Creating Playbooks for IntelOwl

Google Summer Of Code 2022 Proposal

Personal Information

- **Name:** Aditya Narayan Sinha
- **Slack Username:** @0x0elliot
- **GitHub:** @0x0elliot
- **LinkedIn:** adityanarayansinha
- **Email:** adityanrsinha@gmail.com
- **Nationality:** Indian
- **Primary Spoken Language:** English and Hindi.
- **Location:** India, New Delhi.

Top Project Of Choice:

My top priority would be **IntelOwl**. (GitHub), Focusing on #14, Particularly playbooks (Issue 680) and the Option to run only free analyzers (Issue 733).

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

I am willing to work on other projects from HoneyNet as well, Particularly working on the frontend for GreedyBear.

Please describe your preferred coding languages and experience.

I have explored and worked thoroughly in a couple of domains in the past. Most prominent of which is CyberSecurity and FullStack Development.

- **Python**

I have been programming in python for the last 3 years. My Internship at SupplyNote which I have been pursuing since October'21 that requires me to continuously automate things using it and add features to our Flask backend. Before this, I also briefly did some freelance work as a FullStack dev where I used Flask mainly for the backend.

Other than that, I have worked using Python:

- to automate CTFs
- Push my coding skills
- Write CTFs in for CyberSecurity communities like AppSec Village DEFCON (I wrote a handful CTFs for them for DEFCON '29 and RSA Conference 2021. They even gave me a shirt!). If it helps, The leader I worked under for AppSec Village even wrote me a **letter of recommendation from AppSec Village** you can find here.
- CTFs for DigitalOverdose Autumn CTFs in 2021. (Digital Overdose has an annual Security Convention in which I am a scheduled speaker this year as well!)

- I have also helped write and host CTFs for other smaller communities that I have managed like BUHC (A small hacking community at my university led by me where I conducted a month-long CTF event) and Gateway (A discord cybersecurity community I led back in high school for which I wrote and hosted CTFs).

- **TypeScript/JavaScript (Angular, Next)**

My internship has enabled me to pick up lots of TypeScript/JavaScript. I work on Internal tools for our Point Of Sales software Posify (Many of which are hosted on Firebase with cloud functions written in JavaScript by me) and the frontend of our products which are all written in Angular as well.

Other than that, I have experience with JavaScript while coming up with payloads for XSS while doing CTFs and bug bounties on HackerOne and OpenBugBounty.

I have also contributed to https://csi-bu.in/ which is the official website for the computer society of India's chapter at my university, which I happen to be a part of as well.

- **Celery, Docker, Postgresql, Bash, Git, Redis, AWS Services (API Gateway, EC2, Lambda functions, Dynamodb etc) and Firebase etc**

I also felt like mentioning these technologies because I have to work with them on a day-to-day basis professionally.

Please describe any Windows, Unix or Mac OS X development experience relevant to your chosen project. If your project does not require OS-specific expertise, feel free to leave it empty.

I use MacOS and Linux (Manjaro) for development mostly.

Please describe any previous Honeynet Project or honeypot related development experience, including details of any patches, code or ideas you may have previously submitted.

PR	Status
Fixes #56: WHOIS RIPE analyzer	Merged

Fixes #634: BitcoinAbuse.com analyzer	Merged
Solves #722. Added CAPE Sandbox analyzer	Merged
Fixes #428: Add analyzer	Merged

Issues where I was helpful:

Issue
Add Issue Template
Playbooks (automated scripts)

The last issue: **Playbooks (automated scripts)** is where I want to focus on during this summer!

Please describe any previous open-source development experience, including projects you have worked on.

I am still fairly new to Open Source but I have somehow navigated my way across the at first, the overwhelming realm of Open Source and made a small number of contributions:

- **NullCTF Bot**

> https://github.com/NullPxl/NullCTF/issues/15

I fixed a bug that could have caused potential annoyance to large communities. (Fun fact, the bot calls you cool if you have contributed to it! :D)

- **Ask**

I took some liking to a small project back in High School called Ask which was a simple language written in Python to write REST APIs more easily. Although I was still a rookie, It sure did help me step into OSS better!

I helped fix an installation bug in Ask:

https://github.com/Buscedv/Ask/issues/86

And I helped add a small feature to it as well.

https://github.com/Buscedv/Ask/pull/58

Other than this,

I have been working on one of my projects a lot written in Python (web framework is written in **Django** and the utility server is written in **flask** and **click**) and Angular Typescript:

https://github.com/UturnOSS/Uturn-web

https://github.com/UturnOSS/Uturn-CLI

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

I am currently attending Bennett University (Greater Noida, UP, India), pursuing my B.Tech in CSE from there. My specialization is supposed to be in AI.

How many years have you attended there?

Almost 6 months by now. My course is for 4 years.

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

Greater Noida, UP mostly.

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

I would be able to devote 35-40 hours per week during my summer vacations and 20-30 hours per week when my colleges open up. I am hoping to contribute roughly 175 hours towards my addition.

Here is my detailed semester schedule. Due to the pandemic, the length/timing of my vacations would be slightly shortened for this semester. They would be starting from 30-06-2022 to an unspecified period (which I am assuming is at least till the 16th of September.)

However, My university happens to prioritize programs like GSoC and is open to supporting me accordingly if my proposal is selected.

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

I would make sure to have no commitments other than GSoC during my summer vacations.

Have you participated in any previous Summer of Code projects? If so please describe your projects and experience, including what you liked or didn't like about the experience

I haven't. This is my first time.

Have you applied for (or intend to apply for) any other Google Summer of Code 2022 projects? If so, which ones?

I haven't and I do not intend to do the same. I am, however, As mentioned earlier, open to working at other HoneyNet Projects.

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

Here is my resume.

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

My personal blog: https://medium.com/@0x0elliot

My YouTube Channel: https://www.youtube.com/0x0elliot

Project details

I have spoken about the technicalities of this issue thoroughly with the mentors here.

I plan on accomplishing the following 10 goals:

Tasks

1. Implement serializers and dataclasses
2. Implement the Plugin class (classes file) and controller file.
3. Implement the default `python_module`
4. Implement urls and view files.
5. Adding the frontend support for playbooks.
6. Adding playbook cache feature
7. Adding support for Playbooks in PyIntelOwl.
8. Creating a free-only analyzer playbook.
9. Documentation!

Before I expand on the tasks, I would like to explain what Playbooks themselves are supposed to be:

You would be able to find yourself a less technical definition here.

Playbooks are supposed to be pre-written flows that execute particular analyzers and connectors on the given IoC. The flow for this is supposed to be defined in a particular JSON format in `playbook_configuration.json` which would look somewhat like the following:

```
{
    "IP_BASIC_INFO": {
        "supports": ["ip"],
        "description": "Fetch basic info for an IP",
        "python_module": "ip_basic_info.IPBasicInfo",
        "analyzers": {
            "AbuseIPDB": null,
            "Shodan": {
                "include_honeyscore": true
            },
            "FireHol_IPList": null
        },
        "connectors": {
            "MISP": {
                "ssl_check": true
            },
            "OpenCTI": {
                "ssl_verify": true
            }
        }
     }
}
# Flow helpfully laid out by Eshaan
```

I would try to make sure that all the backend components follow the same structure as any other Plugin class (Analyzers and Connectors).

1. Implement serializers and dataclasses

Expected look for `playbook_config.json`:

```
{
    "IP_BASIC_INFO": {
        "supports": ["ip"],
        "description": "Fetch basic info for an IP",
        "analyzers": {
            "AbuseIPDB": null,
```

```
            "Shodan": {
                "include_honeyscore": true
            },
            "FireHol_IPList": null
        },
        "connectors": {
            "MISP": {
                "ssl_check": true
            },
            "OpenCTI": {
                "ssl_verify": true
            }
        }
    }
}
```

I would be making the serializer and data classes to read our `playbook_config.json` file in the format specified above. Since it would be inheriting from `AbstractConfigSerializer`, I would like to add a default `python_module` file as suggested by Eshaan here.

The `python_module` would contain the concrete class for Playbooks and its abstract plugin class would be separately written in classes.py.

2. Implement the Plugin class (classes file) and controller file.

I would like to implement a `Playbook` abstract `Plugin` class somewhat like we have for analyzers and connectors which takes care of the basic logging and health check functionality. Alongside, there would be a need for a controller file that would contain functions to trigger the celery functions with better control.

3. Implement the default `python_module` file

This default file would contain the logic for triggering the apt analyzers and connectors requested. Because of the way we are expecting and dealing with `python_module` someone can just extend our Playbook Abstract plugin class and add their own logic. When they create a new playbook, They can just pass in the path to their new `python_module`. This would help us achieve our goal of letting IntelOwl remain a very customizable framework.

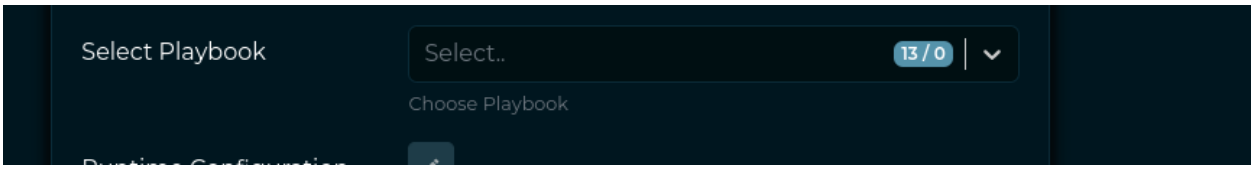
4. Implement urls and view files.

Our endpoints should be like the following:

Endpoint	Method	Description
/api/execute_playbook	POST	Execute Playbooks. Expects parameters `observable_name`, `file` and `playbook`.
/api/job/<job_id>/playbook/<playbook_name>	DELETE, GET	Stop a playbook or Retry the Playbook.
/api/playbook/<str:name>	GET	Get playbook details
/api/playbook/playbook_configuration	GET	Get all playbooks and their details.

5. Adding the frontend support for playbooks.

My plan is to add the following column in `/scan`



And to add support for Playbooks at places like the job results in `/jobs`

This would of course, Include me adding support for Playbook-related frontend components like in tables.jsx, UI changes in the plugin page to add another section for playbooks and job history tab.

6. Adding playbook cache feature

After every scan which uses more than one Analyzer and Connectors, We would present the user with a pop-up that would say “Do you want to save this scan as a Playbook?”

If the user agrees to this, The configuration would be sent to the backend where it would be saved “lazily” in the database. The database model for each entry would somewhat look like this:

```
class CachedPlaybook(models.Model):
  name = models.Charfield(max_length=225)
class CachedPlaybookEntries(models.Model):
  playbook = models.ForeignKey(to=CachedPlaybook, on_delete=models.CASCADE)
  type = models.CharField(max_length=50) # analyzer/connector
  name = models.CharField(max_length=225)
  active = models.BooleanField()
  # ^ if False, then it means that analyzer/connector
  # has been altered
```

This would be in the `models.py`

The result of the cached playbooks would be appended to the JSON returned by `/api/playbook/playbook_configuration`

Some further technical support would be required to be added internally for the other endpoints so that we would be looking for the playbook information in both the playbook JSON file and the cached database entries.

If the playbook entry has an analyzer/connector then we would be marking the cached playbook entry as False so that the entry is ignored.

7. Adding support for Playbooks in PyIntelOwl.

The following methods would be required to be added to the PyIntelOwl SDK and add the same as click commands in the CLI:

Name	Parameters	Description

get_all_playbooks()	None	Would return all the possible playbooks in List[Dict[str, Any]]
kill_playbook()	job_id : int, playbook_name : str	Would kill the playbook processes
retry_playbook()	job_id : int, playbook_name : str	Would retry the playbook.

8. Creating a Playbook with just Free analyzers

As mentioned here by Eshaan, The best way to solve that issue is by adding a playbook with just free analyzers there. I would be doing the same after playbooks have been made.

9. Documentation!

The documentation for a feature must be written with every point being finished.

Timeline

Period	Description	Milestone
Bonding Period (May 20 - June 12)	I bond with the community and understand the code base better.	Bond with the community.
Week 1 (June 13 - June 17)	I work on #1 Implement serializers and dataclasses and #2 Implement the Plugin class (classes file) and controller file.	Initialize Playbooks.
Week 2 (June 20 - June 24)	#4 Implement urls and view files.	Finish Views
Week 3 (June 27 - July 1)	I work on #3 Implement the default `python_module` file	Wrap up the backend.
Week 4 (July 4th - July 8th)	I work on #5 Adding the frontend support for playbooks.	Wrap up adding frontend support
Time till phase #1 Evaluation (July 11th - July 25th)	I am leaving this as some buffer period to wrap up things before Phase 1 evaluation if something was left out. I would also work on #6 Adding playbook cache feature.	Wrap up playbooks by phase #1 Evaluation and

		playbook cache feature.
Phase 1 Evaluation week (July 25th - July 29th)	I would tackle #7 Adding support for Playbooks in PyIntelOwl and #8 Creating a Playbook with just Free analyzers	Wrap up support for PyIntelOwl and Free Analyzers
Week 2 Post first evaluation (August 8th - August 12th)	Wrap up anything that couldn't be wrapped up by this week and implement any feedback accordingly.	Wrap up all my work.

Deliverables

- Playbooks feature.
- A free-only playbook feature consisting of free analyzers.
- Playbook support for the frontend and PyIntelOwl
- Documentation for all of the above.

Why are you well suited to perform this project and why are you interested in it?

- I have industry software development experience which I have picked up while working at SupplyNote as a FullStack developer Intern where I worked on their Point Of Sales Software, Posify. This has helped me become more competent with delivering software and become more flexible with the technology I use. since working here has pushed me to work with different languages and interact with different technologies. My tech stack here is the same as the tech stack used by IntelOwl currently (at least the current master branch) (Angular TypeScript and Python). I have gotten the chance to work on many ongoing projects here. Right now, for example, I am helping us scale up which has been a wonderful learning experience.
- I have a clear understanding of the code-base being a past contributor.
- I have been into security since the final years of my HighSchool. Having found a handful of real-world vulnerabilities myself, I have become more comfortable with pointing out security flaws in code and helping make it more secure.

- I understand where IntelOwl is heading. I have done my due diligence and I clearly understand where the mentors want to take IntelOwl. This is why I proposed this idea.

Have any of our members met you face to face, such as at one of our recent public events? If so, please list who/where.

None as of now.