А. А. Станкевичус

ПРИМЕНЕНИЕ ВЕРИФИКАЦИИ ДЛЯ ЗАЩИТЫ ГРИД ОТ ВРЕДОНОСНОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Введение

Защита Грид от вредоносного прикладного программного обеспечения (ППО) — важная и актуальная задача, перспективным подходом к решению которой является автоматическая верификация безопасности ППО [1]. Существующие на данный момент верификаторы, в том числе и использующие абстрактную интерпретацию, не позволяют верифицировать безопасность ППО для вычислительной среды Грид. Предложенные в данной работе комплексный метод верификации, абстрактный домен и модель ячейки памяти делают возможным создание автоматического верификатора безопасности ППО Грид.

1. Методика защиты

Для выявления вредоносного ППО автором данной работы разработан синтетический метод верификации, сочетающий несколько разнотипных техник: статический анализ формальных свойств и мониторинг формальных свойств ППО [2].

Статический анализ формальных свойств заключается в проверке свойств автоматически вырабатываемой на основе исходного кода модели. Для построения модели предлагается использовать абстрактную интерпретацию [3] исходного кода программы. Проверять необходимо невозможность нарушения при произвольных входных данных верифицируемым ППО требований безопасности, определенных для ППО Грид. В некоторых случаях абстрактная интерпретация не позволяет достоверно установить безопасность или небезопасность участка ППО. В этих случаях предлагается применять мониторинг формальных свойств ППО. В основу мониторинга формальных свойств ППО положено выделение требующих верификации формальных свойств ППО и встраивание проверки этих свойств в систему мониторинга. При исполнении ППО следует проводить постоянный мониторинг, целью которого является контроль выделенных свойств.

Благодаря таким особенностям вычислительных Грид, как отсутствие необходимости непосредственного взаимодействия ППО с пользователями ЭВМ и локальными ресурсами ЭВМ, а также вычислительный характер решаемых им задач, для ППО вычислительной Грид может быть выработана строгая политика безопасности, ограничивающая перечень допустимых способов взаимодействия с программной и аппаратной платформой.

Многие современные ОС допускают самомодификацию ППО и не гарантируют уничтожения конфиденциальных данных в памяти перед передачей этой памяти в пользование другого ППО. Некоторые программы самостоятельно контролируют очистку используемой памяти, однако в случае аварийного завершения их работы содержимое памяти, как правило, становится общедоступным. В состав политики безопасности предлагается включать модель ячейки памяти, учитывающую особенности конкретной аппаратной и программной платформы, на которой будет производиться исполнение потенциально вредоносного ППО. Модель может учитывать необходимость выделения памяти перед записью или чтением данных, недопустимость повторного освобождения свободной ячейки, необходимость инициализации выделяемой памяти перед чтением и т. д. Применение такой модели при верификации позволяет предотвратить самомодификацию ППО Грид, которая сделала бы недействительными результаты статического анализа, а также выявить попытки доступа к конфиденциальной информации в памяти ЭВМ.

2. Абстрактная интерпретация

В процессе проверки безопасности исходного кода ППО Грид требуется определить значения, принимаемые различными переменными программы без выполнения программы. Это

может быть достигнуто путем абстрактной интерпретации программы. Абстракция заключается в рассмотрении набора вероятных путей исполнения программы, определяемых стандартной семантикой языка ее реализации, с использованием упрощенной абстрактной модели этой семантики [4]. Например, семантика языка Си может быть выражена в терминах доменов целых чисел, чисел с плавающей точкой, символов и т. д., а также через объединения этих доменов и определенные на них непрерывные функции. При создании упрощенной абстрактной модели семантики Си определяются упрощенные абстрактные домены, на которых определяются абстрактные версии операций.

Идея анализа программ путем проведения вычислений с использованием абстрактных значений не нова. Подобная идея применена в [5, 6], в верификаторе Astree [7], предназначенном для выявления ошибок времени выполнения в исходном коде программ на языке Си для встраиваемых систем, не содержащих динамического выделения памяти и рекурсии.

Для обхода ограничений, таких как невозможность абстрактной интерпретации программ, содержащих рекурсивные вызовы и переходы на произвольные метки, связанных с необходимостью определения неподвижных точек циклов, предлагается использовать разработанный автором данной статьи алгоритм верификации, не требующий поиска неподвижных точек, описанный в [8]. Для успешной абстрактной интерпретации побитовых операций может быть применен новый абстрактный домен, введенный в данной работе.

3. Абстрактный домен двоичных n-мерных векторов

Введем абстрактный домен двоичных **n-мерных векторов, построенный как декартово произведение** n абстрактных доменов двоичных переменных (1).

$$X^{Vn} = \{x | x = \{x_1, x_2, ..., x_n\}, \text{ где } x_1 \in \{0, 1, \perp\}\ \forall i \in 1, 2, ..., n\}. \quad (1)$$

Этот домен позволяет проводить абстрактную интерпретацию программ, содержащих такие целочисленные операции, как побитовый сдвиг, побитовое исключающее или, побитовое исключение, побитовое или, побитовое и. Кроме того, этот домен позволяет получать информацию о знаке и порядке величины при осуществлении таких операций, как сложение, вычитание, умножение, деление и взятие остатка от деления. За счет применения этого абстрактного домена становится возможной верификация ППО, содержащего побитовые операции.

4. Модель ячейки памяти

Предлагаемая в работе методика защиты Грид от вредоносного ППО подразумевает проверку корректности работы с динамически выделяемой оперативной памятью, в связи с чем возникает необходимость формального описания ячеек оперативной памяти ЭВМ. На рисунке 1 приведена построенная автором данной работы модель ячейки памяти, требующей инициализации.

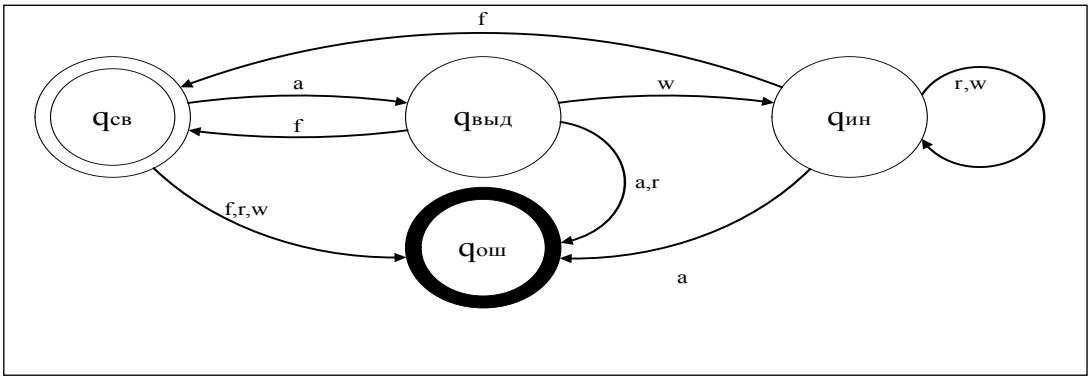

Рис. 1. Модель ячейки памяти

Ячейка может находиться в одном из четырех состояний: $q_{св}$ — ячейка свободна, $q_{выд}$ — ячейка выделена, но не инициализирована, $q_{ин}$ — ячейка выделена и инициализирована, $q_{ош}$ — ошибка.

Начальным состоянием является $q_{св}$. При переходе в состояние $q_{ош}$ происходит остановка работы модели. Переходы в модели помечены как a — выделение, f — освобождение, r — чтение данных из ячейки и w — запись данных в ячейку. Такая модель может использоваться в системах, не обеспечивающих уничтожения данных, хранившихся до выделения в ячейке памяти, применявшейся ранее другими программами для хранения конфиденциальной информации. При статическом анализе формальных свойств и мониторинге формальных свойств ППО Грид необходимо осуществлять контроль соответствия поведения ППО построенной модели.

5. Мониторинг взаимодействия с диспетчером Грид

При взаимодействии ППО Грид с диспетчером Грид необходимо предотвращать реализацию атак «отказ в обслуживании» на диспетчер Грид. Атаку «отказ в обслуживании» следует считать успешно реализованной в случае, когда общий объем передаваемых диспетчеру Грид всеми узлами Грид данных в единицу времени W значительно превышает пропускную способность канала или системы обработки данных диспетчера Грид V. Таким образом, атака «отказ в обслуживании» не может быть реализована в случае, если передача данных узлами Грид производится со скоростью $U < V/N$, где N — общее количество вычислительных узлов Грид. При этом необходимо учитывать общий объем передаваемых диспетчеру Грид данных, включающий заголовки и служебную информацию используемых сетевых протоколов различного уровня.

Мониторинг сетевого взаимодействия может позволить защитить диспетчер Грид от атак «отказ в обслуживании» со стороны вредоносного ППО Грид. Выявление потенциальных атак «отказ в обслуживании» следует производить и на диспетчере Грид, и на вычислительных узлах. Вычислительный узел должен оценивать средний объем данных, производимый ППО Грид в единицу времени, и периодически сообщать его диспетчеру Грид. В случае превышения допустимой средней скорости поступления данных от ППО передача данных должна задерживаться. В свою очередь, диспетчер Грид должен производить оценку среднего объема данных, производимого ППО Грид в единицу времени. В случае если это количество превышает пропускную способность диспетчера Грид, должно производиться завершение работы ППО на вычислительных узлах.

СПИСОК ЛИТЕРАТУРЫ:

1. *Станкевичус А. А.* О некоторых методах защиты Грид от вредоносного кода // Безопасность информационных технологий. 2008. № 3.

2. *Кулямин В. В.* Методы верификации программного обеспечения // Всероссийский конкурсный отбор обзорно-аналитических статей по приоритетному направлению «Информационно-телекоммуникационные системы». 2008.

3. *Jones N. D.* Abstract Interpretation: a Semantics-Based Tool for Program Analysis. DIKU. University of Copenhagen. Denmark, 1994.

4. *Cousot P., Cousot R.* Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints // Conf. Rec. 4th ACM Symp. on Principles of Prog. Lang., POPL'77. 1977. P. 238–252.

5. *Sintzoff M.* Calculating Properties of Programs by Valuations on Specific Models // Proceedings ACM Conference on Proving Assertions about Programs. 1972. P. 203–207.

6. *Hecht M. S.* Flow Analysis of Computer Programs. Elsevier North-Holland, 1977.

7. *Cousot P., Cousot R., Feret J., Mauborgne L., Miné A., Monniaux D., Rival X.* The ASTRÉE analyzer. Programming Languages and Systems // Proceedings of the 14th European Symposium on Programming. Volume 3444 of Lecture Notes in Computer Science. 2005.

8. *Станкевичус А. А.* Инструментальное средство проверки безопасности кода, предназначенного для исполнения в Грид-сетях // Безопасность информационных технологий. 2009. № 1.