Моделирование противоборства программных агентов в Интернете: общий подход, среда моделирования и эксперименты

И. В. Котенко, *д. т. н., профессор, руководитель научно-исследовательской группы компьютерной безопасности СПИИРАН*
ivkote@comsec.spb.ru
А. В. Уланов, *аспирант СПИИРАН*
ulanov@comsec.spb.ru

Окончание. Начало см. в № 4 2006.

Среда для агентно-ориентированного моделирования

Так как все моделируемые процессы происходят в сети Интернет, в основе среды моделирования должна быть модель этой сети. Для выбора инструментария моделирования сети и процессов передачи информации был проведен анализ различных пакетов моделирования (Network simulators), включая NS2 [51], OMNeT++ INET Framework [52], SSF Net [53], J-Sim INET Framework [54] и ряда других.

В качестве *основных требований, которые предъявлялись к используемому инструментарию моделирования,* были выбраны следующие [55]:

- детальная реализация протоколов, которые задействованы в атаках DDoS, начиная от сетевого уровня, чтобы была возможность моделирования известных атак DDoS;
- возможность написания и подключения собственных модулей для реализации агентского подхода;
- возможность изменения параметров моделирования во время проведения экспериментов;
- реализация для Windows или Linux (либо независимость платформы);
- развитый графический интерфейс;
- бесплатность при использовании в исследовательских целях.

Проведенный анализ показал, что этим требованиям в наилучшей степени удовлетворяет *OMNeT++ INET Framework* [52]. Система OMNeT++ представляет собой инструментарий моделирования дискретных событий. Изменение состояния моделируемой системы происходит в дискретные моменты времени по списку будущих событий (future event list), отсортированных по времени. Событием может быть: начало передачи пакета, тайм-аут и т. п. События происходят на основе выполнения простых модулей (simple module). У такого модуля есть функции инициализации, обработки сообщения, действия и завершения работы. Обмен сообщениями между модулями осуществляется по каналам (channel), с которыми модули соединены своими шлюзами (gate), или непосредственно через шлюзы. Шлюз может быть входящим или исходящим, соответственно, для приема и посылки сообщений.

На основе INET Framework разрабатывается среда для многоагентного моделирования сетевых атак и механизмов защиты от них [55, 56]. *Архитектура предлагаемой среды* моделирования состоит из четырех основных компонентов:

- базовой системы имитационного моделирования (OMNeT++ Framework);

- пакета для моделирования сети Интернет (INET Framework);
- системы многоагентного моделирования (Multi-agent Framework);
- библиотеки с набором атак и механизмов защиты (библиотека DDoS Framework).

OMNeT++ Framework представляет собой систему моделирования на основе дискретных событий. OMNeT++ INET Framework – комплект модулей для OMNeT++, позволяющих реалистично моделировать узлы и протоколы сети Интернет. Multi-agent Framework и DDoS Framework – разработанный авторами комплект модулей, позволяющих моделировать механизмы атак DDoS и защиты на основе противоборства команд интеллектуальных агентов.

В реализованной к настоящему времени версии среды система OMNeT++ INET Framework подверглась множеству различных модификаций. Например, были созданы таблица фильтрации пакетов на сетевом уровне для моделирования действий агентов защиты и модуль, позволяющий просматривать весь трафик данного узла для ведения статистики, а также для моделирования действий агентов защиты. Подверглись изменению модули, отвечающие за работу Sockets для моделирования атак и механизмов защиты.

Агенты атаки и защиты были реализованы в виде сложных модулей. Они содержат простые модули, отвечающие за работу по различным сетевым протоколам, и ядро агента. Ядро агента служит для управления остальными модулями. Агент, как сложный модуль, имеет ряд шлюзов для подключения к стандартному сетевому узлу из INET Framework. Эти шлюзы относятся к соответствующим сетевым протоколам. Подключение или установка агента может происходить во время проведения моделирования.

OMNeT++ предоставляет альтернативу для реализации логики работы модулей: обработка сообщений модулем или описание действий модуля (activity). В первом случае действия модуля «завязаны» на приход

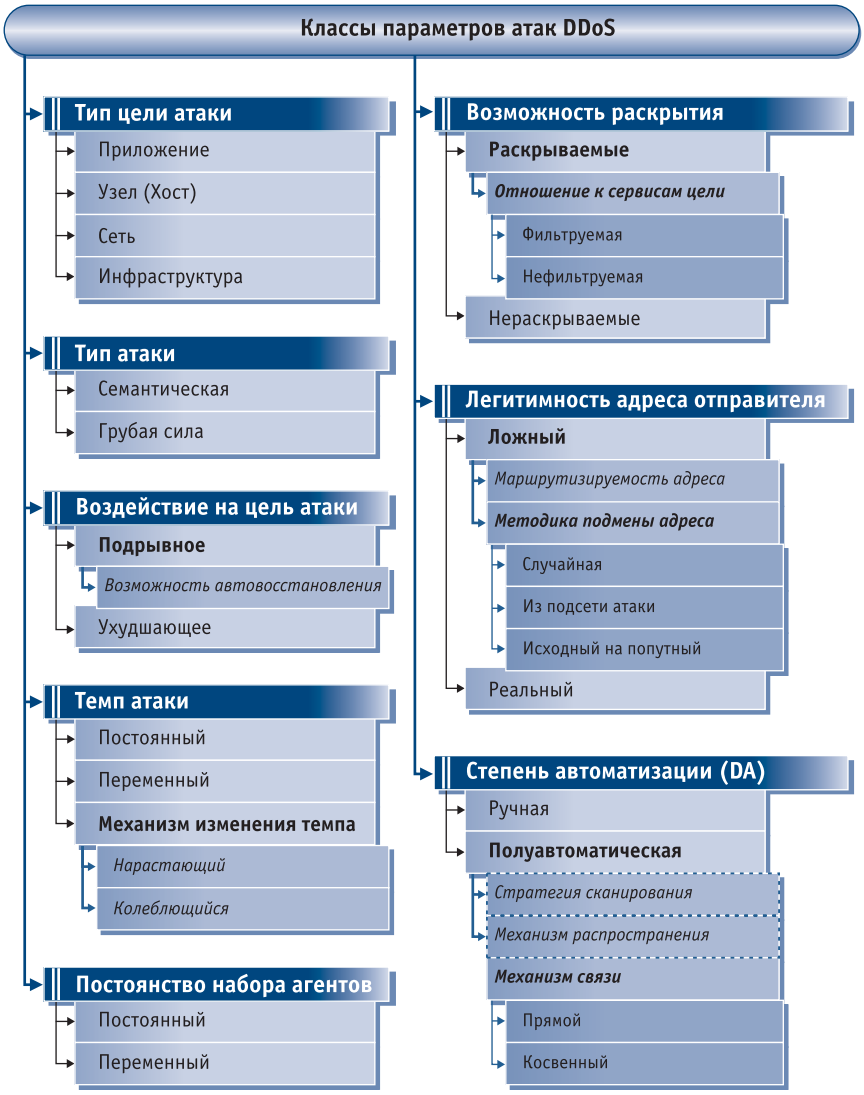


Рис. 8. ***Основные классы параметров атак DDoS***

сообщений. Следующее событие может произойти только после завершения работы функции-обработчика сообщений. Во втором – действия модуля выполняются как сопрограммы (co-routine). Они допускают произвольное ветвление в другие контексты управления и произвольное возобновление потока из точки ветвления. Дополнительно существует возможность описания действий модуля в виде машин состояний.

Ядра агентов были выполнены на основе сопрограмм, так как это удобно для реализации протоколов взаимодействия, положенных в основу командной работы агентов. Остальные модули реализованы как обработчики сообщений от ядра и внешней среды. От машин состояний пришлось отказаться из-за того, что они усложняют читаемость кода и в ряде случаев делают логику работы неявной. Однако этого недостатка можно было бы избежать, если бы использовался графический редактор – менеджер машин состояний (как это, например, сделано в MASDK [57]).

С использованием предлагаемой среды можно моделировать различные типы атак DDoS и механизмов защиты. Рассмотрим *типы и значения параметров, которые можно изменять в процессе экспериментов.*

При моделировании возможно использование следующих *основных классов и значений параметров атак DDoS* (рис. 8).

- *Тип цели атаки.* Выбирается приложение, узел или сеть. Необходимо указать IP-адрес и порт цели атаки.
- *Тип атаки:* грубая сила (UDP/ICMP flood, smurf/fraggle и др.) или семантическая (TCP SYN,

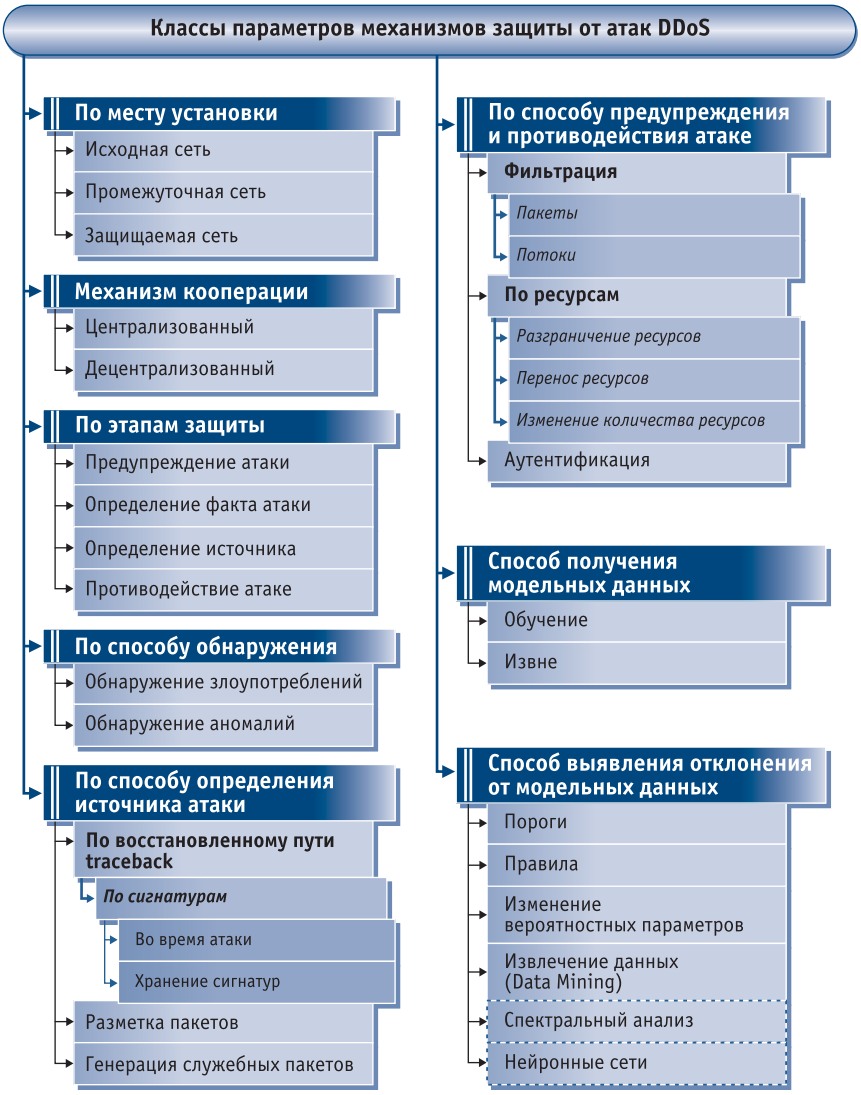


Рис. 9. ***Основные классы параметров механизмов защиты от атак DDoS***

Incorrect packets, Hard requests и др.).

- *Воздействие на цель атаки.* Можно выбрать подрывную атаку, когда все демоны атакуют одновременно, или ухудшающую, когда постепенно в атаку включаются все большее их число. В первом случае атаку легче обнаружить.
- *Темп атаки.* Темп атаки может быть постоянным или переменным. В последнем случае интенсивность атаки меняется во времени. Демонам задается функция изменения частоты посылки пакетов атаки от времени. Изменение может быть нарастающим (демоны в каждый следующий момент времени посылают все больше и больше пакетов атаки) и колеблющимся.
- *Постоянство набора агентов.* Набор может быть постоянным (в атаке участвуют все демоны) и переменным. В последнем случае мастер делит всех демонов на несколько групп, и каждая из них атакует попеременно.
- *Возможность раскрытия.* Атака может быть раскрываемой, когда можно выявить пакеты атаки. Выбирается отношение к сервисам цели атаки. Нефильтруемые пакеты атаки составляются так, чтобы не отличались от легитимных (обычное использование предоставляемого целью атаки сервиса). Фильтруемые пакеты атаки легко выявляются по следующим признакам: значения полей, размер, используемый протокол.
- *Легитимность адреса отправителя.* При посылке пакетов атаки используемый адрес может быть реальным или ложным. Он заменяется на случайно выбранный адрес или адрес из той же подсети, что и демон. Используемый адрес может быть маршрутизируемым или нет.
- *Степень автоматизации.* Атака может выполняться полностью автоматически после задания ее параметров или под контролем злоумышленника. В этом случае на любом этапе атаки он может вмешаться и изменить какой-либо параметр, например, интенсивность атаки. Механизм связи между демонами и мастером может быть прямым (мастер знает адреса всех демонов) и косвенным (все агенты обмениваются сообщениями через один сервер).

При моделировании возможно использование следующих *основных классов и значений параметров механизмов защиты от атак DDoS* (рис. 9).

- *Место установки.* Выбирается место установки механизма защиты: исходная, промежуточная и (или) защищаемая сети.
- *Степень кооперации.* Механизм управления работой отдельных компонентов защиты может быть централизованным или децентрализованным. В последнем случае компоненты защиты являются автономными, но могут объединять свои усилия.
- *Используемые этапы защиты.* Выбираются этапы (механизмы) защиты, которые должен охватывать предлагаемый метод: (1) предупреждение атаки, (2) обнаружение факта атаки, (3) определение источника атаки, (4) противодействие атаке.
- *Способ обнаружения атаки* (если применимо). Обнаружение может происходить по злоупотреблениям или по аномалиям. Задается конкретный метод обнаружения или их множество, например, Hop-count Filtering (HCF), Source IP address monitoring (SIPM), Bit Per Seconds (BPS) и т. п.
- *Способ определения источника атаки* (если применимо). Определение (или отслеживание – traceback) источника атаки возможно по сигнатурам пакетов, с помощью разметки пакетов, генерации служебных пакетов и др.

- *Способ предупреждения атаки* (если применимо). Возможно применение фильтрации (пакетов или потоков), управления ресурсами (разграничение, изменение количества, перенос) и аутентификации.
- *Способ противодействия атаке* (если применимо). Также возможно применение фильтрации (пакетов или потоков), управления ресурсами (разграничение, изменение количества, перенос) и аутентификации.
- *Способ получения модельных данных* (если применимо). Формирование при обучении или из внешних источников.
- *Способ выявления отклонения от модельных данных* (если применимо). Возможно использование порогов, правил (для пакетов и соединений), выявление изменений в вероятностных параметрах трафика, извлечение данных из статистики по трафику и пр. (в зависимости от метода защиты).

Основные элементы среды моделирования. Архитектура агентов

Пример многооконного пользовательского интерфейса среды моделирования показан на рис. 10. На основном окне визуализации (вверху справа) отображается компьютерная сеть для проведения моделирования. Окно управления процессом моделирования (внизу справа) позволяет просматривать и менять параметры моделирования. Более детальное представление этого окна отображено на рис. 11. В данном окне на шкале времени можно наблюдать события, значимые для понимания атак и механизмов защиты. Шкала времени отображается над окном с текстовым описанием событий. На рис. 11 можно видеть, например, события посылки пакета ACK, действие сенсора, инициирование атаки и др.

Для отображения текущего состояния команд агентов служат соответствующие окна состояний (рис. 10, сверху посередине). Можно открывать различные окна, характеризующие функционирование (статистические данные) отдельных хостов, протоколов и агентов. Так, на рис. 10 (слева) отображено несколько окон, характеризующих в графическом и текстовом виде (в том числе в форме графика зависимости количества переданных битов от времени) функционирование одного из хостов.

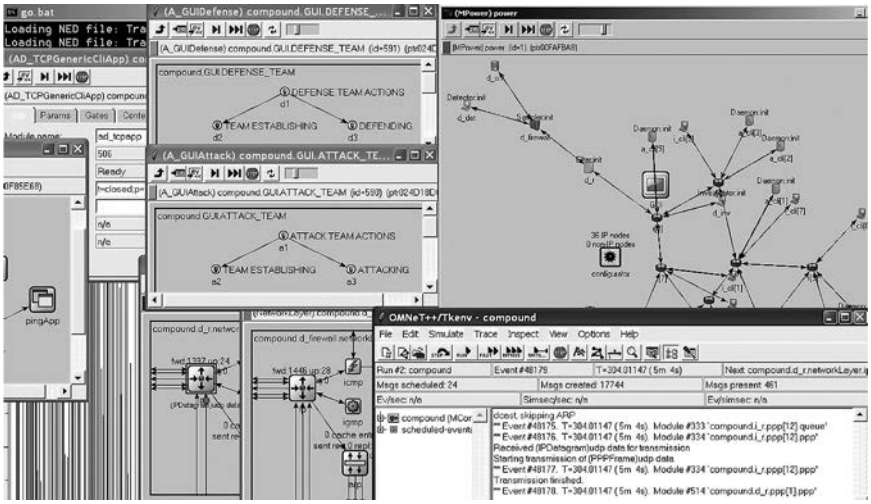


Рис. 10. ***Пример пользовательского интерфейса среды моделирования***

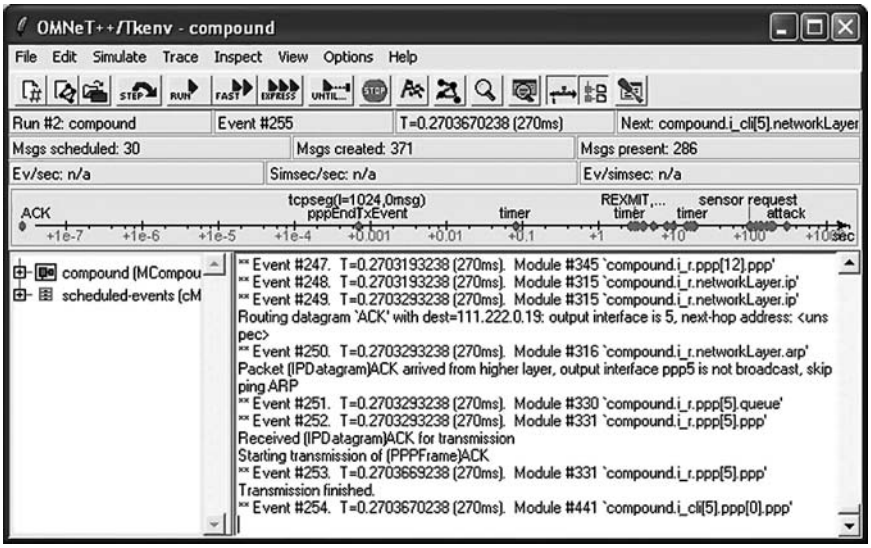


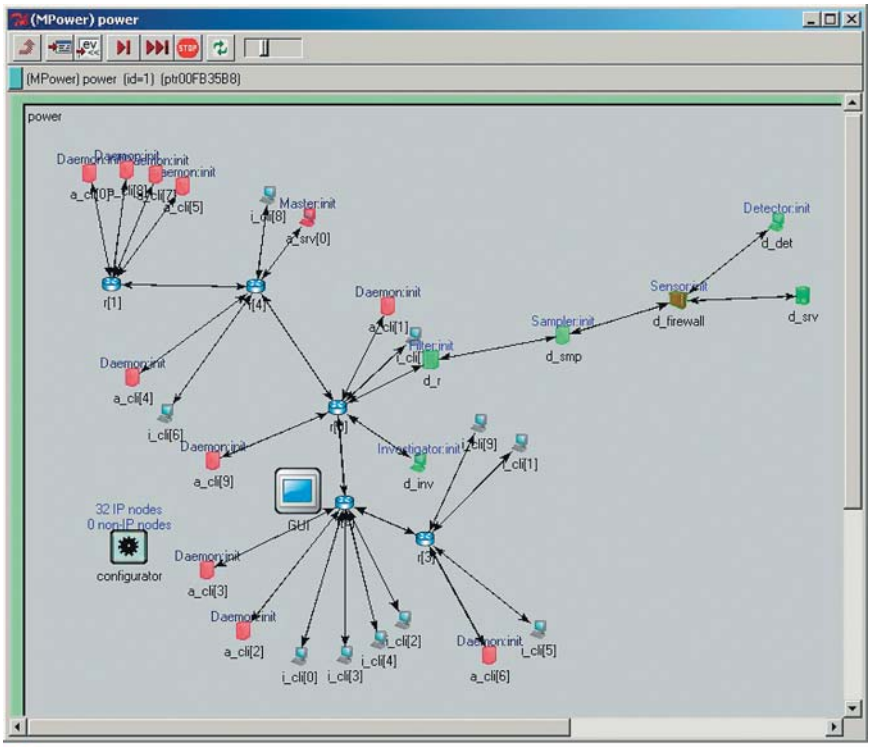
Рис. 11. ***Окно управления процессом моделирования***

Пример основного окна, на котором отображается компьютерная сеть для проведения моделирования, показан на рис. 12. Исследуемая компьютерная сеть представляет собой набор узлов, соединенных каналами связи. Узлы могут выполнять различную функциональность в зависимости от их параметров или набора внутренних модулей.

Узлы сети соединяются между собой каналами связи, параметры которых можно изменять. Примеры базовых параметров: delay (задержка распространения сигнала), datarate (скорость передачи данных).

Стандартный узел сети (рис. 13) состоит из следующих модулей:

- ppp – отвечает за канальный уровень (у маршрутизатора может быть несколько таких модулей – по количеству интерфейсов);
- networkLayer – отвечает за сетевой уровень;
- pingApp – отвечает за приложения, связанные с протоколом ICMP;
- tcp – модуль, обслуживающий протокол TCP;
- udp – модуль, обслуживающий протокол UDP;
- tcpApp[0] – приложение TCP (таких модулей может быть несколько);
- notificationBoard – модуль для регистрации событий, происходящих на узле;



– *маршрутизаторы.*

Красным подсвечены узлы, на которых располагаются агенты команды атаки, зеленым – узлы, на которых находятся агенты команды защиты. Надписи над окрашенными узлами говорят о том, какой агент установлен и в каком состоянии он находится. Остальные узлы – типовые, создающие стандартный трафик сети.

Рис. 12. ***Пример компьютерной сети для проведения моделирования***

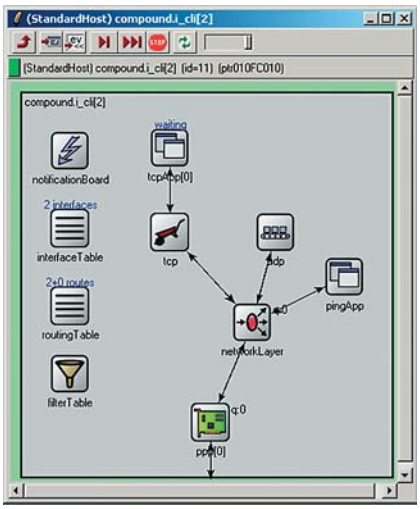


Рис. 13. ***Типовой узел сети***

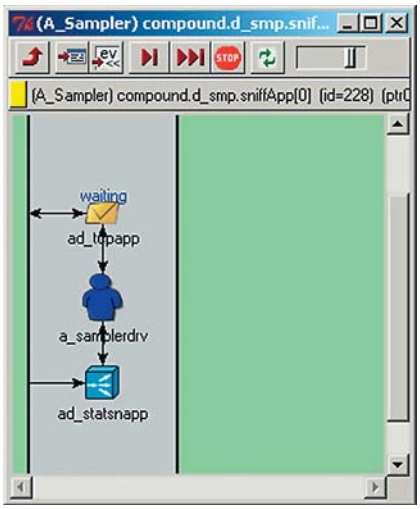


Рис. 14. ***Общее представление структуры агента «сэмплер»***

- interfaceTable – содержит таблицу сетевых интерфейсов;
- routingTable – содержит таблицу маршрутизации;
- filterTable – содержит таблицу правил фильтрации.

Приложения (в том числе и агенты) устанавливаются на узлы, подключаясь к соответствующим модулям протоколов.

При проектировании и реализации агентов были использованы элементы абстрактной архитектуры FIPA [58]. Основная идея такого представления заключается в обеспечении взаимодействия агентов и возможности их повторного использования. Такое описание позволяет увидеть взаимосвязи между основными элементами многоагентной системы.

Для агентов в разрабатываемой системе были использованы следующие элементы абстрактной архитектуры: язык коммуникаций, транспортный и сетевой уровни, каталог агентов. Для всех агентов была необходима реализация языка взаимодействия и транспортного уровня для передачи сообщений. Агентам «мастер» и «детектор», координирующим работу агентов в своих командах, нужен также каталог агентов. Демону необходима реализация двух транспортных модулей: для осуществления коммуникаций и для атаки. Агент фильтрации не может обойтись без реализации сетевого уровня для возможности применения правил фильтрации. Агентам «сенсор» и «сэмплер» также необходим сетевой уровень для обработки и сбора данных в целях построения модели нормального трафика.

Агенты устанавливаются в среду моделирования с помощью подключения к транспортному и сетевому уровням OMNeT++ INET Framework. На рис. 14 изображено обобщенное представление структуры агента «сэмплер». Сэмплер включает в себя транспортный уровень (изображен в виде сообщения), необходимый для коммуникации с другими агентами, сетевой уровень (изображен в виде голубого кубика) для сбора данных по трафику и ядро агента (представлен в виде синего образа фигурки человека). Последнее включает язык коммуникаций, базу знаний и обработчики сообщений от соседних модулей. Представление установки агента «сэмплер» в среду моделирования показано на рис. 15. Видно, что агент подключается к узлу сети через модуль tcp, обслуживающий протокол TCP, и модуль sniffer анализатора сетевых пакетов.

Сеть для многоагентного моделирования состоит из трех подсетей:

- подсеть защиты, где расположена команда защиты;
- промежуточная подсеть, в которой расположены узлы, создающие типовой трафик в сети, в том числе к защищаемому узлу;
- подсеть атаки, где расположена команда атаки.

Подсеть защиты (рис. 16, внизу) состоит из 4 узлов, на которых установлены 4 агента (детектор, сенсор,

фильтр, агент расследования), и защищаемого узла, на котором установлен атакуемый сервер. Агенты и сервер представляют собой приложения, функционирующие на соответствующих узлах. IP-адреса узлов выдаются автоматически. Остальные параметры приложений необходимо задать перед моделированием.

Сервер установлен на узле d_srv. Задается порт для взаимодействия и время задержки ответа клиентам. Сервер входит в INET Framework. Детектор размещен на узле d_det. Назначается адрес защищаемого сервера, порт для командного взаимодействия, интервал опроса сенсоров, максимально допустимая скорость передачи данных к серверу (BPS). Сенсор расположен на узле d_firewall (на входе в подсеть сервера). Задается собственный порт, IP-адрес и порт детектора для командного взаимодействия. Фильтр размещен на узле d_r (маршрутизатор). Назначается собственный порт, IP-адрес и порт детектора для командного взаимодействия. Агент расследования установлен на узле d_inv (во внешней сети). Задается собственный порт, IP-адрес и порт детектора для командного взаимодействия.

Промежуточная подсеть (рис. 16, в центре) состоит из N узлов i_cli[…] с типовыми клиентами, соединенных с маршрутизатором i_r. Клиент входит в INET Framework. Количество узлов N определяется параметром моделирования. Задаются следующие параметры клиентов: адрес и порт сервера, время начала работы, количество и размер запросов при соединении с сервером, размер ответа, время подготовки ответа и интервал бездействия.

Подсеть атаки (рис. 16, вверху) включает M узлов i_cli[…], на которых размещаются демоны и один узел с мастером. Они соединены с маршрутизатором i_r. Количество узлов M задается параметром моделирования. Мастер имеет следующие параметры: порт для командного взаимодействия, адрес и порт цели атаки, время начала атаки, ее интенсивность (пакетов в секунду). Для демона задаются собственный порт, IP-адрес и порт мастера для командного взаимодействия.

Пример сценария моделирования

На примере моделирования процессов реализации распределенных атак «отказ в обслуживании» и механизмов защиты от них проведен ряд экспериментов по имитации противоборства в сети Интернет. Для демонстрации предлагаемого подхода и возможностей разработанной среды моделирования рассмотрим пример одного из наиболее простых сценариев моделирования [55, 56]. В данном эксперименте используется ограниченный фрагмент сети, имитируется упрощенное поведение агентов атаки (они не подменяют адреса отправителя в сетевых пакетах), и исследуется механизм обнаружения атак, основанный на выявлении в атакуемой сети превышения порога трафика для входящих потоков, направленных с определенного IP-адреса.

Компьютерная сеть для проведения моделирования изображена на рис. 16. Маршрутизаторы в этой сети соединены между собой волоконно-оптическими каналами связи со

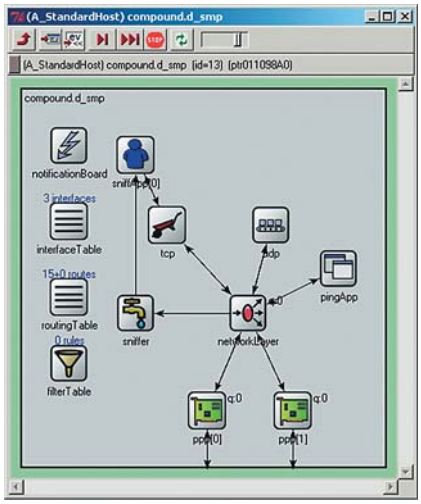


Рис. 15. ***Представление установки агента «сэмплер» в среду моделирования***

скоростью передачи данных 512 Мбит. Остальные узлы соединены каналами связи Ethernet 10 Мбит.

В подсети защиты установлены сервер, детектор, сенсор, фильтр и агент расследования (синие надписи над соответствующими узлами). Сервер на узле d_srv предоставляет сервис по порту 80, задержка ответа – 0. Для детектора защищаемый узел – d_srv, используется порт

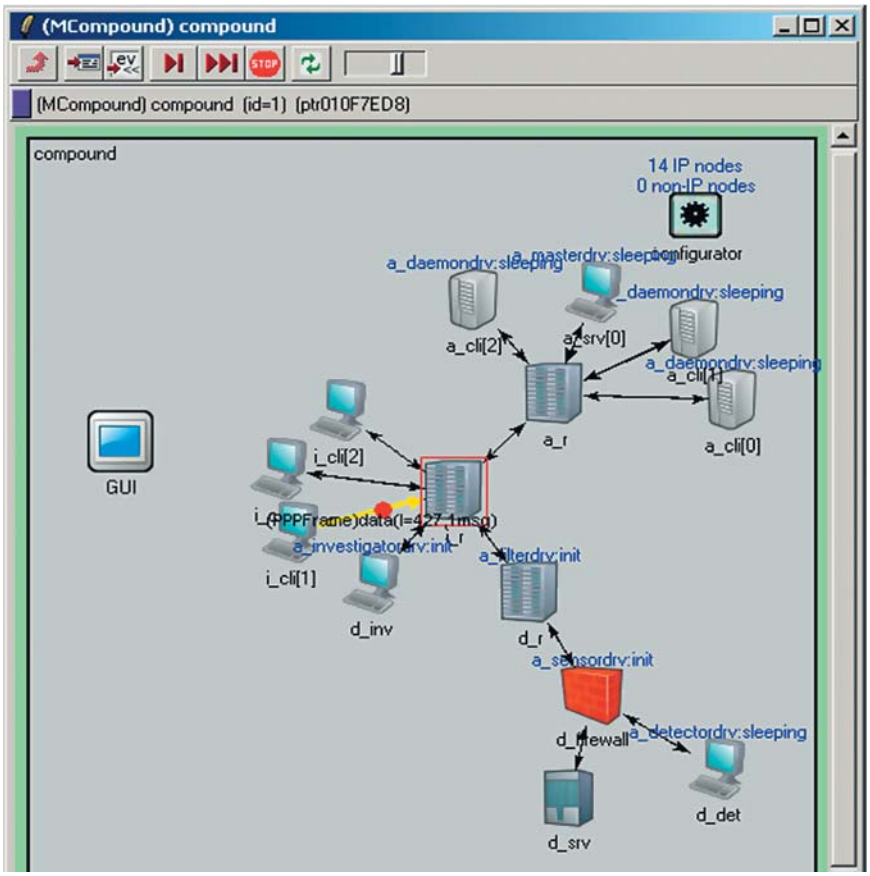


Рис. 16. ***Начальный момент моделирования***

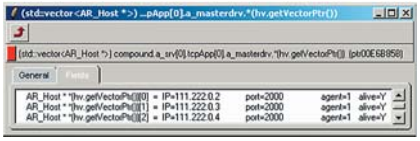


Рис. 17. ***Знания мастера после составления команды***

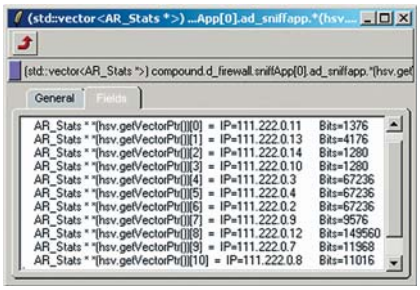


Рис. 18. ***Данные сенсора во время атаки***

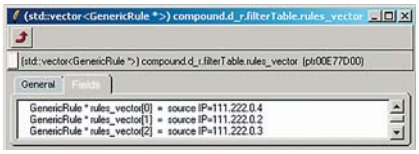


Рис. 19. ***Правила фильтрации, примененные фильтром на узле d_r***

№ 2000, интервал опроса сенсоров – от 10 до 60 секунд, BPS = 1100 бит/с.

Для сенсора, фильтра и агента расследования заданы: порт для взаимодействия – № 2000, адрес и порт детектора – d_det и № 2000 соответственно.

Через некоторое время после запуска моделирования клиенты начинают посылать запросы серверу, а он на них отвечать (рис.16). На рис. 16 красным кружком отмечен пакет, предназначенный серверу. Таким образом происходит генерация нормального сетевого трафика.

Через некоторое время после запуска симуляции начинает формироваться команда защиты. Агенты расследования, сенсор и фильтр соединяются с детектором и посылают ему сообщения о своей работоспособности. Детектор заносит данные о них в память. Формирование команды атаки происходит аналогичным образом. На рис. 17 приведен фрагмент базы знаний мастера после составления команды атаки. На данном фрагменте показано, что мастер знает IP-адреса демонов, порты, по которым можно с ними взаимодействовать, и их состояние (alive = Y означает, что демон находится в работоспособном состоянии).

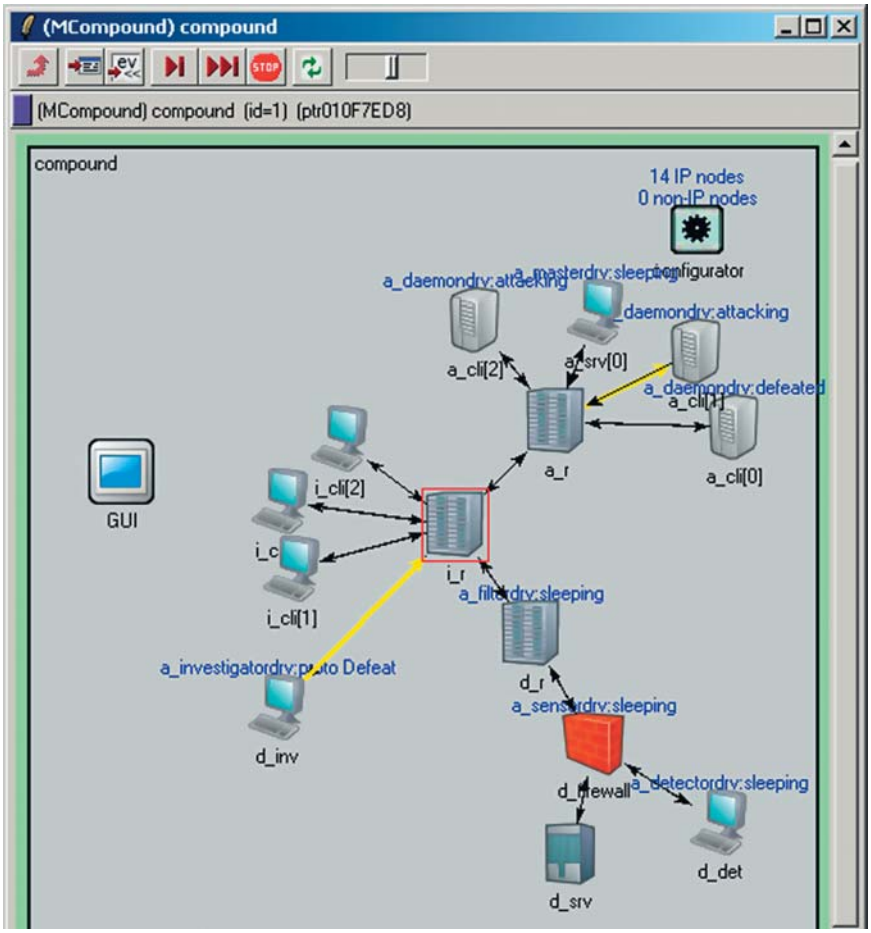


Рис. 20. ***Действия агента расследования во время атаки***

После составления команды защиты, начинаются командные действия. Сенсор выполняет сбор статистики по трафику для каждого адреса (количество переданных битов). Детектор через заданный интервал (например, каждые 60 секунд) опрашивает сенсор, получает от него статистику, определяет, не происходит ли атака. Затем он соединяется с фильтром и агентом расследования и сообщает им IP-адреса подозрительных узлов. Пока атаки не происходит, они бездействуют.

Через определенное время (например, 300 секунд) после начала моделирования команда атаки начинает атакующие действия. Сначала мастер опрашивает всех демонов, выясняя их работоспособность. После проверки оказывается, что все они находятся в рабочем состоянии. Мастер рассчитывает распределение нагрузки. Заданная интенсивность атаки (2 пакета в секунду) делится на количество работоспособных демонов (3), таким образом вычисляется индивидуальная интенсивность атаки каждого демона. После этого мастер отсылает каждому демону команду атаки: адрес цели атаки (d_srv), порт (2000) и интенсивность (0,67). Демоны приступают к атаке.

Спустя примерно 100 секунд происходит очередной запрос детектора к сенсору. В этот момент у сенсора содержится список данных об IP-адресах и количестве переданных битов за последние 60 секунд от различных узлов (рис. 18).

Детектор производит вычисление BPS для каждого узла, кроме сервера (111.222.0.12). Очевидно, что для узлов 111.222.0.4, 111.222.0.3, 111.222.0.2 этот параметр превышает максимально допустимый (1100). Детектор отсылает фильтру эти адреса для фильтрации трафика, а агенту расследования для отслеживания агентов атаки и их нейтрализации. После применения фильтром правил фильтрации (рис. 19) на прохождение пакетов от данных адресов, трафик к серверу снижается.

Агент расследования пытается обезвредить агентов атаки. На рис. 20

видно, что ему удается это сделать в отношении одного из демонов. В основном окне визуализации над этим демоном появляется надпись defeated – поражен. Агент пытается обезвредить еще одного демона. В окне визуализации желтыми стрелками отображается прохождение пакетов от агента расследования к демонам. В результате агенту расследования удается обезвредить двух демонов.

Состояние трафика для сервера на 460-й секунде возвращается к нормальному состоянию, существовавшему до атаки (рис. 21).

Оставшийся демон продолжает атаку. Мастер перераспределил на него нагрузку после выхода из строя остальных. Однако пакеты атаки не доходят до цели, а отбрасываются на входе в защищаемую сеть.

В результате данного эксперимента атака была блокирована через 100 секунд после ее начала (и 400 секунд после начала моделирования), было применено три правила фильтрации и выведено из строя два агента атаки (демона).

График зависимости количества переданных битов в подсеть сервера от времени для узла d_r приведен на рис. 22. В промежутке времени 0–300 секунд основной трафик создавался обращениями клиентов к серверу и его ответами. Этот процесс отмечен вертикальными прямыми с низкой интенсивностью. С момента начала реализации атаки (отметка 300 секунд) появился интенсивный трафик – плато от 300 до 400 секунд. Однако, примерно на 400 секунде моделирования были применены механизмы фильтрации, и пакеты, направленные демонами, стали отбрасываться на входе в сеть сервера. После этого трафик вернулся в нормальное состояние.

Заключение

В статье отражены отдельные результаты работы авторов, связанные с исследованием различных аспектов антагонистического взаимодействия команд агентов в сети Интернет и направленные на разработку формальных моделей противоборства злоумышленников и систем защиты в сети Интернет, а также рекомендаций по построению перспективных систем защиты. Задача многоагентного моделирования процессов кибернетического противоборства представлена как моделирование коэволюционного антагонистического взаимодействия команды агентов-злоумышленников и агентов защиты.

В статье рассмотрен предлагаемый подход к агентно-ориентированному моделированию процессов защиты информации. Подход представлен на примере антагонистического противоборства двух команд агентов: агентов реализации атак DDoS и агентов защиты от данного класса атак. Рассмотрены основные черты подхода, охарактеризованы модели команд агентов, описана разработанная среда моделирования и приведен пример одного из реализованных сценариев моделирования.

Для исследовательского моделирования процессов кибернетического противоборства предложено использовать семейство различных моделей (аналитических, гибридных (аналитико-имитационных), имитационных на уровне сетевых пакетов, полунатурных и натурных). Выбор моделей диктуется, в первую очередь, необходимой точностью и масштабируемостью моделирования.

Проведенные эксперименты показали эффективность предлагаемого подхода и возможность его использования для моделирования перспективных механизмов защиты и анализа защищенности проектируемых сетей.

Направлениями дальнейших исследований является развитие предложенных моделей противоборства, в том числе разработка формальных моделей антагонистического взаимодействия команд агентов защиты и нападения, расширение функциональных возможностей программного прототипа за счет введения дополнительных атак и механизмов защиты от них, теоретическая и экспериментальная оценка эффективности разработанных механизмов защиты, выработка рекомендаций по построению эффективных механизмов защиты от DDoS-атак, дальнейшее развитие среды моделирования,

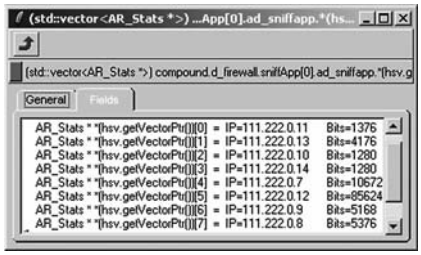


Рис. 21. ***Данные сенсора после применения правил фильтрации***

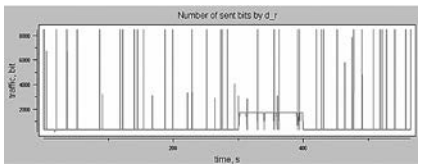


Рис. 22. ***Зависимость количества переданных битов в подсеть сервера от времени для d_r***

исследование и совершенствование механизмов внутрикомандного взаимодействия агентов, а также реализация механизмов адаптации и самообучения агентов. ■

ЛИТЕРАТУРА

1. *Nomad Mobile Research Centre. www.nmrc.org*
2. *Mirkovic J., Dietrich S., Dittrich D., Reiher P. Internet Denial of Service: Attack and Defense Mechanisms. Prentice Hall PTR, 2004.*
3. *Городецкий В. И., Котенко И. В. Концептуальные основы стохастического моделирования в среде Интернет // Труды института системного анализа РАН, том 9: Фундаментальные основы информационных технологий и систем. Под ред. С. В.Емельянова. М.: УРСС, 2005.*
4. *Nwana H. S. Software agents: An overview // Knowledge Engineering Review, № 11(2), 1995.*
5. *Котенко И. В., Карсаев О. И. Использование многоагентных технологий для комплексной защиты информации в компьютерных сетях // Известия ТРТУ. № 4, 2001.*
6. *Gorodetski V., Kotenko I., Karsaev O. Framework for Ontology-based Representation of Distributed Knowledge in Multiagent Network Security System // Proceedings of the 4th World Multi-conference on Systems, Cybernetics and Informatics (SCI-2000), Vol. III: «Virtual Engineering and Emergent Computing». Orlando, USA, July 2000.*
7. *Whittaker G. M. Asymmetric Wargaming: Toward A Game Theoretic Perspective. The MITRE Corporation. September 2000.*
8. *Новиков Д. А., Чхартишвили А. Г. Рефлексивные игры. М.: СИНТЕГ, 2003.*
9. *Чхартишвили А. Г. Теоретико-игровое моделирование информационного управления в активных системах / Человеческий фактор в системах управления. Москва, 2005.*
10. *Городецкий В. И., Котенко И.В. Командная*

работа агентов-хакеров: применение многоагентной технологии для моделирования распределенных атак на компьютерные сети // КИИ-2002. VIII Национальная конференция по искусственному интеллекту с международным участием. Труды конференции. – М.: Физматлит, 2002.

11. Perumalla K. S., Sundaragopalan S. High-Fidelity Modeling of Computer Network Worms // Technical Report GIT-CERCS-04-23. Center for Experimental Research in Computer Science. Georgia Institute of Technology. 2004.

12. Cohen P., Levesque H. J. Teamwork // Nous, 35, 1991.

13. Grosz B., Kraus S. Collaborative Plans for Complex Group Actions // Artificial Intelligence, Vol. 86, 1996.

14. Tambe M. Towards flexible teamwork // Journal of AI Research, Vol. 7, 1997.

15. Jennings N. R. Controlling cooperative problem solving in industrial multi-agent systems using joint intentions // Artificial Intelligence, Vol. 75, No. 2, 1995.

16. Martin D., Cheyer A., Moran D. The open agent architecture: A framework for building distributed software systems // Applied Artificial Intelligence, Vol. 13, No. 1–2, 1999.

17. Yen J., Fan X., Sun S., Wang R., Chen C., Kamali K., Miller M., Volz R. A. On Modeling and Simulating Agent Teamwork in CAST // Proceedings of the Second International Conference on Active Media Technology, 2003.

18. Giampapa J. A., Sycara K. Team-Oriented Agent Coordination in the RETSINA Multi-Agent System // Technical Report CMU-RI-TR-02-34, Robotics Institute, Carnegie Mellon University, 2002.

19. Zachary W. W., Mentec J. L. Modeling and simulating cooperation and teamwork // Military, government, and aerospace simulation, Chinni M. J. (ed.), Vol. 32. 2000.

20. Котенко И. В., Станкевич Л. А. Командная работа агентов в реальном времени // Новости искусственного интеллекта. № 3, 2003.

21. Charniak E., Goldman R. P. A Bayesian Model of Plan recognition // Artificial Intelligence, V.64, No. 1, 1993.

22. Kautz H., Allen J. F. Generalized plan recognition // Proceedings of the Fifth National Conference on Artificial Intelligence, 1986.

23. Vilain M. Getting Serious about Parsing Plans: A Grammatical Analysis of Plan Recognition // Proceedings of the Eighth National Conference on Artificial Intelligence, Cambridge, MA, 1990.

24. Wellman M. P., Pynadath D. V. Plan Recognition under Uncertainty // Unpublished web page, 1997.

25. Goldman R. P., etc. A New Model of Plan Recognition // Proceedings of the 1999 Conference on Uncertainty in Artificial Intelligence, 1999.

26. Geib C. W., Goldman R. P. Plan recognition in intrusion detection systems // DARPA Information Survivability Conference and Exposition, DARPA and the IEEE Computer Society, 2001.

27. Котенко И. В. Распознавание планов агентов-хакеров при обнаружении компьютерных атак // Труды Международных научно-технических конференций «Интеллектуальные системы (IEEE AIS'04)» и «Интеллектуальные САПР (CAD-2004)». М.: Изд-во Физико-математической литературы, 2004.

28. Лефевр В. А. О самоорганизующихся и саморефлексивных системах и их исследовании // Проблемы исследования систем и структур. – М., 1965.

29. Лефевр В. А. Рефлексия. – М., «Когито-Центр», 2003.

30. Лепский В. Е., Рапуто А. Г. Моделирование и поддержка сообществ в Интернет (препринт). – М.: Институт психологии РАН, 1999.

31. Стогний А. А., Кондратьев А. И. Теоретико-игровое информационное моделирование в системах принятия решений. – Киев: Наукова думка, 1986.

32. Дружинин В. В. и др. Введение в теорию конфликта. – М.: Радио и связь, 1989.

33. Gorodetski V., Kotenko I. Attacks against Computer Network: Formal Grammar-based Framework and Simulation Tool // A.Wespi, G.Vigna, L.Deri (Eds.). Recent Advances in Intrusion Detection. Fifth International Symposium. RAID 2002. Zurich, Switzerland. October 2002. Proceedings. Lecture Notes in Computer Science, V.2516, 2002.

34. Csuhaj-Varju E. COLONIES: a multi-agent approach to language generation // A. Kornai, editor, Proceedings of ECAI'96 Workshop on Extended Finite State Models of Language, Budapest, 1996.

35. Kelemen J. Colonies: grammars of reactive systems // I. Plander, editor, Proceedings of AICRS'97. World Scientific, Singapore, 1997.

36. Paun Gh., Salomaa A. (ed.) Grammatical models of multi-agent systems. Gordon and Breach, Amsterdam, 1999.

37. Anchorena S., Cases B. Modeling chaotic series by simple eco-grammar systems with reproduction, death, and maturation // Grammars, 6, 2003.

38. Городецкий В., Котенко И., Карсаев О. Обучение и мета-обучение в многоагентных системах на примере задачи обнаружения вторжений в компьютерных сетях // Труды 4-го международного семинара по прикладной семиотике, семиотическому и интеллектуальному управлению ASC/IC'99. – М.: ПАИМС, 1999.

39. Редько В. Г. Эволюционная кибернетика. Тезисы курса лекций. 1999. www.keldysh.ru/pages/BioCyber/Lectures.html

40. Back T., Fogel D. B., Michalewicz Z. Evolutionary computation. Vol. 1. Basic algorithms and operators. Institute of Physics Publishing. 2000.

41. Back T., Fogel D. B., Michalewicz Z. Evolutionary computation. Vol. 2. Advanced algorithms and operators. Institute of Physics Publishing. 2000.

42. Емельянов В. В., Курейчик В. В., Курейчик В. М. Теория и практика эволюционного моделирования. М.: Физматлит, 2003.

43. Алгулиев Р. М. Методы синтеза адаптивных систем обеспечения информационной безопасности корпоративных сетей. М.: Издательство УРСС, 2001.

44. Gu D., Yang E. Multiagent Reinforcement Learning for Multi-Robot Systems: A Survey // Technical Report of the Department of Computer Science, University of Essex, CSM-404. 2004.

45. Котенко И. В. Многоагентные технологии для анализа уязвимостей и обнаружения вторжений в компьютерных сетях // Новости искусственного интеллекта. № 1, 2004.

46. Kotenko I. Agent-Based Modeling and Simulation of Cyber-Warfare between Malefactors and Security Agents in Internet // 19th European Simulation Multiconference «Simulation in wider Europe». ESM'05. Riga, Latvia, 1–4 June 2005.

47. Kotenko I., Stankevitch L., Akhapkin S. Time-constrained Teamwork // Proceedings of China-Russia Bilateral Conference on Intelligent Information Processing. CRBCIIP-2002. December 5–10, 2002, Beijing, China. p.29–37.

48. Kotenko I., Ulanov A. Multiagent modeling and simulation of agents' competition for network resources availability // Second International Workshop on Safety and Security in Multiagent Systems. Utrecht, The Netherlands. 2005.

49. Jin C., Wang H., Shin K. G. Hop-count filtering: An effective defense against spoofed DDoS traffic // Proceedings of the 10th ACM Conference on Computer and Communications Security, 2003.

50. Peng T., Leckie C., Kotagiri R. Proactively Detecting DDoS Attack Using Source IP Address Monitoring // Networking 2004, Athens, Greece, May, 2004.

51. NS-2 homepage. www.isi.edu/nsnam/ns/

52. OMNeT++ homepage. www.omnetpp.org/

53. SSFNet homepage. www.ssfnet.org

54. J-Sim homepage. www.j-sim.org

55. Kotenko I. V., Ulanov A. V. Agent-based simulation of DDOS attacks and defense mechanisms // Journal of Computing, Vol. 4, Issue 2, 2005.

56. Kotenko I. V., Ulanov A. V. The Software Environment for multi-agent Simulation of Defense Mechanisms against DDoS Attacks // The International Conference on Intelligent Agents, Web Technologies and Internet Commerce. IAWTIC'2005. Vienna, Austria. 2005.

57. Gorodetski V., Karsayev O., Kotenko I., Khabalov A. Software Development Kit for Multi-agent Systems Design and Implementation // LNAI 2296, 2002.

58. FIPA. www.fipa.org