Cheat Detection in Cyber Security Capture The Flag Games: An Automated Cyber Threat Hunting Approach

Robert Chetwyn – University of Oslo

László Erdődi – Norwegian University of Science and Technology

Purpose

- Implementation of automated cyber threat hunting methods to detect cheating activities of participants in capture-the-flag (CTF) style games.
- Problem:
  - CTF games generate secret flags/tokens on successful completion of a game.
  - Secret flags/tokens are trust-based, the user uploads the flag to confirm completion of a game.
  - Trust-based upload confirms completion of challenge but does not verify the legitimacy of the compromise.
- Research Question:
  - Can automated cyber threat hunting methods be used to verify participant activities in CTF style games.
- What we achieved:
  - An automated log querier that can determine user’s activities as suspicious or benign in network-based CTF games.

Motivation

- Lack of verification is a problematic scenario in academic and industry environments, where plagiarism affects the integrity of the provided courses and participant's certification.
- Example case:
  - In 2019 plagiarism was reported for the Offensive Security Certified Professional (OSCP) exam. Ex-participant produced public write-ups on the OSCP exam challenges, leaking the exam solutions [1].
- Examination 'brain-dumps' (the publishing of exam questions, topics and answers) create a problem with information reuse.
- Participants can reuse the information provided from brain-dumps to complete CTF challenges, skipping pre-requisite steps and submitting the flags.
- With the popularity of using CTF challenges for delivering cyber security training we are motivated to ensure the integrity of this delivery and to monitor each challenge for plagiarism.

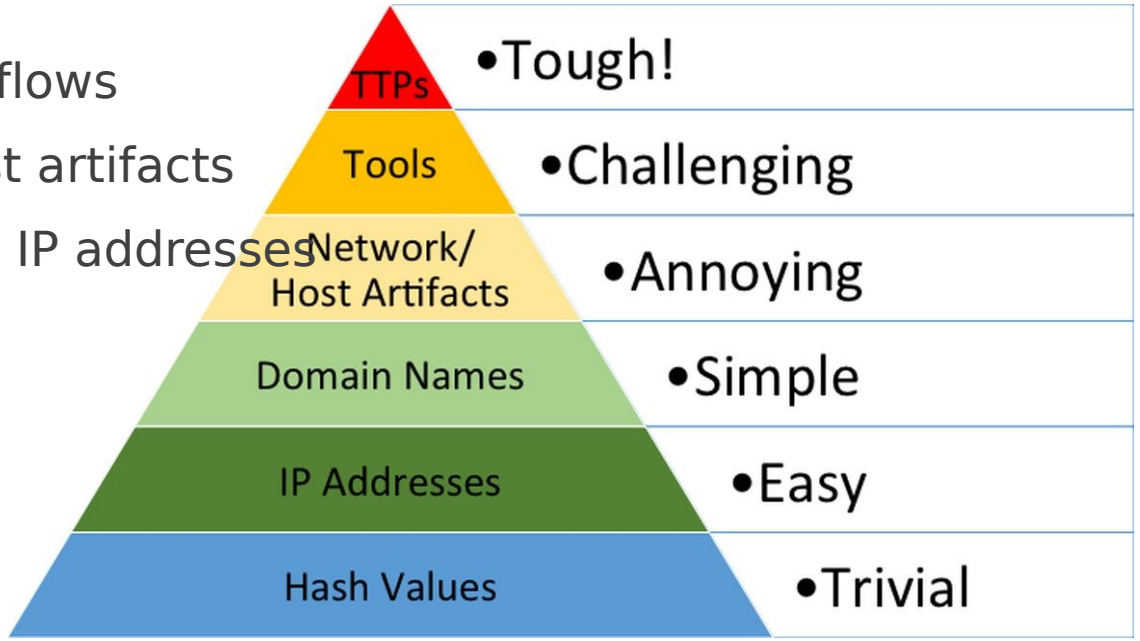
[1] J. Porup, OSCP cheating allegations a reminder to verify hacking skills when hiring, 2019. URL: https://www.csoonline.com/article/3336068/oscp-cheating-allegations-a-reminder-to-verify-hacking-skills-when-hiring.html

Background - CTF Games

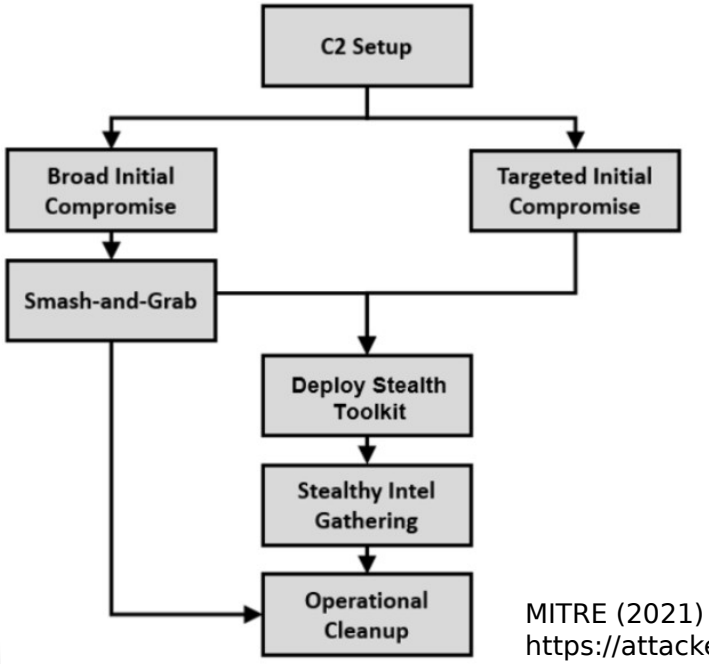
- CTF games present a set of hacking tasks or challenges where each challenge is defined by one vulnerability or a chain of vulnerabilities associated with a secret flag.
- Based on the secret flag an unambiguous criterion is provided for each challenge to decide whether a challenge was solved or not.
- Participants have to carry out attack steps in a particular order. By solving a step the participants might receive new information to achieve the following step. This continues until the challenge is completed.
- Each participant interaction within the CTF game generates security event traces.
- Each game requires a minimum set of challenge steps to be compromised.
- The challenge step dependencies can be transformed into ‘Indicators of Compromise’ (IOCs).
- IOCs are artefacts of forensic evidence that are matched to logged events from the participant’s interactions with the CTF challenges.
- This research utilises the Hacking Arena – Network based CTF platform hosted by UiO/NTNU

Cyber Threat Hunting Method

- Pyramid of Pain
- Adversarial operational flows
- Focused on network/host artifacts
- Also collect anonymised IP addresses



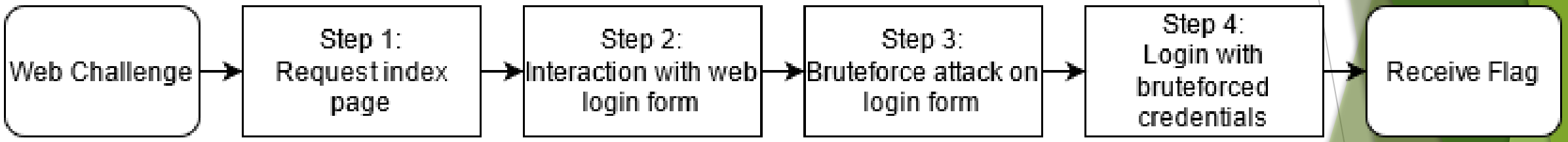
D. Bianco (2014) The Pyramid of Pain
http://detect-respond.blogspot.com/2013/03/the-pyramid-of-pain.html



APT29 Operational Cadences

MITRE (2021) APT 29 Emulation Plan
https://attackevals.mitre-engenuity.org/enterprise/apt29/operational-flow

Example Operational Flow

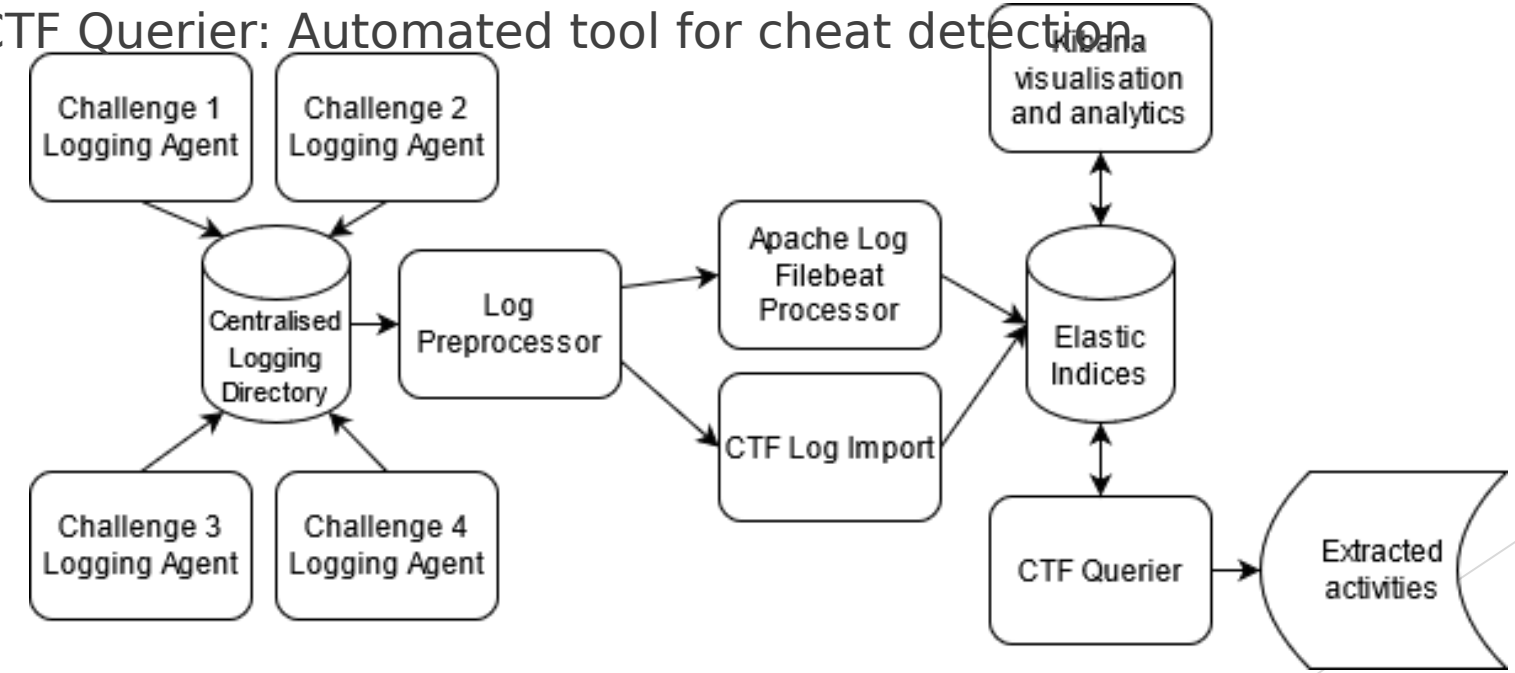


Example Queriable Network IOCs In the infrastructure

- 1. ”match_phrase”: **{”query”: ”*audi”}**
- 2. ”match_phrase”: **{”query”: ”*\\/etc/passwd*”}**
- 3. ”match_phrase”: **{”url.original”:”/index.php?car=php://filter/convert.base64-encode/resource=index.php”}**
- 4. ”match_phrase”: **{”url.original”: ”/loginforusers/index.php”}**
- 5. ”match_phrase”: **{”post”: ”POST: car=’ or position()=3]/*[5]|a[’;\”**

Infrastructure

- Logging Agent: network based logging agent for each challenge.
- Centralised Logging: single source point for retrieving logs.
- Log Preprocessor: applies additional logic to the logs dependent on challenge. Forwards to log importer/filebeat processor.
- Elastic indices: Elasticsearch indices store and index logs for querying.
- CTF Querier: Automated tool for cheat detection.



Network Event Logs

- Environment uses Apache web server logs & custom HTTP logging
- Apache access logs are formatted using the Apache Common log format:
  - • IP Address• Timestamp• HTTP request method (GET/POST)• Status code• Return byte size• Referrer• Web user agent
- Apache error logs are formatting using the default logging
  - Contains any errors returned by the system (e.g. documents unavailable, PHP errors)
- Custom log formatting used for HTTP based user-activities:
  - Timestamp• IP address• Challenge name• Requested page• HTTP GET content• HTTP POST content• Site cookies• Web user agent• Unique ID
- These indexed log elements can be queried for IOCs where the elements can be attributed to a participant fulfilling challenge dependencies.

Automated CTF Querying Tool

- CTF Querier is an automated tool that leverages the Python Elasticsearch Client [1]
- CTF querier searches the Elasticsearch indices for IOCs that are present within a participant’s activities.
- For optimisation and narrowing the search scope the final flag query is used to obtain only those participants who compromised the web challenge.
- IOCs are manually generated and used as input data for CTF querier
- The CTF Querier conducts the following steps:

[1] Elasticsearch https://elasticsearch-py.readthedocs.io/en/v7.15.2

Automated CTF Querying Tool

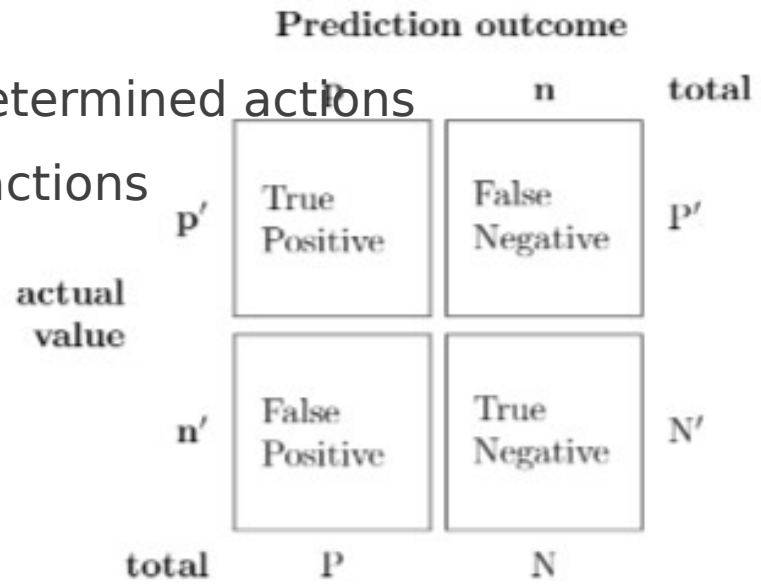
- CTF Querier obtains a list of participants and conducts the following steps:
- 1. Get timestamp of final flag IoC for each participant.
- 2. For each step in a challenge dependency gather the following
  - Get all participants who match the IOC
  - Get initial timestamp of IOC match for each participant
- 3.Check the fulfilment of challenge dependencies for each participant by checking the following:
  - Challenge step complete or missing? If missing:
    - Define the action as suspicious, add to suspicious activity list.
  - Did the challenge step occur before a previous step?
    - Define the action as suspicious, add to suspicious activity list.
- Create a report for the analyst that sums total amount of times a participant is present in the suspicious list with the logged actions.

Testing the method

- We captured anonymised student input data over an 8month period.
- 4 different network based CTF games were chosen.
- Simulated participants took part in each challenge:
  - **Benign Participants**: fulfills the challenges step by step
  - **Malicious Participants:** conducts following activities:
    - Certain dependencies are deliberately skipped or missing.
    - Steps are completed out of sequence.
    - Challenge flag is directly requested.
- Each action is logged by the CTF agents.
- Each action is queried against the challenge dependencies to determine the results of the participants activities.

Testing the method

- We quantify the performance of the CTF querier using a confusion matrix
- **Suspicious** actions are the **positive** determined actions
- **Benign** are the **negative** determined actions



CTF Game 1

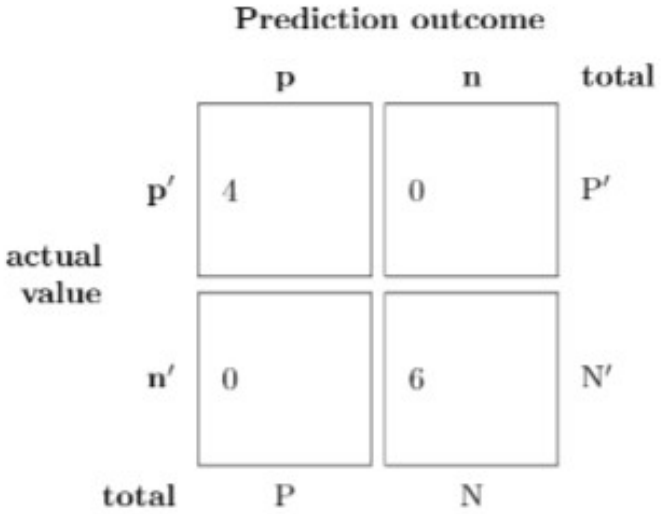
- First challenge is an information disclosure challenge.
- Participant must submit a parameter string disclosed to the web server to retrieve the secret flag.
- The challenge steps:
  - Request the challenge index page.
  - Requests and analyse the **robots.txt** file.
  - Request the excluded '**/Something/**' directory.
  - Request the excluded '**/PennyLane/**' directory.
    - *Requesting '/PennyLane/' presents a web-form.*
  - Submitting **'Hello**' in the '**greeting**' parameter returns the secret flag.
- Suspicious Actor: Skips to stage 4

CTF Game 1 Results

- CTF Querier finds classifies and reports the correct set of actions for all users in game 1

```
Benign Actions: Player X Detected: 5 Times
Benign Actions: Player Y Detected 1 Times

Suspicious Actions: Player X Detected: 0 Times
Suspicious Actions: Player Y Detected: 4 Times
```



CTF Game 2

- Second challenge is a local file inclusion attack
- Participant is required to exploit a local file inclusion vulnerability and conduct an xpath injection attack.
- Challenge Steps:
  - Request the index page of the web challenge.
  - Interacts with the ’car’ web-form parameter, returns a.txt file prefixed with the brand of car the user inputs.
  - Conduct local file inclusion on the car parameter by **requesting:car=/etc/passwd/**
  - Base64 encode the source file of the index.php page string using **parameter:php://filter/convert.base64-encode/resource=index.php** in the ’car’ parameter
  - Request decoded **/logingforusers/** directory
  - Conduct xpath injection on the login form to expose the flag using the string:**’ or position()=3]/*[5]|a[’**
- Suspicious Actor steps:
  - Step 1, Step 5, Step 6

CTF Game 2 Results

- As with Game 1, the CTF querier determines all actions correctly

		Prediction outcome: p	Prediction outcome: n	total
actual value	p′	4	0	P′
	n′	0	8	N′
total		P	N	

Final set of results from all challenges

- In total all games had each participant's actions correctly classified
- CTF querier is able to accurately perform multi-phase event detection and classification of captured web traffic for CTF Games
- A signature based approach produces no false positives for our environment
- No false positives is important for the integrity of the results

		Prediction outcome		
		p	n	total
actual value	p′	14	0	14
	n′	0	24	24
	total	14	24	

Assessment

- CTF querier can verify the authenticity of participant’s actions in CTF Games
- Converting the challenges into IOCs based on the operational flows achieves high accuracy in static network based challenges
- CTF querier can conduct this process in an automated way with minimal user input
- False positives are mitigated by utilising the minimum amount of required steps in an operational flow
- System is limited to network based CTF games only. Host based challenges would require further development for generating operational flows.

Conclusion & Future Work

- Integration with host-based challenges requires more sophisticated operational flows and detection methods.
- We plan to test this environment in the December 2021 Norwegian Cyber Challenge hosted at NTNU
- Implementation machine-learning classifiers of user behaviours to detect and classify unknown operational flows.
- IOCs are manually generated. We plan to explore ways to detect the IOCs from the captured security logs.
- Integration into other SIEM tools via SIGMA rules.