Analysis of Metasploit-based Exploit

Project 5 Group 13

Chejarla Rajesh Reddy
School of Information Technology
Indian Institute of Technology
Kharagpur, India
chraj.kool@gmail.com

Soumadip Biswas
School of Information Technology
Indian Institute of Technology
Kharagpur, India
soumadip.cse@gmail.com

Vaibhav Neharkar
School of Information Technology
Indian Institute of Technology
Kharagpur, India
nvaibhavv@yahoo.com

***Abstract*-** This paper is basically about gathering information about Metasploit, which is actually a collections of exploits and create a knowledge base and classify them into standard classifications as mentioned in Computer Attack system Classification [1].
Overall try is to create a parser which can automate the process of classification and which can be used in detection of notion of suspicious activities of attackers.

***Keywords*-** Metasploits, Payload, Mixin, msfconsole.

I. Introduction

An **exploit** is a piece of software, a chunk of data, or sequence of commands that takes advantage of a bug, glitch or vulnerability in order to cause unintended or unanticipated behaviour to occur on computer software, hardware, or something electronic (usually computerised). This frequently includes such things as gaining control of a computer system or allowing privilege escalation or a denial of service attack.

The **Metasploit Project** is an open-source computer security project which provides information about security vulnerabilities and aids in penetration testing and IDS signature development. Its most well-known sub-project is the **Metasploit Framework**, a tool for developing and executing exploit code against a remote target machine.

The Metasploit Framework is a penetration testing toolkit, exploit development platform, and research tool. The framework includes hundreds of working remote exploits for a variety of platforms. Payloads, encoders, and nop slide generators can be mixed and matched with exploit modules to solve almost any exploit-related task. The framework consists of tools, libraries, modules, and user interfaces. The basic function of the framework is a module launcher, allowing the user to configure an exploit module and launch it at a target system. If the exploit succeeds, the payload is executed on the target and the user is provided with a shell to interact with the payload. Hundreds of exploits and dozens of payload options are available.

II. Motivation & Objectives

a) Motivation

Though Metasploits is providing a lot of information such as different exploit codes, payload etc. But there is always a need of formatted information so that that information can be used for mining knowledge, thus this idea of making a knowledgebase out of the resources provided at the Metasploit framework comes into the picture.

b) Objectives

Our main objective is to analyze all those codes given in the framework, and then classify them according to their affiliation of attack.

The objective is to parse the following information

1. Name
2. References
3. Privileged
4. Platform
5. Architecture
6. Targets

Then another thing we are to find out is the required options for each exploit separately.
Our objective is to make this to be done automatically by writing some parser.

Note:

This project is being assigned to 4 groups, all are working on the same framework their domain is divided and they are disjoint. We have assigned the following area.

- msf3\modules\exploits\windows\brightstor\
- msf3\modules\exploits\windows\browser\

III. Proposed Methodology

In this project have created a parser which can parse our requirements, as per the need of our objectives. For this we used some methodology which can help us parsing strongly, as we have a strong requirement to handle regular expressions as different sizes of files are need to be parsed, we propose to use the language 'Perl' for this purpose. In the back-end we have used MY-SQL as database.

We have implemented 3 codes

- First one will extract the fields Name, Category, Mixing, Description, References, Platform, Privileged and store it in DB.
- Second one extracts the fields Options, Supported payloads, targets from command line and stored in files. For this we have used shell script.
- Then third one uses above files extract the appropriate fields and store it in DB.

The algorithms for these codes are as below:

Algorithm 1:
For parsing Name, Category, Mixin, Description, References, Platform, Privileged from exploits code.

create Database connection;
prepare SQL queries for truncating and inserting the parsed values

Create one Dictionary in perl

Open the directory which contains all the files to be parsed
opening files one after another an calls PARSE function for each file

PARSE FUNCTION

split the file into words
for each word MATCH(dictionary words)

Remove some unwanted strings and tabs from file using perl Regular Expressions
E.g. s/Metasploit3 < Msf::Exploit::/;

If word= class extract class into variable class

If word= include extract Mixin into variable Mixin

If word= Name extract name into variable Name

If word= Platform extract platform into variable plat

If word= Privileged extract Privileged into variable Privileged

Close the file handler

Open same file with another handler for handling multiple lines

Parse each file according to brackets using "extract_bracketed" function

if (brackets= '{ }') then

copy all lines between the bracket into curlybrack array

copy curlybrack[1] into Description variable

if (brackets= '[ ]') then

copy all lines between the bracket into sqaurebrack array

copy squarebrack[1] into Description variable

Close file handler

Then insert all parsed variables into the database

Algorithm 2:

for Extracting Options, Supported payloads, targets from MSFCONSOLE

//make sure: that the metasploit framework is properly installed.

First this code generates all the exploits name with the path as in the modules/exploit directory

Then next few symmetric lines actually finds the required exploits from which following data need to be extracted

1. payloads ==>stores in a file in same directory named "_pay"
2. options ==>stores in a file in same directory named "_opt"
3. targets ==>stores in a file in same directory named "_targ"

Then for each exploits specified in the list it will find the options, targets and payloads in the following format with the help of Unix commands and "msfcli" command:

E.g.
options=`msfcli $mod O 2>/dev/null | tail -n +4 | tr -s ' ' | cut -d ' ' -f 1 --complement | tr '\n' ':' | tr -s ':' | sed s/.$//`

options:
    <name of the exploits>
     <option_name yes/no>:<option_name yes/no>: ... :<option_name yes/no>
targets:
    <name of the exploits>
     <target_id target_name>:<target_id target_name>: ... :<target_id target_name>
payloads:
    <name of the exploits>
     <payload#1>:<payload#2>:<payload#1>: ...

// then the main "perl" script actually opens these files and appends the data to the database

Algorithm 3:

For parsing Options, Supported payloads,targets from each file created in the previous section

create Database connection;

prepare SQL queries for truncating and insering the parsed values

Open the directory containing that files
for each file
        split the file according to "\n" character and insert it into array

        if(filename == Targets)

                insert first index of array into Name field of target Database

                insert second index of array into targets field of target Database

        if(filename == options)

                insert first index of array into Name field of target Database

                insert second index of array into options field of options target Database

        if(filename == payload)

                insert first index of array into Name field of target Database

                insert second index of array into payload field of options payload Database

close the file

IV. Results & Discussion

a) Results

We have already analysed exploits those are given to us manually, and have extracted some typical information like name of the file, classification type[1], platform they works on and mixin. After that we build the parser which will extract all those field automatically and store it in the DB.

b) Discussion

This task of creating a parser which will find required information automatically is somewhat troublesome when the options and the classification is concerned, this needs to be handled some other way, as we have extracted the most of the classifications manually, so it can be extracted from there and that data can be used for further classifications and predictions using some available algorithms for further detection of new attacks, those notions can be pre considered previously.

c) Figures & Tables

There are no figures and tables for this project up to now.

V. Conclusion & Future Work

As we have created the knowledge base , this knowledge base will help in finding out new threat and new types of attacks, and also it will help in future, when developing a system one can use this knowledge base to patch all known loopholes previously so that the newly made system can become more promising.

Future work on this can be done in an unlimited fashion as the more complex system we will be developing, the more new attacks will come into the picture, thus there are opportunity of adding more and more data to the base and get more stronger knowledge for preventing attacks.

References

[1] N. Paulauskas, E. Garsva, ” Computer System Attack Classification”. ELECTRONICS AND ELECTRICAL ENGINEERING,Vol. 2, No. 66. (2006), pp. 84-87.

[2] www.metasploit.com/framework

[3] en.wikipedia.org/wiki/Metasploit_Project

[4] blog.metaslpoit.com