer networks. Created system allows to significantly easier construct topologies and scenarios of clients' behavior for making experiments in comparison of testbed solutions. In addition, system allows embedding known or novel architecture-dependent defense methods on any of network's nodes and on real server for improving accuracy of experiments.

К.А. Борисенко, Я.А. Бекенева, Н.Н. Шипилов, А.В. Шоров

СИСТЕМА ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ ДЛЯ РАЗРАБОТКИ И ТЕСТИРОВАНИЯ МЕТОДОВ ЗАЩИТЫ ОТ DDOS-АТАК С ВОЗМОЖНОСТЬЮ ПОДКЛЮЧЕНИЯ РЕАЛЬНЫХ УЗЛОВ

Введение

DDoS-атака – распределенная атака типа «отказ в обслуживании» различных Интернет-сервисов. При успешном совершении подобных атак на сервер, сервер перестает отвечать на легитимные запросы от пользователей. Крупным DDoS-атакам подвергаются сайты правительства и органов власти, сайты ведущих IT корпораций Amazon, Yahoo, Microsoft и т.д. Эти мощные корпорации, имеющие огромные ресурсы, не всегда могут справиться с атаками, и отразить нападение.

Мировые лидеры по информационной безопасности [1] ставят необходимость обнаружения и противостояния DDoS-атакам как первостепенную задачу в своих исследованиях и разработках. Это свидетельствует о том, что разработка и внедрение методов защиты от DDoS-атак является актуальной задачей.

В прошлых работах авторами была представлена система имитационного моделирования компьютерных сетей и различных типов DDoS-атак. Была поставлена задача разработки механизма включения реальных компьютеров в моделируемую сеть с целью повышения точности проведения экспериментов [2].

В данной статье описана реализация этой задачи, а также представлены эксперименты, произведенные с участием реальных узлов.

Релевантные работы

На сегодня большое количество исследователей и компаний ищут максимально эффективные способы защиты сервисов от DDoS-атак.

В статье [3] представлена система, использующая Spirent Test Center [4] в качестве устройства, генерирующего трафик. В работе представлены шаблоны для настроек конфигурации IP-адресов, сценариев атак. Платформа Spirent Test Center воспроизводит генерацию трафика со следующими ограничениями: размер буфера трафика ограничен, IP-адресы атакующих хостов получают настройки в конкретной заранее определенной последовательности. Кроме того, sequence number генерируется одинаково у всех хостов.

В обзоре [5] рассмотрена система, разработанная Ixia [6]. Авторы отмечают, что система позволяет в режиме реального времени защитить от DDoS-атак мощностью до 70 Гбит/с.

В статье [7] разработана система моделирования DDoS атак, которая позволяет имитировать сеть с различным поведением клиентов внутри нее. Авторами статьи была описана система, а также представлены результаты тестирования известных методов защиты от DDoS атак, а также были разработаны и протестированы собственные методы защиты.

Система моделирования, представленная в данной работе, позволяет внедрить любой известный или созданный пользователем механизм защиты в модель реальной сети, созданной пользователем. Защитные механизмы могут быть архитектурно-зависимыми, что также позволяет проводить эксперименты максимально приближенно к реальной сети. Важным преимуществом является возможность подключения реальных узлов к виртуальной сети, что позволит увеличить точность выполняемых экспериментов, а также протестировать различные настройки и типы серверов.

Формальное описание компонентов системы моделирования

Для того чтобы понять всю структуру разработанной системы, необходимо описать параметры моделей, используемых в ней. Далее дано формальное описание моделей различных компонентов системы.

Формальная модель сети:

$$Network = \left\langle NetConf,\ Router_{1..q},\ Host_{1..n},\ Server_{1..m} \right\rangle, \quad (1)$$

где: *NetConf* – настройки сети; *Router* – маршрутизатор, их в сети должно быть не менее 1; *Host* – хост, их в сети должно быть не менее 1; *Server* – сервер, их в сети должно быть не менее 1.

Настройки сети представлены в виде следующей модели:

$$NetConf = \left\langle IPStart,\ ExtIP,\ ServPath,\ NetPar \right\rangle, \quad (2)$$

где: *IPStart* – начальный IP-адрес для присвоения узлам. Например, если *IPStart* = = "1.0.0.1", а сеть состоит из 20 узлов (персональных компьютеров (ПК), маршрутизаторов, серверов), то IP-адрес последнего узла будет равен 1.0.0.20; *ExtIP* – IP-адрес внешнего сервера. На данном этапе в модели используется только один внешний (реальный) сервер, но модель можно переделать и подключить любое количество внешних серверов; *ServPath* – путь к виртуальному серверу, который будет являться заглушкой (отражает положение реального сервера) для виртуальной сети; *NetPar* – на основании вышеописанных компонентов каждый узел сети получит свои настройки интерфейсов, таблиц маршрутизации.

В виртуальной сети, для успешной работы модели необходимо создать виртуальный сервер с IP-адресом реального сервера. Это позволит модели создать необходимые параметры *NetPar* и, когда пакет дойдет до маршрутизатора, в локальной сети которого находится виртуальный сервер, пакет будет перенаправлен в реальную сеть.
Настройки маршрутизации сети могут быть представлены следующим образом:

$$NetPar = \langle RouteTable,\ IfaceTable,\ NotifyBoard \rangle, \tag{3}$$

где: *RouteTable* – таблица маршрутизации пакетов; *IfaceTable* – таблица интерфейсов узла; *NotifyBoard* – утилита поиска дальнейшего пути для полученного узлом пакета. Она сопоставляет IP-адрес получателя с таблицей маршрутизации, в случае успешного сопоставления отправляет пакет по маршруту, в случае неудачи – узел генерирует пакет с ошибкой.

Теперь рассмотрим модель маршрутизатора:

$$Router_k = \langle NetPar,\ NPcap,\ Def_{0..n},\ ExtDevN_{0..q},\ DelConf \rangle, \tag{4}$$

где: *NetPar* – параметры маршрутизатора, полученные при инициализации *Netconf* (2); *NPcap* – количество модулей учета потока трафика; *Def* – алгоритм защиты, запущенный на маршрутизаторе. Алгоритм пишется программно, в маршрутизаторе может присутствовать как любое количество алгоритмов, так и ни одного; *ExtDevN* – количество интерфейсов, подключенных к внешней сети. Необходимо для маршрутизаторов, в локальной сети которых находится внешний узел; *DelConf* – настройки задержки для пакетов. Служит для придания виртуальной сети характеристик реальной: задержку между хостами, разброс задержек.

Маршрутизатор, в локальной сети которого расположен реальный сервер, для удобства, будем называть перенаправляющим маршрутизатором.
Далее представлены модели, которые могут присутствовать в маршрутизаторе в зависимости от значений параметров его модели (4):

$$Pcap_k = \langle FileName,\ IfaceCapture \rangle, \tag{5}$$

где: *FileName* – имя файла, в который будет записываться лог трафика; *IfaceCapture* – имя интерфейса, с которого будет записываться проходящий трафик.

$$ExtDev_k = \langle ExtRoute,\ \text{ExtIface},\ Device, FilterString \rangle, \tag{6}$$

где: *ExtRoute, ExtIface* – таблица маршрутизации и интерфейса для перенаправления пакетов в реальную сеть. Это маршруты и интерфейсы, которые необходимо добавить к таблицам маршрутизатора, созданным с *NetPar*, чтобы сделать коммутацию пакетов в реальную сеть и обратно доступной; *Device* – имя реального интерфейса, на который будут перенаправляться пакеты, предназначенные для реальной сети. *FilterString* – настройка для фильтрации пакетов, необходимых для получения виртуальной сетью.

Рассмотрим модель виртуальных клиентов:

$$Host_k = \langle NetPar,\ DDoSApp_{0..1} \rangle, \tag{7}$$

где: *NetPar* – параметры клиента, полученные при инициализации *Netconf* (2); *DDoSApp* – приложение, которое будет выполнять атаку по заданному сценарию.

В модели сценария, выполняющего атаку на сервер, предусмотрены следующие параметры:

$$DDoSApp = \left\langle \begin{matrix} VictimPath,\ AType,\ Lvl,\ Dport \\ DeltaT,\ AStart, AEnd, MaxP, SpecialP \end{matrix} \right\rangle, \quad (8)$$

где: *VictimPath* – путь к серверу-жертве (для расширения возможностей атак на несколько серверов); *Atype* – тип атаки на сервер: 1 – HTTP Flooding, 2–7 – различные варианты TCP Flooding (SYN, SYN-ACK, RST, UDP и др.); *Lvl* – количество клиентов, участвующих в атаке в процентном соотношении; *Dport* – порт назначения. Порт отправления определяется каждый раз случайным образом; *DeltaT* – время между отправками пакетов (в случае HTTP Flooding – между сессиями); *AStart, AEnd* – время начала и окончания атаки, можно задать различные условия для интервалов старта и окончания атак между клиентами; *MaxP* – максимальное количество отправок пакетов (в случае HTTP атаки – сессий). Если стоит –1, то атака будет проходить на всем промежутке времени между *AStart* и *AEnd; SpecialP* – специальные параметры для атаки. Для HTTP – строка HTTP запроса, для TCP – возможность включить подмену IP-адреса отправителя.

Разработанные модели позволяют подключить любое количество внешних серверов к виртуальной сети. Также можно настроить сценарии атак (8) и методы защиты сети от DDoS атак в маршрутизаторе (4).

Архитектура и реализация системы

Архитектуру системы можно представить следующим образом (рис. 1).

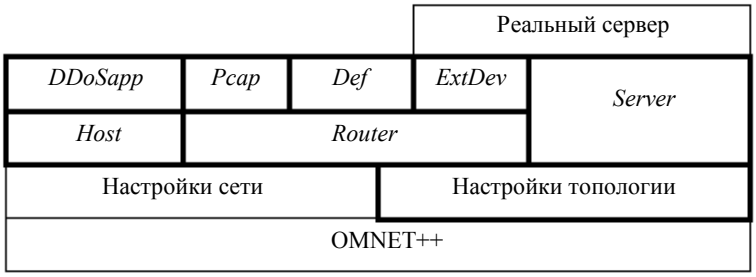

Рис.1. Архитектура разработанной системы

Для разработки системы использовалась дискретно-событийная система моделирования OMNeT++ [8]. Она была выбрана в качестве ядра системы, так как находится в открытом доступе, поддерживается большой командой разработчиков. Кроме того, создано большое количество библиотек, которые позволяют создавать характеристики виртуальной сети максимально схожими с характеристиками реальной сети.

В качестве настроек сети, коммутации пакетов используется библиотека INET [9]. В качестве настроек топологии была доработана библиотека ReaSE [10]. Ее поддержку авторы прекратили в 2011 году, поэтому пришлось обновить ее до версии OMNeT++ 4.5. На следующих уровнях представлены модели, описанные в разделе 2. Также необходимо иметь настроенный реальный веб-сервер для проведения экспериментов. На рис. 1 жирными линиями выделены те модели, которые были разработаны авторами статьи.

Созданные модели и архитектура были использованы при разработки системы. Разработанная система позволяет удобно и быстро построить различные топологии виртуальных сетей и настроить подключение к реальному серверу. Система позволяет создавать сценарии поведения клиентов, при этом модель сценариев атак *DDoSApp* можно использовать как сценарий поведения легитимного клиента. Все модели могут быть легко изменены. В экспериментах можно использовать любой алгоритм защиты:

как уже существующий, так и разработанный пользователем. Использование системы позволит максимально сократить время на подготовку экспериментов, что увеличит количество времени на подготовку и тестирование новых методов защиты.

Из-за специфики атаки TCP Flooding необходимо обойти стандартный сценарий работы TCP-протокола, а именно, трехэтапное рукопожатие. Но для HTTP Flooding TCP-протокол должен функционировать нормально. Поэтому модули httpApp и floodApp разделены, но все настройки у них одинаковые, кроме *SpecialP* (8).

Разработанная система также может быть развернута и на семействе операционных систем Windows. На данный момент система моделирования работает в однопоточном режиме. Загрузка процессора при проведении экспериментов составляет 50% мощности одного ядра. При этом используется более 200 виртуальных ПК, атакующих с задержкой в 10 мс.

Во время проведения экспериментов система может записывать логи трафика, которые в дальнейшем можно будет использовать для анализа работы механизмов атаки и защиты.

Созданная виртуальная сеть была верифицирована относительно реальной сети. Она была построена с помощью проекта PlanetLab [11]. В системе была создана виртуальная сеть, соответствующая сети PlanetLab.

Эксперименты

Перед тем как описывать проведенные эксперименты, необходимо указать характеристики и настройки реального сервера, который использовался при проведении экспериментов. Характеристики сервера: Процессор: Intel Core i5-4440S 2.8 Ghz * 4 ядра; RAM: 8 GB DDR3; ОС: Ubuntu 14.04.1 LTS; Версия Apache: Apache/2.4.7; Версия PHP: PHP 5.5.9-1.

Топология сети для выполнения экспериментов состояла из 7 маршрутизаторов, 204 клиентов и 1 реального сервера. Задержки между узлами сети равны 1-й микросекунде.

Эксперимент со сценарием HTTP Flooding. В атаке участвовало 200 хостов. Каждый хост инициирует TCP соединение с HTTP-запросом с частотой 2 пакета в секунду, на сервере запросы обрабатываются с помощью скрипта PHP, что увеличивает нагрузку на сервер. Из-за возрастающей нагрузки время ответа увеличивается с увеличением количества хостов. Исследуемый хост участвует в атаке на протяжении всего эксперимента, остальные хосты включаются в атаку распределенно по времени с 10 до 40 с и прекращают генерацию пакетов с 70 до 90 с.

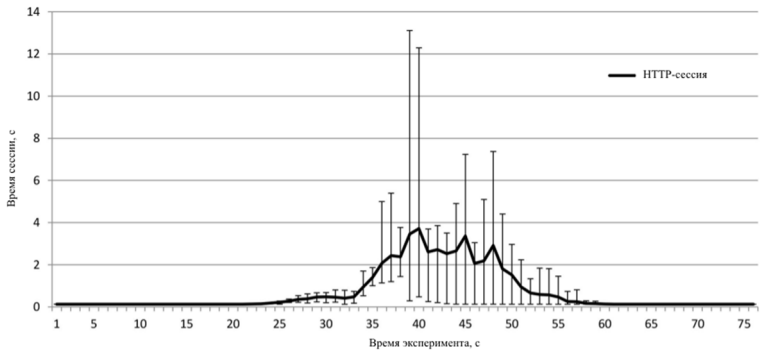

Рис. 2. График результата эксперимента с HTTP-атакой

На графике (рис. 2) показана зависимость длительности сессии от ее порядкового номера. Линией показано среднее значение для 10 экспериментов, а также представле-

ны планки погрешностей. Как видно, во время атаки возрастает нагрузка на сервер, что увеличивает время обработки запроса.

Эксперименты со сценарием SYN Flooding с использованием Egress Filtering. Egress Filtering – это один из наиболее популярных методов защиты от DDoS-атак. Он устанавливается не на исходящие маршрутизаторы, например, провайдером сети для того, чтобы за пределы сети не выходили пакеты с IP-адресами, которые не совпадают ни с одним существующим IP-адресом в этой сети. Была проведена серия экспериментов с использованием фильтра на 2 – 4 маршрутизаторах в виртуальной сети. Для успешной атаки SYN Flooding на сервере были отключены SYN cookies и включена подмена IP-адресов отправителя. Три клиента также атаковали с начала и до конца эксперимента (рис. 3).

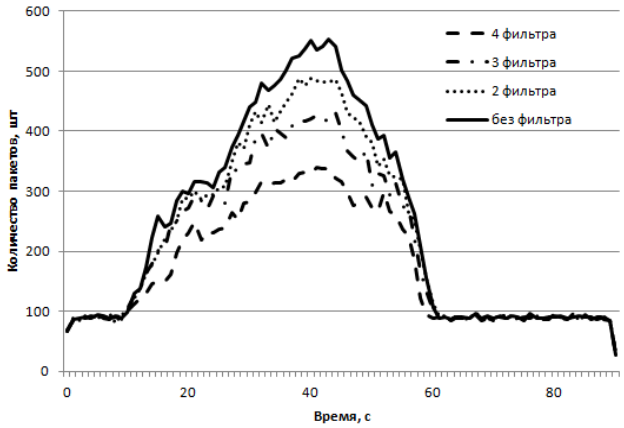

Рис. 3. График результата экспериментов с методом защиты Egress Filtering

Для подведения итогов экспериментов с фильтрами необходимо знать, что в топологии всего было 6 маршрутизаторов, в локальной сети которых были ПК. Соответственно эксперименты были проведены и без использования фильтров, и с использованием фильтров на 2, 3, 4 маршрутизаторах. При этом клиенты, атакующие сервер, постоянно находились в локальной сети маршрутизатора, не использующего Egress Filtering. С увеличением количества фильтров наблюдается уменьшение мощности атаки на сервер.

На основании приведенных экспериментов можно заключить, что система при верной установке работает исправно, не требует большого количества времени для настроек топологий и сценариев атаки, а также позволяет встраивать методы защиты и тестировать их работу.

Заключение

В работе дано формальное описание компонентов системы. Представлена архитектура системы моделирования и ее отдельных компонентов. Созданные компоненты и система в целом были протестированы, верифицированы. Представленные выше эксперименты доказывают, что разработанную систему можно использовать в качестве лаборатории по созданию и тестированию механизмов защиты от DDoS атак. При этом настройка топологий, клиентов, маршрутизаторов, сценариев атак в системе занимает небольшое количество времени.

Разработанная система может быть использована для исследования DDoS-атак, а также методов защиты от них. Также администраторы сети могут быстро и точно воспроизвести обслуживаемую ими сеть, выполнить нагрузочные работы, протестировать

возможности сервера, пропускную способность сети, а также качество работы механизмов защиты.

В дальнейшем авторы планируют использовать систему для разработки и тестирования методов защиты на основе интеллектуального анализа данных.

СПИСОК ЛИТЕРАТУРЫ:

1. CyberArk, Security for the Heart of Enterprise. http://www.cyberark.com/ (дата обращения: 26.06.2015).
2. Bekeneva Ya., Shipilov N., Borisenko K., Shorov A. Simulation of DDoS-attacks and protection mechanisms against them // Proceedings of the 2015 IEEE North West Russia Section Young Researches in Electrical and Electronic Engineering Conference, February 2-4 2015, Russia, St-Petersburg, p. 50-56.
3. Wang J., Phan R. C.-W., Whitley J. N., Parish D. J. Advanced DDoS Attacks Traffic Simulation with a Test Center Platform // International Journal for Information Security Research (IJISR). 2011. Vol. 1(4). P. 169-177.
4. Spirent TestCenter. http://www.spirent.com/Ethernet_Testing/Software/TestCenter (дата обращения: 26.06.2015).
5. Butler B. Interop network squares off against controlled 70G bit/sec DDoS attack. http://www.networkworld.com/article/2166091/data-center/interop-network-squares-off-against-controlled-70g-bit-sec-ddos-attack.html (дата обращения: 26.06.2015).
6. Ixia. http://www.ixiacom.com/about-us/company (дата обращения: 26.06.2015).
7. Котенко И.В., Коновалов А.М., Шоров А.В. Исследовательское моделирование бот-сетей и механизмов защиты от них // Приложение к журналу «Информационные технологии». М.: Новые технологии, 2012, № 1. 32 с.
8. OMNeT++, Discrete Event System Simulator. http://www.omnetpp.org/intro (дата обращения: 26.06.2015).
9. INET Framework. http://inet.omnetpp.org/ (дата обращения: 26.06.2015).
10. ReaSE, Realistic Simulation Environments for OMNeT++. https://i72projekte.tm.uka.de/trac/ReaSE (дата обращения: 26.06.2015).
11. About PlanetLabEurope. https://www.planet-lab.eu/about (дата обращения: 26.06.2015).

REFERENCES:

1. CyberArk, Security for the Heart of Enterprise. http://www.cyberark.com/ (дата обращения: 26.06.2015).
2. Bekeneva Ya., Shipilov N., Borisenko K., Shorov A. Simulation of DDoS-attacks and protection mechanisms against them // Proceedings of the 2015 IEEE North West Russia Section Young Researches in Electrical and Electronic Engineering Conference, February 2-4 2015, Russia, St-Petersburg, p. 50-56.
3. Wang J., Phan R. C.-W., Whitley J. N., Parish D. J. Advanced DDoS Attacks Traffic Simulation with a Test Center Platform // International Journal for Information Security Research (IJISR). 2011. Vol. 1(4). P. 169-177.
4. Spirent TestCenter. http://www.spirent.com/Ethernet_Testing/Software/TestCenter (дата обращения: 26.06.2015).
5. Butler B. Interop network squares off against controlled 70G bit/sec DDoS attack. http://www.networkworld.com/article/2166091/data-center/interop-network-squares-off-against-controlled-70g-bit-sec-ddos-attack.html (дата обращения: 26.06.2015).
5. Ixia. http://www.ixiacom.com/about-us/company (дата обращения: 26.06.2015).
7. Kotenko I.V., Konovalov A.M., Shorov A.V. Issledovatel'skoe modelirovanie bot-setej i mekhanizmov zashhity ot nikh // Prilozhenie k zhurnalu «Informatsionnye tekhnologii». M.: Novye tekhnologii, 2012, № 1. 32 c.
8. OMNeT++, Discrete Event System Simulator. http://www.omnetpp.org/intro (дата обращения: 26.06.2015).
9. INET Framework. http://inet.omnetpp.org/ (дата обращения: 26.06.2015).
10. ReaSE, Realistic Simulation Environments for OMNeT++. https://i72projekte.tm.uka.de/trac/ReaSE (дата обращения: 26.06.2015).
11. About PlanetLabEurope. https://www.planet-lab.eu/about (дата обращения: 26.06.2015).