XXXIII Неделя науки СПбГПУ. Материалы межвузовской научно-технической конференции. Ч.V: С.145-146, 2005.

УДК 50.41.00, 50.37.23

О.А.Сумкина (4 курс, каф. ИБКС), В.В.Платонов, к.т.н., проф.

ЗАЩИТА МОБИЛЬНЫХ АГЕНТОВ ОТ ЗЛОНАМЕРЕННОГО ВОЗДЕЙСТВИЯ ХОСТОВ

С развитием сетевых технологий задачи администрирования и защиты сетей усложняются за счет физической невозможности присутствия администратора или постоянного ручного обновления систем защиты в удаленных сегментах подсети. Для решения указанных проблем намечается тенденция использования технологии мобильных агентов (МА).

Распределенные системы, построенные на основе технологии мобильных агентов, содержат активные элементы, способные перемещаться в сети и выполнять определенные действия. Такой подход позволяет во многих случаях уменьшить сетевой трафик, а иногда и повысить защиту самого приложения. Так, например, система обнаружения вторжений на основе мобильных агентов не только обладает активным модулем реакции, способным перемещаться в сегмент сети, подвергшийся нападению, но и является более сложной мишенью для атаки, поскольку ее элементы динамически распределяются по сети.

При всех преимуществах, при использовании мобильных агентов возникают определенные классы угроз безопасности:

- угроза атаки на хост со стороны мобильного агента;
- угроза атаки на мобильного агента со стороны хоста;
- угроза атаки на мобильного агента со стороны других мобильных агентов.

В данной работе были исследованы атаки на мобильных агентов со стороны хоста. В то время как механизмы, позволяющие защитить платформу, выполняющую роль МА, на сегодняшний день вполне реальны, защита самого агента и, следовательно, его владельца все еще остаются предметом для исследований. Для систем защиты, основанных на технологии МА, также характерно, что агент реализует функции защиты, следовательно:

- является доверенной сущностью, воздействие на которую нежелательно;
- требует защиты от чтения, поскольку реализует алгоритмы и методы защиты.

Вследствие невозможности полной защиты агента от атаки со стороны хоста были рассмотрены различные подходы к обнаружению подобных атак. Проведенный анализ показал, что наиболее перспективным оказывается использование повторного исполнения агента с целью обнаружения несоответствия итоговых состояний, вырабатываемых агентом при выполнении на разных хостах. Тем не менее, ни один из существующих подходов к защите агентов по тем или иным причинам не был признан удовлетворительным. В качестве альтернативного решения был предложен новый протокол, с использованием которого можно обнаружить атаки на мобильных агентов, проведение которых влияет на их итоговое состояние. При этом атака не только обнаруживается, но администратору также предоставляются все необходимые данные для доказательства проведения хостом атаки.

В основе протокола лежит принцип сохранения начального и итогового состояния агента, а также входных данных, полученных им в процессе выполнения. Для проверки корректности на каждом хосте проводится повторное выполнение агента с использованием сохраненных данных. При этом гарантируется, что если агента атакует только один хост, то изменение в полученном на этом хосте итоговом состоянии агента будет замечено следующим хостом. Таким образом, атака будет обнаружена и атакованный агент не будет исполнятся на остальных хостах.

Несмотря на то, что предложенный протокол не имеет недостатков, присущих другим существующим методам, он обладает некоторыми собственными ограничениями. В

частности, без дополнительной оптимизации с помощью данного протокола невозможно определить атаки даже двух последовательных хостов. Также при его использовании нарушается конфиденциальность данных, полученных агентом на выполняющем его хосте, так как они необходимы следующему хосту для проверки.

Предложенный протокол примерно удваивает затраты на сессию выполнения на недоверенном хосте, за которым идет еще один недоверенный хост. В работе также описаны способы оптимизации протокола.

В результате выполнения работы был разработан прототип распределенной системы, функциональность которой обеспечивается мобильными агентами. Для защиты агентов в системе была предложена реализация описанного в работе протокола, обеспечивающая адекватное обнаружение атак на агентов со стороны хостов. Для уменьшения затрат на применения протокола было предложено передавать исходное состояние в виде набора элементов, которым оно отличается от полученного хостом итогового состояния. На разработку прототипа системы повлияло требование как можно больше снизить сетевой трафик, что сказалось на схемах передачи сообщений (преимущество отдается по возможности локальному взаимодействию).

На основании построенной архитектуры возможно дальнейшее развитие системы. Предложенная схема формирования и хранения данных, необходимых для проверки итогового состояния агента, позволяет легко расширить протокол для проверки выполнения агента n хостами. Для ускорения этапа проверки можно разбить предаваемые между серверными компонентами данные на несколько частей и передавать в первую очередь те из них, которые необходимы для запуска клона агента на выполнение (то есть исходное состояние и, при необходимости, первые входные параметры). Такое изменение позволить запускать клон на проверку до окончания передачи данных. Можно также добавить в код агента опции отключения вывода (отчасти это было реализовано в виде отмены повторного создания клоном файла input.dat).

Разработка прототипа системы проводилась в рамках изучения распределенных систем обнаружения вторжений, поэтому ее архитектура удовлетворяет принятой для данного типа систем, что облегчает дальнейшее развитие предложенной системы.

В качестве языка разработки системы был выбран язык Java, в качестве реализации мобильных агентов – система IBM Aglets 2.0.2. Подобный выбор объясняется стремлением к использованию открытых систем и технологий, ориентированных на переносимость кода. Поскольку изначально стояла задача разработки систем для разнородных сетей, то проблемы переносимости является одной из главных. Тем не менее, недостатком использования Java является более низкая скорость выполнения, чем при использовании, например, технологии .Net.