SNARE/TANNER

Google Summer of Code 2017 Proposal

Basic Information

Name : Ravinder Nehra

Slack username : rnehra01

Email : hnehra1@gmail.com

Nationality : Indian

Language : English

Top Project Choice

My top priority project is **Snare/Tanner**.

Are you willing and able to work on other projects instead?

I can work on **mitmproxy core, mitmproxy UI** and **Port Independent Protocol Identification Library**.

Coding skills

1. **Fluent in Python. Sound knowledge of Object Oriented Programming**

   I've been using python since last year. Some of my python scripts can be found here.

2. **HTML, Javascript, CSS - moderate knowledge**

   I've developed a chrome extension LPreview and a web app NotifyMe.

3. **Good knowledge of C/C++ and basic knowledge of Android App Development in Java**

4. **Good experience with git**

Development Environment

1. Ubuntu 16.04, python 2.7+ installed
2. Sublime text / nano
3. Git as version control system

Describe any previous usage with Honeynet Project tools.

I have used **mitmproxy** for solving ctf challenges. Though I haven't used honeypots in real life previously. But after my introduction to **Honeynet**, now I've become familiar.

Describe any previous usage Honeynet Project development experience.

Here is a list of PRs, I've made so far:

SNARE

1. #54(**merged**) : Custom header input functionality
2. #56(**merged**) : Functionality of session tracking using cookies on SNARE side.

TANNER

1. #112(**merged**) : Functionality of session tracking using cookies on tanner side.
2. #114(**merged**) : Add missing tests for config.py file.
3. #115(**merged**) : Fix #107.
4. #116(**merged**) : Add a functionality of checking attack types for all GET parameter separately.
5. #118(**merged**) : Auto updation of virtual docs.
6. #120(**merged**) : Update docs.
7. #122(**merged**) : Fix LFI false positive(#119) and LFI attribute errors(#121).
8. #128(**merged**) : Fix SQLI bug
9. #132(**merged**) : Improve dummy data using elizabeth
10. #133 : Implement MySQLI Emulator
11. #134(**merged**) : Implement string based SQLI attacks
12. #135(**merged**) : Add tests for base.py

Describe other open source development experience.

I started contributing to open source a few months back. Till now I have made small contributions wherever I found something of my interest which include fixing small bugs, improving documentation etc.

OWASP ZAP

This is an automated tool to find vulnerabilities in a web application. I've implemented a Ms-SQLI scanner in ZAP and improved the released version SQLI scanner. My work can be found at zap-extensions.

LPreview

This is a chrome extension that shows contents of the link when hovered upon. It has been developed using Javascript.

NotifyMe

It is a simple web app that shows weather, to-dos and emails from different accounts.It was developed using CSS, HTML and Javascript.

SoundProfile

This is an android app that manages sound-profile according to schedule. I'm developing this app by following all the phases in the software development cycle.

Apart from the above, I've made some contribution to OWTF-http-request-translator too.

More information about my projects can be found at my **GitHub profile**.

What school do you attend and what is your specialty/major at the school?

I'm a second year undergraduate currently enrolled in a B.Tech Computer Science & Engineering program(IV year) at IIT Roorkee.

How many years have you attended there?

I'm currently in my second year and will complete two years in April, 2017.

What city/country will you be spending this summer in?

I'll be spending my vacations(6 May - 15 July) in my hometown, Jhajjar in India. After that I'll be back to my college and spend rest time there till final evaluation.

How much time do you expect to have for this project?

I can easily devote 40-45 hours weekly during vacations. When college reopens, I'll be spending my time at my college and can devote 30-35 hours weekly.

Please list all jobs, summer classes, vacations and/or other commitments that you'll need to work around.

I've no commitments this summer.

Have you participated in any previous Summer of Code projects?

No, this is my first time.

Have you applied for (or intend to apply for) any other Google Summer of Code 2017 projects? If so, which ones?

No, I'm applying for this project only.

If you have a URL for your resume/CV, please list it here.

My **CV**.

If you wish to list any personal/blog URLs, do so here.

None yet.

Project Details

My entire project is a collection of small tasks that aims on increasing the functionalities of Snare/Tanner Honeypot. Here is a list of all the tasks in detail.

1. Implement Blind SQLI Emulator

1.1. Implement Boolean-based SQLI emulator

In Boolean-based attacks, attacker will try to ask TRUE/FALSE questions. This emulator will check against boolean-based syntax and returns the same page with NO payload if query results out to TRUE else return the index page.

1.2. Implement Time-based SQLI emulator

This can be done simply by adding a custom *sleep* function in SQLITE3 and inject user payload in query, calculate the time of query execution if it comes out to be more than 0.00, then user has TRUE(+VALID) Time-based attack payload, so return the same page with NO payload with some delay else return index page.

2. Implement SQLI emulator based on MySQL

Current SQLI emulator is based on sqlite3. But MySQL is used widely. So it is important to have a MySQL based SQLI emulator too. Initial idea is to implement this emulator with same basic structure as previous SQLI emulator. I've started initial work at **#133.**

3. Implement Command Execution Emulator

This emulator can be developed using the concept of ***virtual-docs*** used in LFI emulator. The basic plan is

- ***Virtual-docs*** will be improved by adding new directories and files for this emulator.
- Attack will be detected using a regex pattern of some mostly used commands such as **cat, echo** and **ls**.
- Proper response will be served to attacker based on input.
- Command set can be extended from suggestions from mentors.

4. Improve existing emulators and attack checking

4.1. Check attack against each GET and POST parameter

The existing emulators don't check ALL the GET and POST params against each attack. This would make something where each POST and GET parameter will be checked against each attack type. Also this would eliminate use of two methods handle_get and handle_post and make a unified method which will be called for each POST and GET parameter.

4.2. Implement cookie support for attacks

With the addition of functionality of cookies(#112), we can now check attacks in cookies too. This will be a good feature to implement.

5. Implement Padding Oracle Emulator

Padding oracle attacks are famous in cookies-based attacks. The idea is

- Send a default encrypted cookie (encrypted using a simple padding based oracle)
- Analyze padding of the cookie sent back by the attacker
- Return payload like **Invalid-padding** or **Response status-500** if invalid padding is found

6. Improve the tanner api

Currently Tanner api supports only two basic commands, this would add the following new commands to Tanner api.

- Get a session info from its sess-uuid
- Get all the sessions from a given IP of a particular snare-uuid
- Get all the sessions with a given owner-type of a particular snare-uuid
- Get all the sessions with a given user-agent of a particular snare-uuid
- Get all the sessions with a given attack-type of a particular snare-uuid
- Get all the session with a timestamp in given time-range of a particular snare-uuid

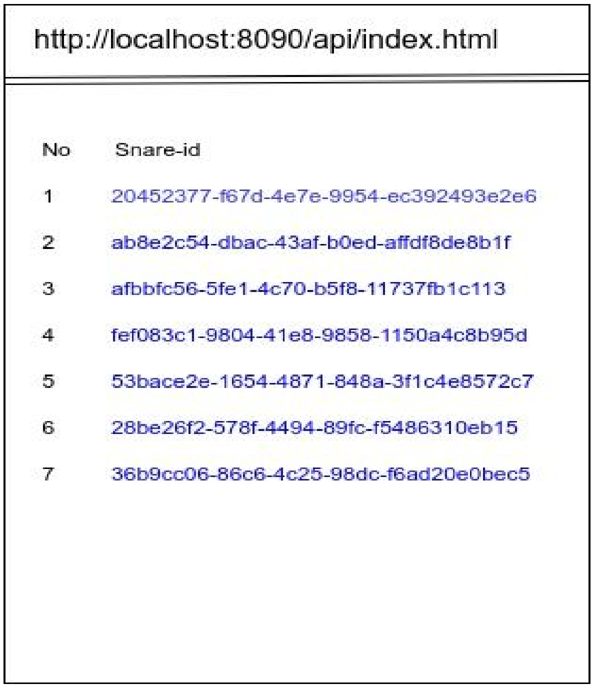
Some other commands can be implemented with suggestions from mentors.

7. Implement a WEB UI for tanner

Currently Tanner don't have a Web-UI, this would implement a basic UI for tanner to get session info. Web-UI is a much wanted feature in every project, so it would be nice to give it a good start.

For the start, three pages will be implemented which shows info about different objects. The initial idea is to make a HTML template file for each of the pages and get the relevant information from the new tanner api(task #6) and display the results by putting the information into the corresponding template file.

7.1. Home page

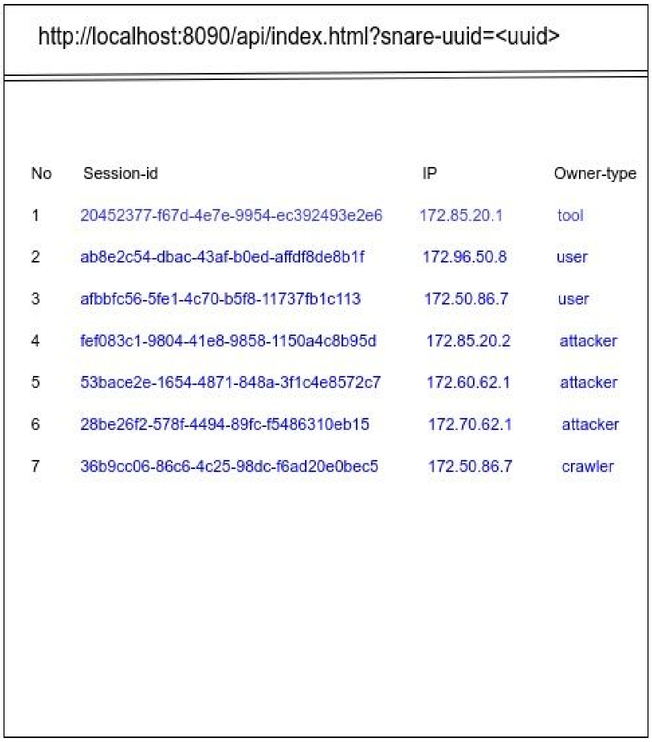


http://localhost:8090/api/index.html

No	Snare-id
1	20452377-f67d-4e7e-9954-ec392493e2e6
2	ab8e2c54-dbac-43af-b0ed-affdf8de8b1f
3	afbbfc56-5fe1-4c70-b5f8-11737fb1c113
4	fef083c1-9804-41e8-9858-1150a4c8b95d
5	53bace2e-1654-4871-848a-3f1c4e8572c7
6	28be26f2-578f-4494-89fc-f5486310eb15
7	36b9cc06-86c6-4c25-98dc-f6ad20e0bec5

This page shows all possible snares present with their respective uuid. Each of the snare object is clickable. On clicking, it will move to the second page.

7.2. Page showing info about a particular snare(uuid)

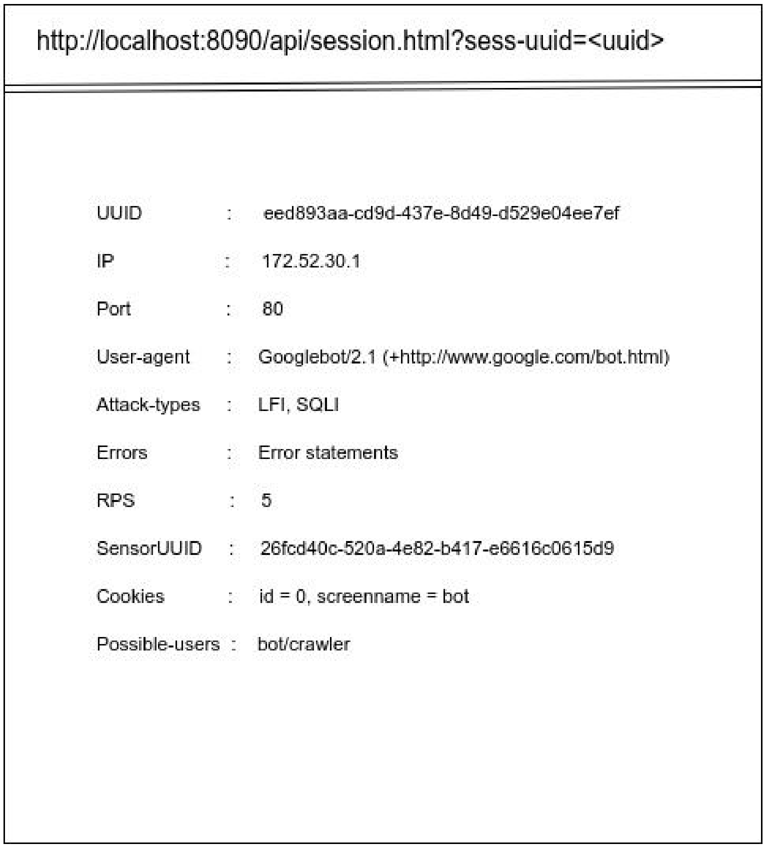


http://localhost:8090/api/index.html?snare-uuid=<uuid>

No	Session-id	IP	Owner-type
1	20452377-f67d-4e7e-9954-ec392493e2e6	172.85.20.1	tool
2	ab8e2c54-dbac-43af-b0ed-affdf8de8b1f	172.96.50.8	user
3	afbbfc56-5fe1-4c70-b5f8-11737fb1c113	172.50.86.7	user
4	fef083c1-9804-41e8-9858-1150a4c8b95d	172.85.20.2	attacker
5	53bace2e-1654-4871-848a-3f1c4e8572c7	172.60.62.1	attacker
6	28be26f2-578f-4494-89fc-f5486310eb15	172.70.62.1	attacker
7	36b9cc06-86c6-4c25-98dc-f6ad20e0bec5	172.50.86.7	crawler

This page shows all sessions and their brief info of a particular snare (based on the GET SNARE-UUID parameter). Usually this page gets loaded after a snare object gets clicked on the homepage. Each session object is clickable. On-clicking, third page will be loaded.

7.3. Page showing info about a particular session (based on uuid)



http://localhost:8090/api/session.html?sess-uuid=<uuid>

UUID : eed893aa-cd9d-437e-8d49-d529e04ee7ef

IP : 172.52.30.1

Port : 80

User-agent : Googlebot/2.1 (+http://www.google.com/bot.html)

Attack-types : LFI, SQLI

Errors : Error statements

RPS : 5

SensorUUID : 26fcd40c-520a-4e82-b417-e6616c0615d9

Cookies : id = 0, screenname = bot

Possible-users : bot/crawler

This page shows complete info about a session (based on the GET SESSION-UUID parameter). Usually this page gets loaded after a session object is clicked on the second page.

8. Make a unified system for Tanner/Snare Communication

This is a tanner **issue** which I think need good attention. The data that TANNER returns should be in a well defined format to make the code more organised and manageable. I would work on any of the approaches as discussed in the issue section of GitHub and finalize the approach after discussing it with the mentor.

9. Implement CRLF Emulator

This will emulate Carriage Return & Line Feed vulnerability.

- Attack will be detected using a regex(detecting **\r** and **\n** pattern)
- Payload will be injected in server-header, something like **this.**

10. Implement Code Execution Emulator

This emulator will implement code execution vulnerability. The basic idea is

- Attack detection using regex (matching common PHP functions like **print, echo, system etc**)
- Execute the code using **PHPox** and return the result to the attacker as payload

11. Test Emulators

Write python scripts which could test the emulators automatically using OWASP-ZAP python api.

This could be extended by using other web-vulnerability scanners which could be done on Mentor's suggestions.

Project Timeline

This is the timeline that I think is suitable. If, for any reason, a task is taking too long or cannot be completed, I will contact the mentor and after discussing the reason and explanation, suitable action will be taken (i.e. it will be decided whether it is fruitful to continue to work on that subproject or it would be more beneficial to SNARE/TANNER for me to take on next task and complete the former later). Hopefully, there will be no need to abandon any task. Also if any task needs to be rescheduled(due to its importance), that can done after proper discussion from the mentor.

I. **Community Bonding Period [5 May to 21 May]**

I'll familiarize myself with the mentors, understand the codebase more deeply and discuss about my project with mentors. Since I'm a past contributor so this won't take much time.

II. **Week 1 [22 May to 28 May]**
   - A. **Implement boolean-based and Time-based SQLI emulator (Task #1)**
   - B. **Write unit tests (if possible)**

III. **Week 2 [29 May to 4 June]**
   - A. **Implement MySQL based emulator(Task #2)**
   - B. **Write unit tests(if possible)**

IV. **Week 3 [5 June to 11 June]**
   - A. **Implement Command execution emulator (Task #3)**
   - B. **Write unit tests**

V. **Week 4 [12 June to 18 June]**
   - A. **Improve the emulator architecture (Task #4.1)**
   - B. **Add support of cookies for attacks (Task #4.2)**
   - C. **Add other possible improvements with suggestions from mentors**
   - D. **Write new unit tests**

VI. **Week 5 [19 June to 25 June]**
   - A. **Tidy the code and write unit test wherever necessary**
   - B. **Complete the documentation**

At this stage, three new emulators with unit tests and documentation should be available and base emulator code should have been improved. If not anything will be done further until Mid-Term evaluation, this is what to be submitted.

VII. **Week 6 [26 June to 2 July]**
   - A. **Implement padding-oracle emulator (Task #5)**
   - B. **Write unit tests**
   - C. **Take feedbacks from Mentors about first evaluation**

VIII. **Week 7 [3 July to 9 July]**
   - A. **Improve tanner api (Task #6)**
   - B. **Write unit tests**

IX. **Week 8 [10 July to 16 July]**
   - A. **Implement UI and backend code for HOME page (Task #7.1)**
   - B. **Implement UI and backend code for second page (Task #7.2)**

X. **Week 9 [17 July to 23 July]**
   - A. **Implement UI and backend code for third page (Task #7.3)**
   - B. **Write unit tests and documentation**

C. Submit code for second evaluation

At this stage, a working UI for TANNER, an improved TANNER api and a new emulator should be available with unit test and documentation.

XI. Week 10 [24 July to 30 July]

A. Finalize the technique to adopt for implementing this feature
B. Implement unified system for SNARE/TANNER communication (Task #8)
C. Write unit tests
D. Take feedbacks from Mentors about second evaluation

XII. Week 11 [31 July to 6 Aug]

A. Implement CRLF emulator (Task #9)
B. Write unit tests

XIII. Week 12 [7 Aug to 13 Aug]

A. Implement Code Injection Emulator (Task #10)
B. Write unit tests

XIV. Week 13 [14 Aug to 20 Aug]

A. Implement Emulator testing feature
B. Test the emulators using above Tester (Task #11)
C. Improve the code based on results of the Tester and suggestions from the mentors

XV. Week 14 [21 Aug to 27 Aug]

A. Write unit tests wherever necessary
B. Complete the documentation
C. Submit the code for Final evaluation

At this stage, two new emulators, emulator tester and an improved communication system should be available.

Apart from my scheduled tasks, my job will be to keep an eye on ISSUES that need good attention and fix them quickly. Also I'll help Evgeniya, my mentor in improving and testing the new cloner during the GSoC period regularly.

More about ME

I contribute to open source regularly, trying to play my role in thanking the open source softwares that I use and improving my skills as well. I'm an information security enthusiastic and love web security in particular. I participate in ctfs regularly as InfoSecIITR and HackInfinity. I write writeups of the challenges which I solve during ctfs. Apart from solving challenges I've also made some challenges for ctfs hosted on backdoor.

Why ME ?

I'm a consistent developer and quick learner which can be seen from my recent contributions to this project. I've a good knowledge of web security which can seen from my hacker profile I talked about. Since this is a web based honeypot, so previous web knowledge gives me an advantage in completing my objective of developing new emulators and improving the existing ones quickly. Also I've developed projects(though small) in all the languages that are required to complete my tasks. Apart from developing I pay equal attention to testing and documentation which are truly the most important parts of a successful project. Also I'm developing an **app** following proper software development guidelines, so I can apply my these skills for successful completion of this project.

Apart from that, I've been contributing to this project for last month, so I've a good knowledge of the source code, so I can start early to achieve my goals. Also I've almost done implementing a new emulator MySQLI which can be seen **here**. This adds another feather to my ability to complete this project.
This project will be a very good learning opportunity for me. It will be my first decent contribution to such a big organization. I would communicate with my mentor every week and will discuss with her my progress, objectives or any issue that I am facing.

What makes ME interested ?

I've been searching through security based GSoC organisations and came across HoneyNet.

The ideas page for SNARE/TANNER is easily understandable and fits to my knowledge. Also the mentor of this project supported me throughout. My motivation towards this project is mainly due to my interest in web security.

I believe Honeypots are of great importance in today's world where Internet access is growing and so as cyber attacks. Being a part of an open source honeypot project will be like contributing my share to world's cyber security.

Working with Honeynet will be best use of my time. More than all that, however, I feel SNARE/TANNER will be benefited with my skillset and I'll be proud to say that I've worked for Honeynet.

Have any of our members met you face to face, such as at one of our recent public events?

No, this is my first interaction with the HoneyNet members.