Н А У Ч Н О - Т Е Х Н И Ч Е С К И Й

В Е С Т Н И К

П О В О Л Ж Ь Я

№10 2022

Направления:

1.2.2. – МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ ЧИСЛЕННЫЕ МЕТОДЫ И КОМПЛЕКСЫ ПРОГРАММ (технические науки)

2.3.1. – СИСТЕМНЫЙ АНАЛИЗ, УПРАВЛЕНИЕ И ОБРАБОТКА ИНФОРМАЦИИ (технические науки)

2.3.3. – АВТОМАТИЗАЦИЯ И УПРАВЛЕНИЕ ТЕХНОЛОГИЧЕСКИМИ ПРОЦЕССАМИ И ПРОИЗВОДСТВАМИ (технические науки)

2.3.5. – МАТЕМАТИЧЕСКОЕ И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ВЫЧИСЛИТЕЛЬНЫХ МАШИН, КОМПЛЕКСОВ И КОМПЬЮТЕРНЫХ СЕТЕЙ (физико-математические науки)

2.3.5. – МАТЕМАТИЧЕСКОЕ И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ВЫЧИСЛИТЕЛЬНЫХ МАШИН, КОМПЛЕКСОВ И КОМПЬЮТЕРНЫХ СЕТЕЙ (технические науки)

2.3.6. – МЕТОДЫ И СИСТЕМЫ ЗАЩИТЫ ИНФОРМАЦИИ, ИНФОРМАЦИОННАЯ БЕЗОПАСНОСТЬ (физико-математические науки)

Казань
2022

СОДЕРЖАНИЕ

2.3.1.

А.Ю. Унгер

МИРЭА – Российский технологический университет,
институт информационных технологий,
кафедра вычислительной техники,
Москва, unger@mirea.ru

ОЦЕНКА БЕЗОПАСНОСТИ ИНФОРМАЦИОННОЙ СИСТЕМЫ НА ОСНОВЕ ФОРМАЛЬНОЙ ГРАММАТИКИ

В работе предложен метод оценки потенциальной уязвимости информационной системы на базе анализа расхождений деревьев разбора, полученных для одного сообщения на передающей и принимающей стороне. Показано, что многие существующие протоколы обмена допускают неоднозначное толкование одного и того же сообщения, что создает прецедент для атак на систему, используя эту неоднозначность. Показано, что традиционные методы защиты, которые основываются на анализе вводимых данных с помощью конечных автоматов, не всегда являются эффективными. Предложены основные направления развития практических методов дифференциального анализа деревьев разбора.

Ключевые слова: *формальная грамматика, дерево разбора, дифференциальный анализ.*

Введение. Основой построения сложных информационных систем является композиция [1]. Она предполагает, построение сложной системы из более простых компонентов. Модульная архитектура позволяет в значительной степени сократить расходы на разработку системы за счет использования уже готовых компонентов.

Для того, чтобы система, как целое, функционировала, необходимо, чтобы модули, ее составляющие, обменивались информацией. Таким образом, каждый модуль рассматривается как черный ящик, имеющий заданный *интерфейс*. Интерфейс является границей, через которую модуль получает данные. Именно на границе модуль верифицирует данные и, либо принимает их в обработку, либо, если данные не удовлетворяют формату передачи, отвергает их. Границы модулей, таким образом, являются основной целью атак безопасности.

Модули общаются друг с другом посредством сообщений. В работе предлагается рассматривать сообщение, как записанное на некотором формальном языке. Формализация сообщений таким образом позволит проектировать интерфейсы компонентов на базе хорошо разработанной теории формальных грамматик и улучшить безопасность информационной системы в целом.

Основные положения. Многие системы, построенные по модульному принципу, разрабатывались, совершенно не заботясь о безопасности. Можно сказать, что это связано с тем, что разработчики использовали готовые модули, полагаясь на их встроенную безопасность. Готовый модуль может быть безопасен, но при условии тщательного соблюдения протокола передачи данных.

Сформулируем фундаментальное требование к протоколу получения компонентом данных: *получаемое сообщение должно интерпретироваться однозначно*. Данное требование подразумевает, что два одинаковых сообщения должны приводит к одинаковой реакции компонента. Последнее требование может показаться тривиальных, однако, два разных компонента, реализующие один и тот же протокол, могут реагировать на одно и то же сообщение по-разному. Протоколы в Интернете могут быть простыми или сложными, но сложность в данном случае имеет строгое определение в теории формальных языков [2]. Согласно классификации Хомского [3], все языки подразделяются на 4 типа (рисунок 1).

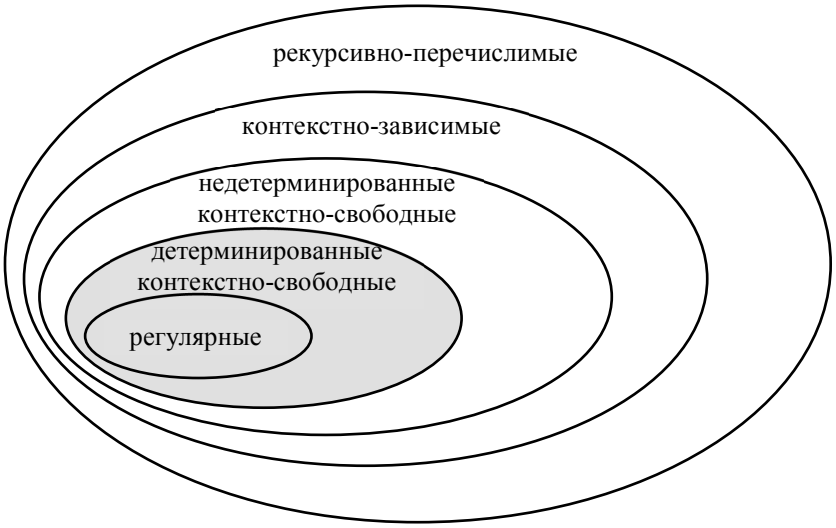

Рис. 1 – Иерархия формальных языков и грамматик

Таким образом, протоколу взаимодействия между компонентами информационной системы соответствует определенный тип языка. Известно [4], что один и тот же язык может быть описан множеством эквивалентных грамматик. Известно также [5], что задача нахождения эквивалентной грамматики для заданного языка является алгоритмически неразрешимой, если грамматика языка более общая чем детерминированная контекстно-свободная.

Рекурсивно-перечислимые языки по определению неразрешимы, что означает, что распознаватель может никогда не остановиться (зациклиться) в случае некорректного ввода. По этой причине использование рекурсивно-перечислимых языков допускает атаки типа отказ в обслуживании (DoS) и является небезопасным.

Менее сложные языки разрешимы всегда, т.е. для них можно построить корректный распознаватель, которые разрешит допустимый ввод и отклонит недопустимый, и при этом не зациклится. Сказанное позволяет сформулировать еще одно положение: *сложность парсера для вводимых в компонент данных, должна соответствовать сложности языка, понимаемого компонентом*.

Любая грамматика, в которой значение любого токена во входной строке может повлиять на структуру другой части строки, является по меньшей мере контекстно-свободной [6]. Из этого, в частности, следует, что что большинство протоколов Интернета необходимо анализировать с помощью КС-парсеров таких, как анализатор *рекурсивного спуска*.

Чем более сложным является анализатор, тем неопределеннее является результат сообщения. В самом деле, теория формальных языков базируется на том, что одному языку L может соответствовать множество эквивалентных грамматик. Семантика любого сообщения M определяется исключительно языком L, однако интерпретация этого сообщения зависит от реализации протокола. Эту ситуацию можно описать на примере двух комплементарных функций: $E(M)$ – кодирования сообщения на стороне отправителя и $D(M)$ – декодирования на стороне получателя. Как уже было сказано, для языков более сложных, чем детерминированные контекстно-свободные, эквивалентность двух грамматик E и D неразрешима. По причине неэквивалентности кодера и декодера $D(E(M)) \neq M$. Это значить, что смысл отправленного сообщения может отличаться от смысла полученного сообщения. Данное обстоятельство создает плацдарм для атак на систему.

Стандартные методы борьбы с атаками заключаются в *экранировании* – превращении входных данных в строковый литерал. Эти методы объединяет то, что они могут усложнить атаку за счет усложнения процедуры *валидации* вводимых данных, однако они не могут устранить принципиальную возможность такой атаки. Рассмотрим другой, более универсальный способ борьбы с подобного рода атаками.

Верификация дерева разбора. Метод, основанный на верификации дерева разбора, подразумевает дублирование распознавателя на стороне отправителя и получателя. Можно

скачать, что ключом к оценке уязвимости системы, использующей метод верификации дерева разбора, является степень различия в реализации распознавателя одного и того же протокола на стороне отправителя и стороне получателя. Несмотря на то, что в обычных условиях эти различия несущественны, и стороны могут понимать два диалекта одного протокольного языка, с точки зрения информационной безопасности это дает злоумышленнику потенциальные возможности для взлома системы. Ниже предлагается подход к оценке уязвимости информационного обмена двумя компонентами системы *A* и *B*.

Пусть на стороне *A* применяется некоторая грамматика *E*, которая кодирует сообщение *M*. На приемной стороне *B* это сообщение распознается грамматикой *D*, таким образом, что сообщение, получаемое стороной *B*, имеет вид *D*(*E*(*M*)). В случае, если грамматики *E* и *D* полностью эквивалентны, то

$$M = D\big(E(M)\big). \quad (1)$$

Однако, если имеет место случай

$$M \neq D\big(E(M)\big), \quad (2)$$

компонент *B* вынужден проделать ряд вычислений, которых в случае (1) не было бы. Таким образом, по расхождению (2) можно оценить нижнюю границу множества потенциальных уязвимостей в процессе коммуникации компонентов *A* и *B*.

$$\delta_A \cup \delta_B, \delta_A = \big|M_A - D_B\big(E_A(M_A)\big)\big|, \delta_B = \big|M_B - D_A\big(E_B(M_B)\big)\big|. \quad (3)$$

Здесь δ_A и δ_B – объемы дополнительных вычислений, которые должны выполнить стороны *A* и *B* в ответ на сообщения M_A и M_B, соответственно.

Согласно (3), безопасность канала передачи данных между компонентами находятся в прямой пропорциональности от расхождения функций кодирования *E* и декодирования *D*. Анализ расхождений удобно проводить с помощью сравнения деревьев разбора на стороне передачи и стороне приема. Для одного протокола согласно (3) существует два направления атаки – функция кодирования *E* и функция декодирования *D*.

Заключение. Основной результат, полученный в данной статье, состоит в оценке уязвимости системы, использующей различные реализации протокола обмена данными.

Список литературы

1. *Geer D*. Vulnerable compliance login // The USENIX Mag. 2010. V.356, no.6. pp.26-30.
2. *Томашевская В.С., Яковлев Д.А.* Способы обработки неструктурированных данных // Russian Technological Journal. 2021. Т.9, №1. с.7-17.
3. *Chomsky N*. On certain formal properties of grammars // Information and Control. 1959. Vol.2, no.2. pp.137-67.
4. *Cook M*. Universality in elementary cellular automata // Complex Systems. 2004. Vol.15. pp.1-40.
5. *Korovina K.S., Rudova I.S.* The execution complexity of logical formulas with restricted quantifiers based on CF-grammars // Journal of Physics: Conference Series. 2021. Vol.2131. pp.022131.
6. *Unger A.Y.* A formal pattern of information system design // Journal of Physics: Conference Series. 2021. V.2094. pp.032045.