МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное автономное образовательное учреждение
высшего профессионального образования

**«МОСКОВСКИЙ ФИЗИКО - ТЕХНИЧЕСКИЙ ИНСТИТУТ
(ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ)»**

ФАКУЛЬТЕТ УПРАВЛЕНИЯ И ПРИКЛАДНОЙ МАТЕМАТИКИ

КАФЕДРА ТЕХНОЛОГИЧЕСКОГО ПРЕДПРИНИМАТЕЛЬСТВА

НА ПРАВАХ РУКОПИСИ
УДК _____

АКУЛЕНКО ВЕРА СЕРГЕЕВНА

РЕАЛИЗАЦИЯ АЛГОРИТМА КЛАСТЕРИЗАЦИИ HTTP-ОТВЕТОВ И ОБНАРУЖЕНИЕ АНОМАЛИЙ АТАК ПЕРЕБОРА

(МАГИСТЕРСКОЙ ДИССЕРТАЦИИ)

Магистерская диссертация

НАПРАВЛЕНИЕ ПОДГОТОВКИ: 010900 «ПРИКЛАДНЫЕ МАТЕМАТИКА И ФИЗИКА»

МАГИСТЕРСКАЯ ПРОГРАММА _____
(ШИФР, НАЗВАНИЕ)

ЗАВЕДУЮЩИЙ КАФЕДРОЙ _____/ _____/

НАУЧНЫЙ РУКОВОДИТЕЛЬ: _____/ _____/

Студент: _____/ _____/

Г. МОСКВА
2016

Содержание

Введение

Актуальность. Веб-приложения являются предметом атак со стороны злоумышленников, которые пытаются получить несанкционированный доступ к информации и ресурсам или внедрить вредоносный код.

Некоторые способы атак включают не однократные попытки получения доступа к одному и тому же ресурсу, что, как правило, приводит к большому числу неудачных попыток. Это широко известно как атака типа перебора или brute force атака.

Злоумышленники, используя brute force, могут нанести вред веб-прилежению:

- получить доступ к несанкционированной информации и ресурсам;
- внедрить вредоносный код;
- размещение несанкционированной рекламы;
- кража персональных данных;
- шантаж владельца сайта;
- воровство денег и тд

Цель данного исследования обнаружение атак типа brute force за счет анализа поведенческого взаимодействия пользователей с защищаемым веб-приложением. Данный анализ проводят с учётом параметров ответа сервера на получаемые запросы пользователей веб-приложения.

Объект исследования данной работы сетевые пары запрос-ответ от интернет-пользователей к защищаемому веб-приложению. Сетевые пары запрос-ответ составленные по протоколу HTTP.

Предмет данного исследования обнаружение сетевых атак типа Brute force (перебора) путем анализа динамики интернет – трафика от сетевых адресов, взаимодействующих с защищаемым веб-приложением.

Научной новизной является то, что пороговое значение количества запросов интернет – пользователей к части веб-приложения, зависит от запросов предыдущих интернет – пользователей к этой части и ответов интернет - сервиса на них. Где частям веб-

приложения соответствуют кластеры, которые формируются из сохраненных характеристик пар «запрос – ответ».

В данном исследовании предложен метод детектирования brute force атаки, состоящий в том, что

принимают поток запросов на интернет – сервис от интернет – пользователей,

сохраняют определенные характеристики каждой пары «запрос – ответ»,

формируют кластеры из сохраненных характеристик пар «запрос – ответ»,

генерируют набор правил попадания вновь принятой пары «запрос –ответ», в один из сформированных кластеров,

задают интервалы времени, за которые собираются данные о парах «запрос-ответ» при обнаружении аномалий,

для каждого интервала времени и каждого интернет – пользователя считаем количество его запросов за этот промежуток времени, попадающих в каждый из сформированных кластеров, сохраняем эти данные,

для каждого промежутка времени и кластера определяют пороговое количество запросов, попадающих в кластер,

запоминают сформированную статистику,

далее переводят систему в режим обнаружения атак,

после чего для каждого промежутка времени и для каждого интернет – пользователя считают количество запросов, относящихся к каждому кластеру,

сравнивают полученное значение с пороговым значением для каждого кластера,

при превышении значения порога принимают решение об аномальном поведении и обнаружении атаки.

Список обозначений

$\mathbb{R}$ — множество вещественных чисел.

$\mathbb{N}$ —множество натуральных чисел.

1 Описание сетевых запросов

1.1 Описание HTTP протокола

В данной работе предметом исследования выступают сетевые запрос и ответы сервера по протоколу HTTP.

HTTP (англ. HyperText Transfer Protocol — «протокол передачи гипертекста») — протокол прикладного уровня передачи данных (изначально — в виде гипертекстовых документов в формате HTML, в настоящий момент используется для передачи произвольных данных). Основой HTTP является технология «клиент-сервер», то есть предполагается существование потребителей (клиентов), которые инициируют соединение и посылают запрос, и поставщиков (серверов), которые ожидают соединения для получения запроса, производят необходимые действия и возвращают обратно сообщение с результатом.

1.2 Структура протокола HTTP

Каждое HTTP-сообщение состоит из трёх частей, которые передаются в указанном порядке:

Стартовая строка (англ. Starting line) — определяет тип сообщения;
Заголовки (англ. Headers) — характеризуют тело сообщения, параметры передачи и прочие сведения;
Тело сообщения (англ. Message Body) — непосредственно данные сообщения. Обязательно должно отделяться от заголовков пустой строкой.
Заголовки и тело сообщения могут отсутствовать, но стартовая строка является обязательным элементом, так как указывает на тип запроса/ответа. Исключением является версия 0.9 протокола, у которой сообщение запроса содержит только стартовую строку, а сообщения ответа только тело сообщения.

1.3 Заголовки HTTP

В данной работе будут использоваться значения определенных заголовков, каких именно будет описано ниже.

Заголовки HTTP (англ. HTTP Headers) — это строки в HTTP-сообщении, содержащие разделённую двоеточием пару параметр-значение. Формат заголовков соответствует общему формату заголовков текстовых сетевых сообщений ARPA (см. RFC 822). Заголовки должны отделяться от тела сообщения хотя бы одной пустой строкой.

Примеры заголовков:

Server: Apache/2.2.11 (Win32) PHP/5.3.0
Last-Modified: Sat, 16 Jan 2010 21:16:42 GMT
Content-Type: text/plain; charset=windows-1251
Content-Language: ru

В примере выше каждая строка представляет собой один заголовок. При этом то, что находится до первого двоеточия, называется именем (англ. name), а что после неё — значением (англ. value).

2 Описание Brute force

Полный перебор (или метод 'грубой силы' от англ. brute-force) - метод решения задачи путем перебора всех возможных вариантов.

Brute froce атака может проявляться по-разному, но в первую очередь состоит в том, что злоумышленник заранее определяет значения, используя которые отправляет запросы к серверу, а затем анализирует сетевые ответы. Для уменьшения количества значений перебираемого параметра, злоумышленник может использовать атаку по словарю или традиционную атаку brute force (с заданным классом символов). Учитывая способ атаки, количество попыток, эффективность атакующей системы, злоумышленник может вычислить время, которое потребуется, чтобы перебрать все заранее заданные значения.

Примеры параметров для перебора: логина, пароля, промо кода и т.д..

3 Обзор способов детектирования Brute force

3.1 Способ 1

Общие принципы и подходы к обнаружению сетевых атак приведены, например, в статье А.А. Корниенко, И.М. Слюсаренко «Системы и методы обнаружения вторжений: современное состояние и направления совершенствования»

http://citforum.ru/security/internet/ids_overview.

3.2 Способ 2

В патентной заявке US 20090031244 описан метод обнаружения сетевых атак путем сбора и анализа статистики при передаче данных пользователем. В заявке US 20080010247 описана система по сбору информации о действиях пользователя и составлению конечного профиля с помощью набора правил. В патенте EP 2109282 описывается возможность построения гистограммы запросов (как профиля) пользователя для дальнейших действий по отсечению трафика.

3.3 Способ 3

Известны система и способ обнаружения сетевых атак методом полного перебора US 2016/0004580.

Решение обеспечивает обнаружение потенциальных атак на домен в ответ на аномальное событие.
Выбирают значение параметра лямбда, соответствующее аномальному событию, из созданной модели для текущего интервала времени. Определяют вероятность того, является ли общее количество аномальных событий для текущего интервала времени легитимным, используют функцию распределения, зависящую от параметра лямбда, определяют факт умышленной атаки, если вероятность меньше или равна выбранного значения альфа.

Недостатком этого решения является отсутствие возможности автоматически идентифицировать события, как аномальные.

3.4 Способ 4

Известны система и способ уменьшения ложных срабатываний при определении сетевой атаки RU 2480937.

Технический результат при анализе поведенческого взаимодействия пользователей с защищаемым ресурсом состоит в уменьшении вероятности ложных срабатываний при обнаружении сетевой атаки. Он достигается тем, что система содержит следующие модули: управляющий модуль, предназначенный для хранения статистики предыдущих сетевых атак для корректировки правил фильтрации для центров очистки; коллекторы, предназначенные для составления правил фильтрации на основании информации о трафике от центров очистки и сенсоров; центры очистки, предназначенные для фильтрации трафика на основании правил фильтрации; сенсоры, предназначенные для агрегирования информации о трафике для дальнейшей передачи на коллекторы.
Способ фильтрации сетевого трафика для защиты сервиса от сетевых атак, содержащий этапы, на которых:
(i). перенаправляют трафик к сервису на сенсоры и центры очистки;
(ii). обрабатывают на сенсорах все запросы к сервису с дальнейшим агрегированием полученной информации;
(iii). обновляют правила фильтрации на коллекторах, используя полученную от сенсоров информацию;
(iv). корректируют обновленные правила фильтрации с помощью управляющего модуля на основании статистики предыдущих сетевых атак;
(v). фильтруют трафик на центрах очистки, используя заданные правила фильтрации, при этом центры очистки подключены к магистральным каналам связи по каналам с высокой пропускной способностью.

Недостатком данного решения является то, что при анализе запросов от некоторого IP адреса не учитывают параметры ответов сервера. Это приводит к возможности, когда система не может отличить N – кратное введение валидного значения промо кода добросовестным пользователем от N – кратного введения не валидного значения злоумышленником.

3.5 Способ 5. Способ защиты от brute-force login attack

В этом разделе описано,что такое взлом аккаунта с помощью brute-force attack (метод последовательного перебора) и как можно защитить сотрудников и приложения в своей компании от этих атак?

Brute-force login attack является наиболее распространённой (и наименее изощренными) атакой, используемой против веб-приложений. Цель данной атаки – получить доступ к аккаунтам пользователей путем многократных попыток угадать пароль пользователя или группы пользователей. Если веб-приложения не имеют никаких защитных мер против этого типа атак, то злоумышленнику довольно просто взломать систему, основанную на парольной аутентификации, осуществив сотню попыток ввода пароля с помощью автоматизированных программок, легкодоступных в Интернете.

Brute-force login attacks может быть использована в ряде случаев. Если известна длина пароля, то может быть испробована каждая комбинация цифр, букв и символов, пока не будут найдены совпадения. Однако процесс этот долгий, особенно по мере увеличения длинны пароля (вот почему длинные пароли более надежны). Альтернативный подход – использование списка общеупотребительных слов, так называемая словарная атака. Общий смысл этой атаки сводится к тому, что будут последовательно подставляться все слова из словаря, с возможностью добавлять цифры и удваивать слово как потенциальный пароль. В этом случае гораздо меньше комбинаций можно попробовать, но все же шанс подобрать пароль довольно высок.

Есть обратный метод, вместо попытки подобрать пароль к одному аккаунту, можно попробовать один пароль к множеству аккаунтов. Это известно как reverse brute-force attack. Данная техника, стоит заметить, не срабатывает там, где есть политика блокировки учетной записи. Reverse brute-force attacks менее распространенная атака еще и потому, что атакующему зачастую сложно составить достаточно большой объем имен для этой атаки.

Есть ряд методов для предотвращения brute-force attack. Первый заключается в использовании политик блокировки учетной записи. Например, после трех неудачных попыток входа в систему, учетная запись блокируется до тех пор, пока ее не

разблокирует администратор. Недостатком этого метода является блокировка сразу множества аккаунтов пользователей в результате атаки одного злоумышленника, на администратора падает сразу много работы, т.к. большое количество пользователей-жертв осталось без доступа к своим аккаунтам.

Лучший, хотя и более сложный метод – progressive delays (прогрессивные задержки). Суть его в том, что учетные записи блокируются на некоторое время после нескольких неудачных попыток входа. Время блокировки возрастает с каждой новой неудачной попыткой. Это защищает от автоматизированных средств, проводящих brute-force attack, и фактически делает нецелесообразным проведение данных атак.

Другой метод заключается в использовании теста запроса-ответа на странице входа в систему для предотвращения автоматизированных представлений. Такие бесплатные утилиты как reCAPTCHA могут быть использованы для того, чтобы попросить пользователя ввести слово или решить простую математическую задачу, тем самым доказав, что это не робот. Этот метод является эффективным, но создает некоторые неудобства при пользовании сайтом.

Любое веб-приложение должно обеспечивать использование надежных паролей. Так, например, требование от пользователя выбирать пароли длиной восемь и более символов с использованием букв и цифр или специальных символов отличная защита от brute-force attack, особенно в сочетании с одним из вышеописанных методов.

Так же может быть полезно использовать утилиты, которые автоматически считывают журналы интернет-событий и оповещает администратора о неоднократных попытках исходящих от одного IP адреса. Однако, злоумышленник также просто может использовать различные инструменты, чтобы регулярно автоматически сменять свой IP адрес.

Чтобы пользователи могли доверять вашей компании свои личные данные, очень важно убедиться в том, что в веб-приложении используется хотя бы один из методов защиты против brute-force attacks. Использование методов, описанных в этой статье, должно обеспечить надежную защиту от этих распространенных атак.[9]

4 Постановка задачи

4.1 Формулировка задачи кластеризации

Задача кластеризации (или обучения без учителя) заключается в следующем. Имеется обучающая выборка $X_l = \{x_1, \ \dots \ , x_l\} \subset X$ и функция расстояния между объектами $\rho: \ X \times X \to [0, +\infty)$, X – пространство объектов. Требуется разбить выборку на непересекающиеся подмножества, называемые кластерами, так, чтобы каждый кластер состоял из объектов, близких по метрике ρ, а объекты разных кластеров существенно отличались. При этом каждому объекту $x_i \in X_l$ приписывается метка (номер) кластера y_i.

Алгоритм кластеризации — это функция a: X → Y , которая любому объекту x ∈ X ставит в соответствие метку кластера y ∈ Y . Множество меток Y в некоторых случаях известно заранее, однако чаще ставится задача определить оптимальное число кластеров, с точки зрения того или иного критерия качества кластеризации. Решение задачи кластеризации принципиально неоднозначно, и тому есть несколько причин. Во-первых, не существует однозначно наилучшего критерия качества кластеризации. Известен целый ряд достаточно разумных критериев, а также ряд алгоритмов, не имеющих чётко выраженного критерия, но осуществляющих достаточно разумную кластеризацию «по построению». Все они могут давать разные результаты. Во-вторых, число кластеров, как правило, неизвестно заранее и устанавливается в соответствии с некоторым субъективным критерием. В-третьих, результат кластеризации существенно зависит от метрики ρ, выбор которой, как правило, также субъективен и определяется экспертом[5].

4.2 Постановка задачи кластеризации HTTP пар “запрос-ответ” и поиска аномалий типа brute force

Введем следующие обозначения:

Req — пространство сетевых запросов по протоколу HTTP.

X — пространство характеристик сетевых запросов.

$h: Req \to X$, h — функция, отображающая пространство запросов в пространство характеристик.

$\rho: X \times X \to [0, +\infty)$, ρ — функция расстояния, характеризует схожесть сетевых запросов.

$K = \{1, 2, \dots, \mathrm{k}\}$, K — множество кластеров.

R_{t_i} — подмножество запросов за i-ый интервал времени t_i , $R_{t_i} \subseteq Req$.

$X_{t_i} = \{x \in X \mid x = h(req), req \in R_{t_i}\}$, X_{t_i} — образ множества R_{t_i} в пространстве X .

C_{t_i} — разбиение множества X_{t_i} на кластеры $\left\{C_{t_i}^{\ j}\right\}_{j \in K}$, где множество $\left\{C_{t_i}^{\ j}\right\}_{j \in K}$ удовлетворяет следующим условиям:

$$X_{t_i} = \bigcup_{j \in K} C_{t_i}^{\ j},$$

$C_{t_i}^{j} \cap C_{t_i}^{l} = \varnothing$, где $j \in K, \ \ l \in K, \ \ j \neq l$.

S_{t_i} — состояние процесса обучения на начало интервала времени t_i . Подробнее S_{t_i} будет описано далее работы.

Каждый кластер $C_{t_i}^{\ j}$ описывается нормой.

$B: S_{t_i} \to C_{t_i}$, B — алгоритм построения C_{t_i}. На вход алгоритм принимает S_{t_i} и строит C_{t_i}.

$\mathrm{A}: S_{t_i} \times X_{t_i} \to S_{t_{i+1}}$, A — алгоритм обучения. На вход принимает S_{t_i} и X_{t_i}, на выходе алгоритма $S_{t_{i+1}}$.

Таким образом, в данной работе будут исследованы следующие задачи:

1. Введение функции $h: Req \to X$;
2. Введение функции расстояния $\rho: X \times X \to [0, +\infty)$;
3. Введение оптимальности разбиения множества X;
4. Описание S_{t_i}, $i \in \mathbb{N}$;
5. Создание алгоритма $\mathrm{A}: S_{t_i} \times X_{t_i} \to S_{t_{i+1}}$;
6. Создание алгоритма детектирования Brute force аномалий. На вход алгоритм принимает S_{t_i} и HTTP трафик.

5 Описание предлагаемого алгоритма

5.1 Кластеризация

5.1.1 Выделение характеристик для кластеризации

Каждый HTTP “запрос - ответ” содержит следующие заголовки, которые выделены для кластеризации и поиска аномалий в http - трафике:

1. **Длина ответа** (англ. Content-Lenght) — размер содержимого сущности (HTTP-ответа) в октеках (которые в русском языке обычно называют байтами). [1] Обозначим через len значение данного параметра.

2. **Время отклика** (англ. *response time*) — время, которое требуется системе на то, чтобы ответить на HTTP-запрос. Время отклика — *в технологии* время, которое требуется системе или функциональной единице на то, чтобы отреагировать на данный ввод. [3] Обозначим через time значение данного параметра.

3. **Код состояния HTTP** (англ. HTTP status code)— часть первой строки ответа сервера при запросах по протоколу HTTP. Он представляет собой целое число из трёх десятичных цифр. [2] Обозначим через stat_code значение данного параметра.

4. **IP-адрес** с которого получен HTTP – запрос. Обозначим через ip значение данного параметра. [7]

5. **Единый указатель ресурса** (англ. *Uniform Resource Locator*, URL) — единообразный локатор (определитель местонахождения) ресурса.[4] Обозначим через url значение данного параметра.

5.1.2 Сбор данных

Каждому http «запрос - ответу» ставим в соответствие уникальный идентификатор counterid. Для каждого http «запрос - ответа» сохраняем значения его параметров [counterid, len, time, stat_code, ip, url] в таблицу 1 Counters.

Факт: в таблицу постоянно добавляются новые значения характеристик HTTP “запрос - ответа”

counterid	len	time	stat_code	ip	url

Таблица 1. Counters

5.1.3 Создание разбиений

На первом этапе разбиваем на группы http-ответы, относящиеся к одному уникальному значению url'a.

Второй этап. Каждую группу, выделенную на первом этапе, http-ответов разделяем на группы алгоритмом (назовём его clustering_res) , описанным ниже.
Описание алгоритма clustering_res. Таким образом, алгоритм принимает на вход числовые значения параметров http-ответов, а именно [len, time, stat_code], относящие к одному значению url'a. Данный алгоритм применяется к точкам в трехмерном пространстве.

Вводим функцию расстояния между двумя точками, например i = [len_i, time_i, stat_code_i] и j = [len_j, time_j, stat_code_j]:

$$\rho(i, j) = 10^3 \bullet |len\_i - len\_j| + |time\_i - time\_j| + 10^{10} \bullet |stat\_code\_i - stat\_code\_j| \quad (1)$$

Фиксируем минимальное(k_min) и максимальное (k_max) количество кластеров(k_min и k_max натуральные числа, k_max > k_min).

Запускаем алгоритм K-Means [5] для k кластеров, где k пробегает значения от k_min до k_max с интервалом единица, m раз (m фиксируется). Таким образом, алгоритму k-means передаем следующие входные данные:

- k (количество кластеров);
- «точки данных» для алгоритма это числовые значения параметров [Content-Length, *response time*, status code], точки в трехмерном пространстве;
- введенную функцию расстояния (1).

В итоге k-means запустился (m*(k_max — k_min)) раз. На данном этапе имеется (m*(k_max — k_min)) разбиений. Далее среди этих разбиений выберем одно следующим способом, описанным ниже.

5.1.4 Выбор наилучшего разбиения

Вводим коэффициент, оценивающий качество каждого разбиения, обозначим его Validity_index. [6]

$\text{Validity\_index} = \text{silhouette\_index}$, где

Формула silhouette_index подсчета индекса и его значения описаны здесь: [6, стр. 20].

Выбираем одно разбиение, которое имеет минимальный Validity_index индекс.

На данном этапе имеем одно разбиение для одного url’a.

5.1.5 Описание кластеров

Для каждого кластера введем описание:

1. url — значение Url’a, который был кластерезован.
2. Центр кластера.
   - $\text{len\_center} = \frac{1}{\text{counters\_count}} \sum_{i=1}^{\text{counters\_count}} \text{len\_i}$
   - $\text{time\_center} = \frac{1}{\text{counters\_count}} \sum_{i=1}^{\text{counters\_count}} \text{time\_i}$
   - $\text{stat\_code\_center} = \mathrm{mode}\{\text{stat\_code\_i}\}_{i=1}^{counters\_count}$. Mode(мода) – значение во множестве наблюдений, которое встречается наиболее часто. [8]

Counters_count – количество кластеризованных точек.

3. Данные о кластере. Введем обозначения:
   - len_min – минимальное значение Content-Length в кластере;
   - len_max – максимальное значение Content-Length в кластере;
   - time_min – минимальное значение response time в кластере;
   - time_max – максимальное значение response time в кластере;
   - stat _code_min – минимальное значение status code в кластере;
   - stat _code_max – максимальное значение status code в кластере;

Каждому кластеру поставим в соответствие уникальный идентификатор, обозначим его clusterid. Сохраняем следующие данные о кластере [clusterid , url_value, len_center, time_center, stat_code_center, [len_min, time_min, stat_code_min, len_max, time_max, stat _code_max]] в таблицу 2 Clusters.

При выполнении следующих четырёх условий, принимается решение о принадлежности пары http «запрос-ответ» с характеристиками [counterid, len_i, time_i, stat_code_i, url_i] к кластеру с описанием [clusterid , url_value, len_center, time_center, stat_code_center, [len_min, time_min, stat_code_min, len_max, time_max, stat_code_max]]:

- url_value = url_i. Значение Url’a http-ответа совпадает со значением url’a, к которому относится кластер;
- len_min <= len_i <= len_max;
- time_min <= time_i <= time_max;

- status _code_min <= stat_code_i <= status _code_max.

Обозначим данное правило Cluster_rule.

clusterid	url_value	len_center	time_center	stat_code_center	len_min	time_min	stat_code_min	len_max	time_max	stat_code_max

Таблица 2. Clusters

5.2 Поиск аномалий. Детектирование Brute Force

В этом разделе будет описан алгоритм, назовём его Brute_detect, поиска аномалий в http – трафике.

5.2.1 Сбор данных о http-трафике по кластерам

Входными данными для данного алгоритма являются таблица 1 Counters и таблица 2 Clusters. Факт: в таблицу 1 Counters постоянно добавляются характеристики новых пар http – «запрос-ответов».

Фиксируется промежуток времени в секундах, обозначим его timeslot, например, 60 секунд.

Последовательно каждые timeslot секунд выбираем из таблицы 1 Counters строки, которые были записаны в эту таблицу за последние timeslot секунд. По этим строкам собираем следующие данные и сохраняем в таблицу 3 Stat_cluster. Напоминание: в таблице 1 Counters характеристики http – “запрос-ответа” [counterid, len, time, stat_code, ip, url].

- Ip – значение ip из таблицы 1 Counters;
- Clusterid.Используя характеристики http – “запрос-ответа” [counterid, len, time, stat_code, ip, url] и правило Cluster_rule, описанное выше в пункте “Описание кластеров”, определяем clusterid к которому относится данный counterid;
- Count – количество counterid, с данным ip, которые относятся к кластеру clustered;
- stat_cl_id – идентификатор данной характеристики;

stat_cl_id	timeslot	ip	clusterid	count

Таблица 3. Stat_cluster

Так как в таблицу 1 Counters постоянно добавляются строки, то и в таблицу 3 Stat_cluster, тоже буду постоянно добавляться строки.

5.2.2 Определение аномального поведения

Для каждой уникальной пары значений [timeslot, clusterid], Из таблицы 3 выбираем строки, которые им соответствуют, обозначим их количество count_stat. Из этих строк выбираем значения count, обозначим их count_i, формируем из этих значений упорядоченный по возрастанию массив, обозначим его

$$sort\_counts = \{count\_i\}_{i=1}^{count\_stat}.$$

Вычислим величину, обозначим её как q3, ниже которой лежит 75% значений $sort\_counts$.

$dq = q3 - count\_1$, count_1 – первый элемент в массиве sort_count.

Вычислим $count\_req = q3 + dq * 3$.

Сохраняем описанные данные в таблицу 4. Custer_frequency

Custer_frequency_id	timeslot	clusterid	count_req

Таблица 4. Custer_frequency

Напоминание: в таблицу 3 Stat_cluster постоянно добавляются значения [stat_cl_id, timeslot, ip, clusterid, count], обозначим [stat_cl_id_v, timeslot_v, ip_v, clusterid_v, count_v], эти значения, соответственно. Выбираем из таблицы 4 Custer_frequency строчку со значениями в столбцах timeslot = timeslot_v, clusterid clusterid_v, соответствующее значение в столбце count_req, обозначим count_req_v. Условие аномального ip: если count_v > count_req_v, то ip_v считаем аномальным.

6 Преимущества и недостатки предлагаемого решения

6.1 Преимущества

Данное решение не требует доработки веб-приложения для начала использования, не требует привлечения специалиста для перенастройки защиты при изменении веб-приложения.

Правила блокировки от brute force генерируются автоматически в зависимости от поведения пользователей на защищаемом веб-приложении.

6.2 Недостатки

Так как аномальное количество запросов к кластеру выставляется автоматически на основе собранной статистики предыдущих пользователей, то необходимо иметь трафик от самих пользователей к веб-приложению.

Система может быть последовательно обучена таким образом, что аномальное поведение будет считаться нормальным. Злоумышленники, которые знают, что за ними наблюдают при помощи таких систем, могут обучить их для использования в своих целях.

7 Заключение

В данном исследовании был предложен метод детектирования brute force атаки, состоящий в том, что

принимают поток запросов на интернет – сервис от интернет – пользователей,

сохраняют определенные характеристики каждой пары «запрос – ответ»,

формируют кластеры из сохраненных характеристик пар «запрос – ответ»,

генерируют набор правил попадания вновь принятой пары «запрос –ответ», в один из сформированных кластеров,

задают интервалы времени, за которые собираются данные о парах «запрос-ответ» при обнаружении аномалий,

для каждого интервала времени и каждого интернет – пользователя считаем количество его запросов за этот промежуток времени, попадающих в каждый из сформированных кластеров, сохраняем эти данные,

для каждого промежутка времени и кластера определяют пороговое количество запросов, попадающих в кластер,

запоминают сформированную статистику,

далее переводят систему в режим обнаружения атак,

после чего для каждого промежутка времени и для каждого интернет – пользователя считают количество запросов, относящихся к каждому кластеру,

сравнивают полученное значение с пороговым значением для каждого кластера,

при превышении значения порога принимают решение об аномальном поведении и обнаружении атаки.

Кроме того, возможно повышение адекватности модели путём добавления в неё параметров сетевых запросов пользователей и/или сетевых ответов сервера.

Практическим применением результата работы является использование данного алгоритма, в Web Application Firewall(сервис блокировки хакерских атак на веб-приложение)

Список использованных источников

1. https://ru.wikipedia.org/wiki/Список_заголовков_HTTP
2. https://ru.wikipedia.org/wiki/Список_кодов_состояния_HTTP
3. https://ru.wikipedia.org/wiki/Время_отклика
4. https://ru.wikipedia.org/wiki/URL
5. Воронцов К. В. Лекции по алгоритмам кластеризации и многомерного шкалирования [PDF] (http://www.ccas.ru/voron/download/Clustering.pdf), стр. 8-9
6. Сивоголовко Е. В. Методы оценки качества четкой кластеризации [PDF] (http://ipo.spb.ru/journal/content/1349/Методы%20оценки%20качества%20чёткой%20кластеризации.pdf)
7. https://ru.wikipedia.org/wiki/IP-адрес
8. https://ru.wikipedia.org/wiki/Мода_(статистика)
9. Rob Shapland. Techniques for preventing a brute force login attack. (http://www.computerweekly.com/answer/Techniques-for-preventing-a-brute-force-login-attack)