Exploratory Data Analysis

What does the data look like?

- <dataset>
  - gather - (Contains system logs ,located at ‘gather/<host_name>/logs/’)
  - labels - Contains the ground truth of the dataset that indicates which events are related to attacks.
  - rules - contains the label rules
  - processing - Contains the source code that was used to generate the labels.
  - environment - Contains the source code that was used to deploy the testbed and run the simulation

dataset.yml - specifies the start and end time of the simulation.

Data Engineering

type=USER_AUTH msg=audit(1642999060.603:2226): pid=27950 uid=33 auid=4294967295 ses=4294967295 msg='op=PAM:authentication acct="jhall" exe="/bin/su" hostname=? addr=? terminal=/dev/pts/1 res=success' type=USER_ACCT msg=audit(1642999060.603:2227): pid=27950 uid=33 auid=4294967295 ses=4294967295 msg='op=PAM:accounting acct="jhall" exe="/bin/su" hostname=? addr=? terminal=/dev/pts/1 res=success' type=CRED_ACQ msg=audit(1642999060.615:2228): pid=27950 uid=33 auid=4294967295 ses=4294967295 msg='op=PAM:setcred acct="jhall" exe="/bin/su" hostname=? addr=? terminal=/dev/pts/1 res=success' type=USER_START msg=audit(1642999060.627:2229): pid=27950 uid=33 auid=4294967295 ses=4294967295 msg='op=PAM:session_open acct="jhall" exe="/bin/su" hostname=? addr=? terminal=/dev/pts/1 res=success'

{"line": 1860, "labels": ["attacker_change_user", "escalate"], "rules": {"attacker_change_user": ["attacker.escalate.audit.su.login"], "escalate": ["attacker.escalate.audit.su.login"]}} {"line": 1861, "labels": ["attacker_change_user", "escalate"], "rules": {"attacker_change_user": ["attacker.escalate.audit.su.login"], "escalate": ["attacker.escalate.audit.su.login"]}} {"line": 1862, "labels": ["attacker_change_user", "escalate"], "rules": {"attacker_change_user": ["attacker.escalate.audit.su.login"], "escalate": ["attacker.escalate.audit.su.login"]}} {"line": 1863, "labels": ["attacker_change_user", "escalate"], "rules": {"attacker_change_user": ["attacker.escalate.audit.su.login"], "escalate": ["attacker.escalate.audit.su.login"]}}

INSyT

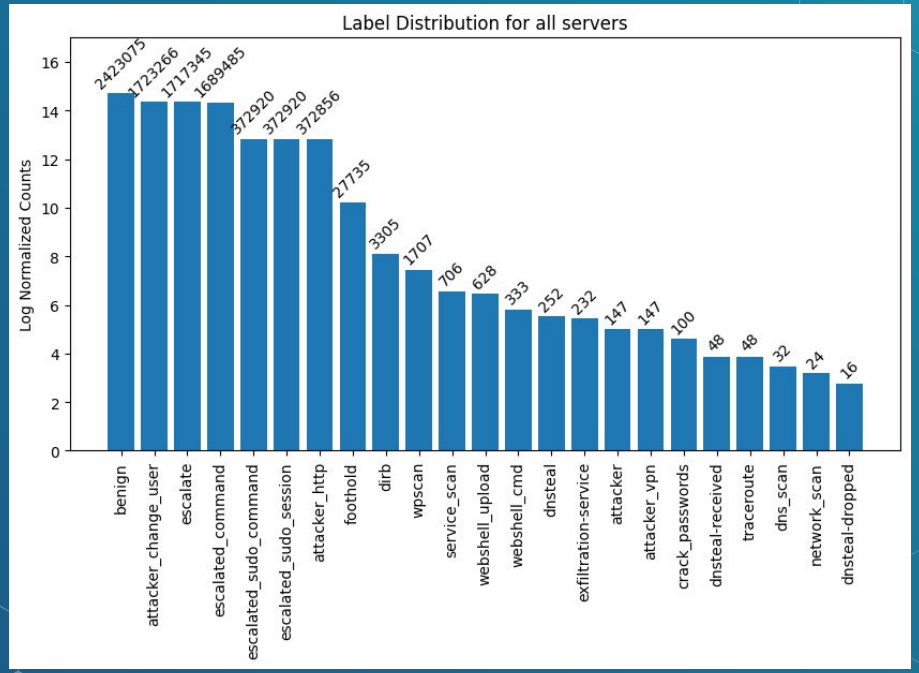
Data Engineering

	line	log_line	labels	rules	gather_path	labels_path	server_name
0	1	Jan 23 06:25:12 intranet-server CRON[27138]: p...	benign	benign	wardbeck/gather/intranet_server/logs/auth.log	wardbeck/labels/intranet_server/logs/auth.log	wardbeck
1	2	Jan 23 06:39:01 intranet-server CRON[27297]: p...	benign	benign	wardbeck/gather/intranet_server/logs/auth.log	wardbeck/labels/intranet_server/logs/auth.log	wardbeck
2	3	Jan 23 06:39:01 intranet-server CRON[27297]: p...	benign	benign	wardbeck/gather/intranet_server/logs/auth.log	wardbeck/labels/intranet_server/logs/auth.log	wardbeck
3	4	Jan 23 06:47:01 intranet-server CRON[27367]: p...	benign	benign	wardbeck/gather/intranet_server/logs/auth.log	wardbeck/labels/intranet_server/logs/auth.log	wardbeck
4	5	Jan 23 06:47:05 intranet-server CRON[27367]: p...	benign	benign	wardbeck/gather/intranet_server/logs/auth.log	wardbeck/labels/intranet_server/logs/auth.log	wardbeck
...	...	...	...	...	...	...	...
329762	299051	Jan 23 23:47:10 dnsmasq[14755]: reply motd.ubu...	benign	benign	wardbeck/gather/inet-firewall/logs/dnsmasq.log	wardbeck/labels/inet-firewall/logs/dnsmasq.log	wardbeck
329763	299052	Jan 23 23:57:15 dnsmasq[14755]: query[AAAA] in...	benign	benign	wardbeck/gather/inet-firewall/logs/dnsmasq.log	wardbeck/labels/inet-firewall/logs/dnsmasq.log	wardbeck
329764	299053	Jan 23 23:57:15 dnsmasq[14755]: forwarded intr...	benign	benign	wardbeck/gather/inet-firewall/logs/dnsmasq.log	wardbeck/labels/inet-firewall/logs/dnsmasq.log	wardbeck
329765	299054	Jan 23 23:57:15 dnsmasq[14755]: nameserver 127...	benign	benign	wardbeck/gather/inet-firewall/logs/dnsmasq.log	wardbeck/labels/inet-firewall/logs/dnsmasq.log	wardbeck
329766	299055	Jan 23 23:57:15 dnsmasq[14755]: reply intranet...	benign	benign	wardbeck/gather/inet-firewall/logs/dnsmasq.log	wardbeck/labels/inet-firewall/logs/dnsmasq.log	wardbeck

329767 rows × 7 columns

INSyT

Initial Label Distribution



INSyT

Rules

- Rules folder contains the description of each labels and the pattern behind each attack that how it was classified

type	id	labels	description
elasticsearch.sequence	attacker.foothold.apache.access	['attacker_http', 'foothold']	This rule matches the attackers recorded HTTP traffic to access log lines based on the web paths. Note that we prefix the web paths with the servers FQDN in the parsing phase to match the PCAP records.
elasticsearch.sequence	attacker.foothold.pcap.requests	['attacker_http', 'foothold']	This rule matches attacker HTTP request pcap logs with their HTTP responses when the response has already been marked as attacker traffic. We do this so we can apply followup rules only on requests that have not been marked yet. Such cases can happen when the response did not make it to the attacker.
elasticsearch.sub_query	attacker.foothold.apache.access_dropped	['attacker_http', 'foothold']	This rule tries to match attacker requests that we where unable to match to a labeled response with access log entries. Such cases can happen if the corresponding response gets lost in the network or otherwise is not sent.
elasticsearch.sequence	attacker.foothold.apache.error	['attacker_http', 'foothold']	This rule matches the attackers recorded HTTP traffic to error log lines based on the web server file the error occurred for. We do this by converting the reported server file paths to web paths in the parsing phase.
elasticsearch.sequence	attacker.foothold.apache.access_error	['attacker_http', 'foothold']	This rule matches already identified access lines to error log lines.
elasticsearch.sub_query	attacker.foothold.apache.error_substring	['attacker_http', 'foothold']	Apache error logs sometimes do not have an url.full attribute, especially events related to the WPScan. Therefore they cannot be labeled by grouping access and error logs with the same url.full value. This rule tries to resolve this issue by checking whether the url occurs in other attributes where it could appear depending on the type of the error event. Note that url.original is used since only this part sometimes occurs rather than the full url. Also note that both regexp and match_phrase is used as elasticsearch sometimes analyzes fields which prevents simple regex matching.
elasticsearch.sub_query	attacker.foothold.apache.error_index	['attacker_http', 'foothold']	This rule applies the attacker http label to all errors produced by requests to directories that map to an index.php file. This is done by first getting all labeled directory requests and then searching for matching error lines with index.php prefixed.
elasticsearch.parent_query	attacker.foothold.apache.error_access	['attacker_http', 'foothold']	This rule looks for unlabeled error messages resulting from VPN server traffic within the attack time and tries to match it to an already labeled access log row.
elasticsearch.parent_query	attacker.foothold.apache.php_warns	['attacker_http', 'foothold']	This rule tries to apply missing labels to error lines that result from multiple PHP errors/warnings in a single request. First all PHP errors that are potentially from the attacker are retrieved and then we try to find a labeled parent by matching pid, port, url.full and the approximate error time.
elasticsearch.query	attacker.dirb.time	['dirb']	This labels attacker http traffic within the recorded dirb execution time with dirb.
elasticsearch.query	attacker.wpscan.time	['wpscan']	This labels attacker http traffic within the recorded wpscan execution time with wpscan.

Problem with Rules Table

type	id
elasticsearch.sequence	attacker.foothold.apache.access
elasticsearch.sequence	attacker.foothold.pcap.requests
elasticsearch.sub_query	attacker.foothold.apache.access_dropped
elasticsearch.sequence	attacker.foothold.apache.error
elasticsearch.sequence	attacker.foothold.apache.access_error
elasticsearch.sub_query	attacker.foothold.apache.error_substring
elasticsearch.sub_query	attacker.foothold.apache.error_index
elasticsearch.parent_query	attacker.foothold.apache.error_access
elasticsearch.parent_query	attacker.foothold.apache.php_warns
elasticsearch.query	attacker.dirb.time
elasticsearch.query	attacker.wpscan.time
elasticsearch.sequence	attacker.webshell.upload.seq
elasticsearch.query	attacker.escalate.webshell.cmd.http
elasticsearch.query	attacker.escalate.webshell.cmd.http_prepare_
elasticsearch.sequence	attacker.escalate.su.login
elasticsearch.sequence	attacker.escalate.systemd.newsession.before
elasticsearch.sequence	attacker.escalate.systemd.newsession.after
elasticsearch.sequence	attacker.escalate.systemd.session
elasticsearch.sequence	attacker.escalate.sudo.command
elasticsearch.sequence	attacker.escalate.sudo.open
elasticsearch.sequence	attacker.escalate.audit.su.login
elasticsearch.sequence	attacker.escalate.audit.systemd.session
elasticsearch.sequence	attacker.escalate.audit.sudo.command.start
elasticsearch.sequence	attacker.escalate.audit.sudo.command.events
elasticsearch.sequence	attacker.escalate.audit.sudo.command.events
elasticsearch.query	dnsteal.domain.match
elasticsearch.query	dnsteal.domain.received
elasticsearch.query	dnsteal.domain.dropped
elasticsearch.parent_query	dnsteal.domain.dropped-retry
elasticsearch.query	attacker.foothold.dnsmasq.wpscan
elasticsearch.query	attacker.foothold.dnsmasq.webshell
elasticsearch.query	attacker.foothold.dnsmasq.dns_scan_probe

- 48+ Different Types of Attacks made it harder to categorize and make Large Language Model to train to categorize them

INSyT

Solution: Merged Labels

type	Merged Label	id	labels	description
elasticsearch.sequence		attacker.foothold.a	['attacker_http', 'footl	This rule matches the attackers recorded **HTTP** traffic to
elasticsearch.sequence		attacker.foothold.(	['attacker_http', 'footl	This rule matches attacker **HTTP** request pcap logs with
elasticsearch.sub_query		attacker.foothold.a	['attacker_http', 'footl	This rule tries to match attacker requests that we where (
elasticsearch.sequence		attacker.foothold.a	['attacker_http', 'footl	This rule matches the attackers recorded **HTTP** traffic to
elasticsearch.sequence		attacker.foothold.a	['attacker_http', 'footl	This rule matches already identified access lines to error
elasticsearch.sub_query	HTTP Attack	attacker.foothold.a	['attacker_http', 'footl	Apache error logs sometimes do not have an url.full attrib
elasticsearch.sub_query		attacker.foothold.a	['attacker_http', 'footl	This rule applies the attacker **http** label to all errors prod(
elasticsearch.parent_query		attacker.foothold.a	['attacker_http', 'footl	This rule looks for unlabeled error messages resulting fro
elasticsearch.parent_query		attacker.foothold.a	['attacker_http', 'footl	This rule tries to apply missing labels to error lines that re
elasticsearch.query		attacker.dirb.time	['dirb']	This labels attacker **http traffic** within the recorded dirb (
elasticsearch.query		attacker.wpscan.t	['wpscan']	This labels attacker **http traffic** within the recorded wpsc
elasticsearch.sequence		attacker.webshell.	['webshell_upload']	This rule labels the web shell upload step by matching th
elasticsearch.query	Web Shell Request	attacker.escalate.'	['webshell_cmd']	This rule matches the web shell web requests via the rec
elasticsearch.query		attacker.escalate.'	['webshell_cmd', 'esc	This rule matches the web shell web requests via the rec
elasticsearch.sequence		attacker.escalate.(	['attacker_change_us	This rule labels auth log rows resulting from the attacker

- Merged a similar attack type to the “merged label” in order to make a categorization easier

INSyT

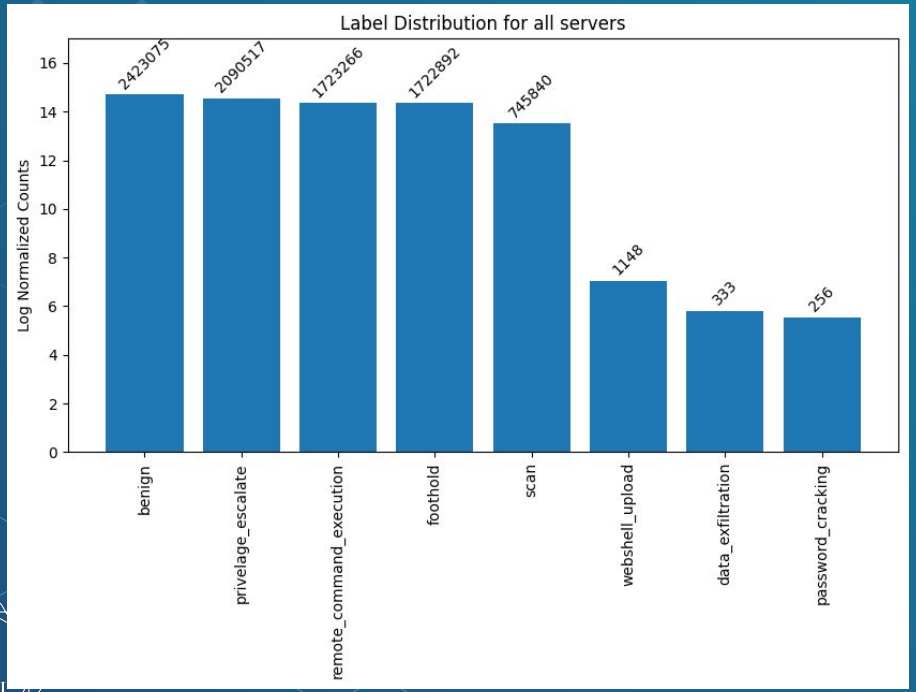
Reduce Labels by Grouping

48+ 

- Benign
- Privilege Escalation
- Remove Command Execution
- Foothold
- Webshell Upload
- Data Exfiltration
- Password Cracking

INSyT

Refined Label Distribution



$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

$$Accuracy = \frac{True\ Positives}{True\ Positives + False\ Positives + True\ Negatives + False\ Negatives}$$

$$F1\ Score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

INSyT

Initial N-Gram

```
Top 100 3-grams:
("'", ':', '{'): 51840 occurrences
(':', '{', '```'): 51840 occurrences
('}', ',', "'"): 32640 occurrences
('{', '```', 'pct'): 32640 occurrences
('```', 'pct', "'"): 32640 occurrences
("'", ':', "'"): 26880 occurrences
("'", ',', "'"): 17280 occurrences
('norm', "'", ':'): 17280 occurrences
(',', "'", 'norm'): 13440 occurrences
("'", 'norm', "'"): 13440 occurrences
('}', '}', ','): 11928 occurrences
("'", '}', ','): 9600 occurrences
('pct', "'", ':0'): 9084 occurrences
("'", 'system', "'"): 7680 occurrences
(':', "'", 'intranet-server'): 5760 occurrences
("'", 'intranet-server', "'"): 5760 occurrences
(',', "'", 'name'): 5760 occurrences
("'", 'name', "'"): 5760 occurrences
('name', "'", ':'): 5760 occurrences
('version', "'", ':'): 5760 occurrences
("'", ':0', '}'): 5557 occurrences
('}', '}', '}'): 5352 occurrences
('intranet-server', "'", ','): 3840 occurrences
('type', "'", ':'): 3840 occurrences
(':', "'", 'system'): 3840 occurrences
(',', "'", '@'): 3840 occurrences
('{', '```', 'cpu'): 3840 occurrences
('```', 'cpu', "'"): 3840 occurrences
('cpu', "'", ':'): 3840 occurrences
(',', "'", 'system'): 3840 occurrences
('system', "'", ':'): 3840 occurrences
('{', '```', 'norm'): 3840 occurrences
```

- N Gram was initially used to find the most frequency of pattern, but it didn't return meaningful data since it's the log file that it returned some format instead of the pattern

INSyT

Advanced N-Gram

- Bronze Live Code Demonstration

Discussion

- Because the data is so unstructured, it is hard to perform meaningful classical EDA
- Data cleaning and Modern NLP techniques such as LLM-tokenization allowed us to gain more meaningful insights

INSyT

Questions?