Scalable Data Analytics Pipeline for Real-Time Attack Detection; Design, Validation, and Deployment in a Honeypot Environment

Eric Badger
Master's Student
Computer Engineering

ECE ILLINOIS

ILLINOIS

Overview

- Introduction/Motivation
- Challenges
- Pipeline Design
- Pipeline Deployment
- Validation of Alerts and Attack Detection Tools
- Future Work
- Conclusion

Research Problem

Our goal is to detect potential attacks as early as possible. Security analysts attempt to detect and prevent attacks, but they can't analyze everything in their infrastructure by hand. They need tools to automate the analysis for early detection of attacks.

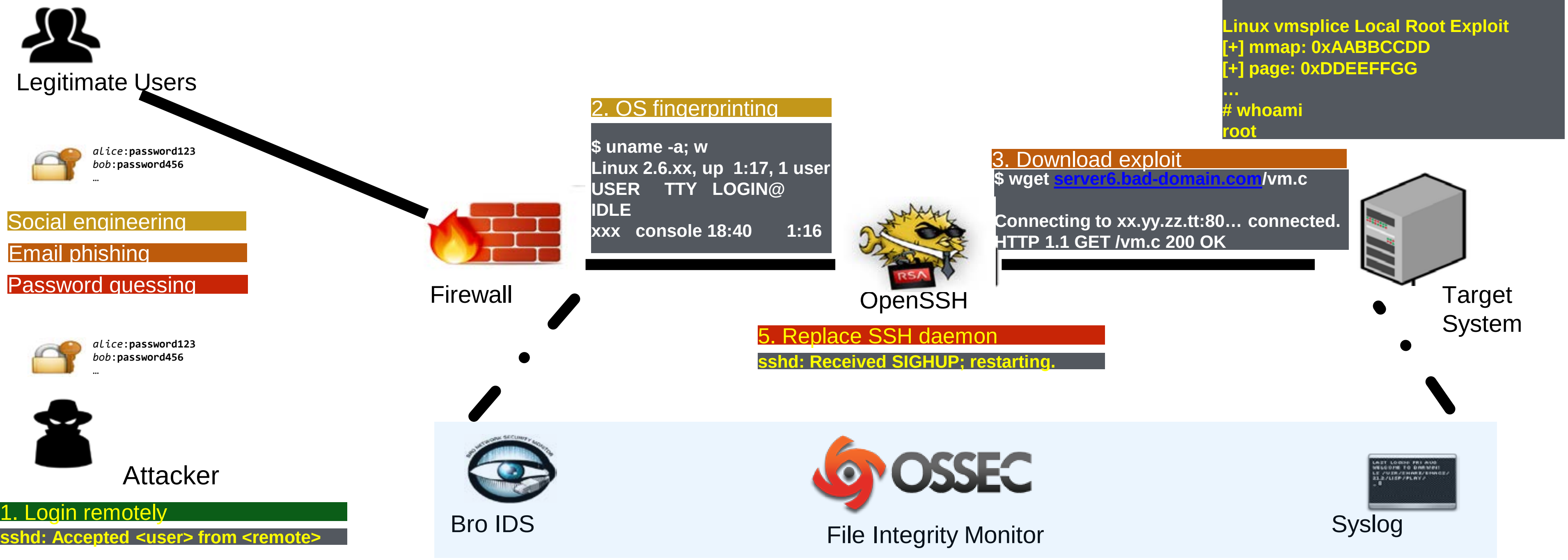
- How do we transition attack detection models from theory to practice?
- How do we validate that the alerts we are using are useful?
  - Does combining alerts from different monitors make the attack detection better?
  - Is the extra performance overhead worth it?
- How do we validate that our attack detection model is adequate and better than others?
  - What models are suitable for real-time attack detection in practical deployment?

Research Challenges:
Transitioning Attack Detection from Theory to Practice

- Identifying which alerts are useful for attack detection
- Normalizing all logs to a common format
- Achieving both high-accuracy and real-time attack detection
- Achieving high-accuracy attack detection in the face of alert randomness, noise, and imperfect monitors
- Scaling the data pipeline

  The chain of tools used for data-driven attack detection

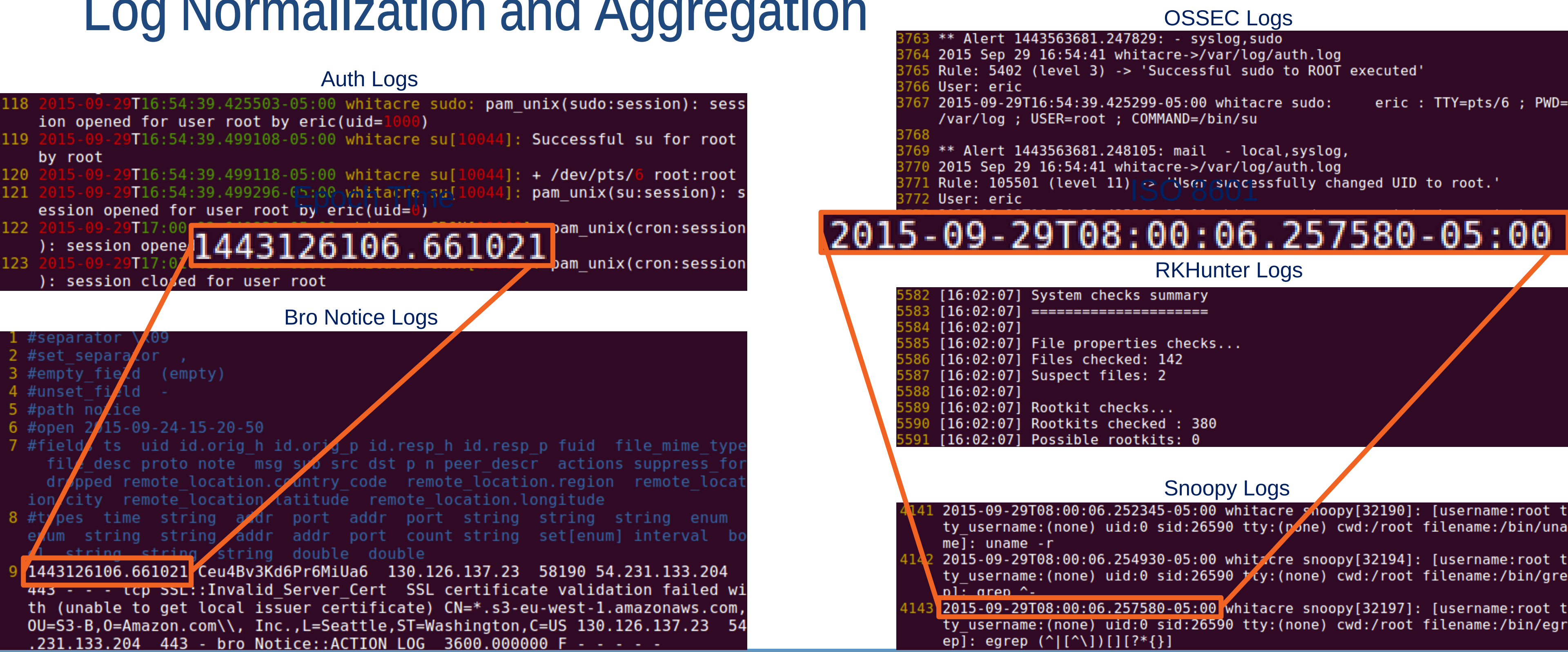
Example Attack Scenario



How to Extract Important Alerts

- **Network Monitors**
  - Bro
    - Network IDS used for packet analysis
  - CriticalStack Intel Feed
- **Host Monitors**
  - OSSEC
    - Runs periodic system checks and file integrity monitoring
    - Aggregates and correlates all other host alerts
  - Snoopy Logger
    - Logs all execv system calls
  - RKHunter
    - Searches for rootkits, hidden folders/files/ports, and other system issues
  - Syslogs
    - Normal GNU/Linux “/var/log” logs, such as auth.log, kern.log, dpkg.log, and others
  - Bash Logs
    - Logs Bash history as the commands are executed

Log Normalization and Aggregation



Log Normalization and Aggregation (2)

- Since the logs are all in different formats, they need to be normalized to a common format

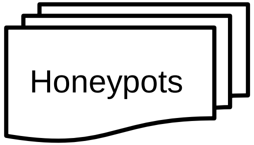
Normalized Log

```
1443457145.717,143.219.0.11:root,ALERT_FAILED_PASSWORD,NaN,NaN,1443461619.483
1443457147.510,143.219.0.11:root,ALERT_FAILED_PASSWORD,NaN,NaN,1443461619.490
1443457661.505,143.219.0.11:LOGIN,login,NaN,NaN,1443461619.516
1443457661.469,143.219.0.11:root,read_host_configuration,NaN,NaN,1443461619.520
1443457662.963,143.219.0.11:ubuntu,ALERT_GET_LOGGEDIN_USERS,NaN,NaN,1443461619.536
1443461754.305,:,ALERT_INTERNAL_ADDRESS_SCAN,NaN,NaN,1443461764.436
```

- All logs needed to be centralized so that we can act on them
- TLS/SSL encryption is necessary to secure the movement of logs through the pipeline
  - If not, the logs could be added, deleted, or changed by a MITM attack

Pipeline Design

Example Tools



Generic Tools

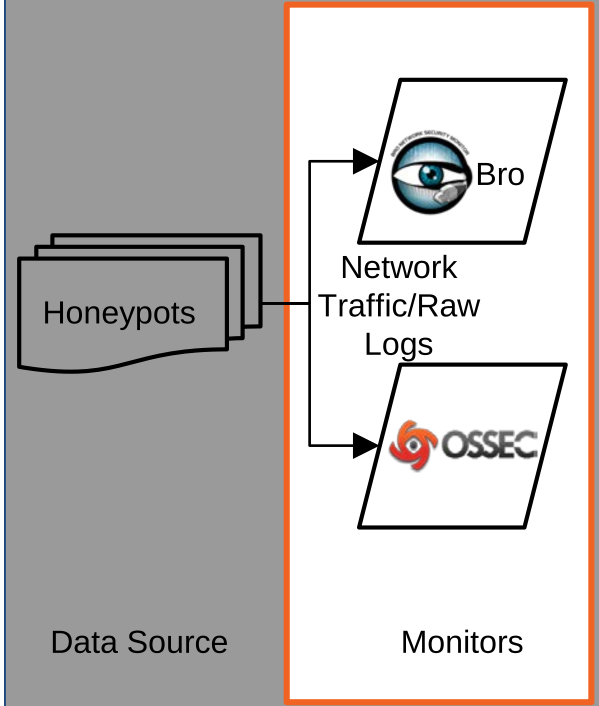
Data Source

- The Data Source can be any sort of online or offline computer or device
  - Online
    - Honeypots, servers, workstations, phones
  - Offline
    - Security testbed, old logs
- We use customized honeypots deployed at the NCSA

Pipeline Design

Example Tools

Generic Tools

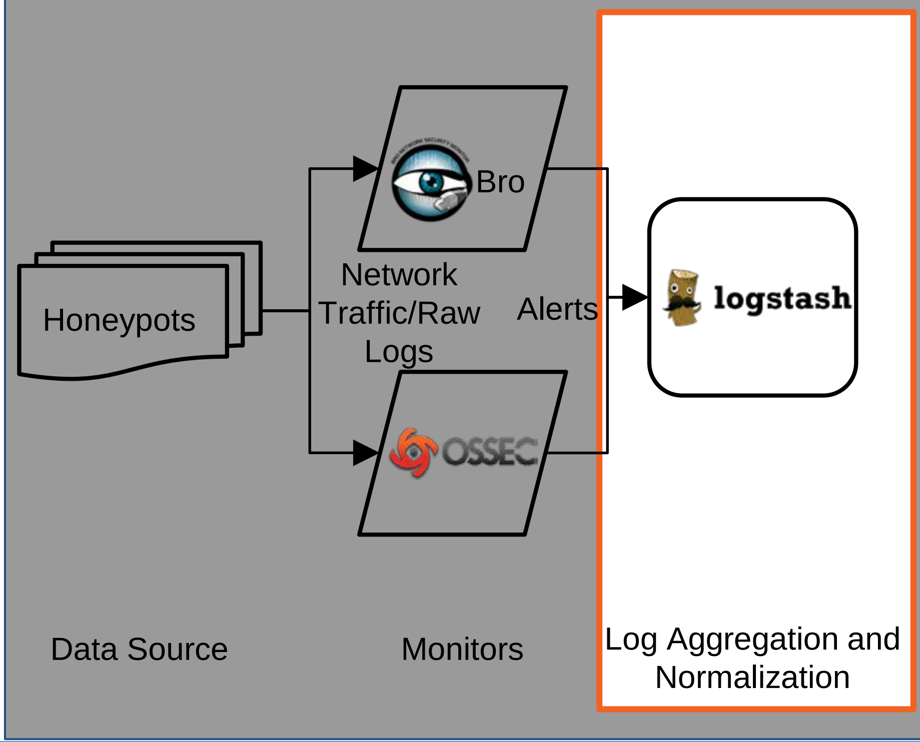


- The Monitors take in data and create alerts
  - The data can be logs, network traffic, or anything that can be alerted on
  - We use Bro, OSSEC, Snoopy Logger, RKHunter, Syslogs, and Bash logs

Pipeline Design

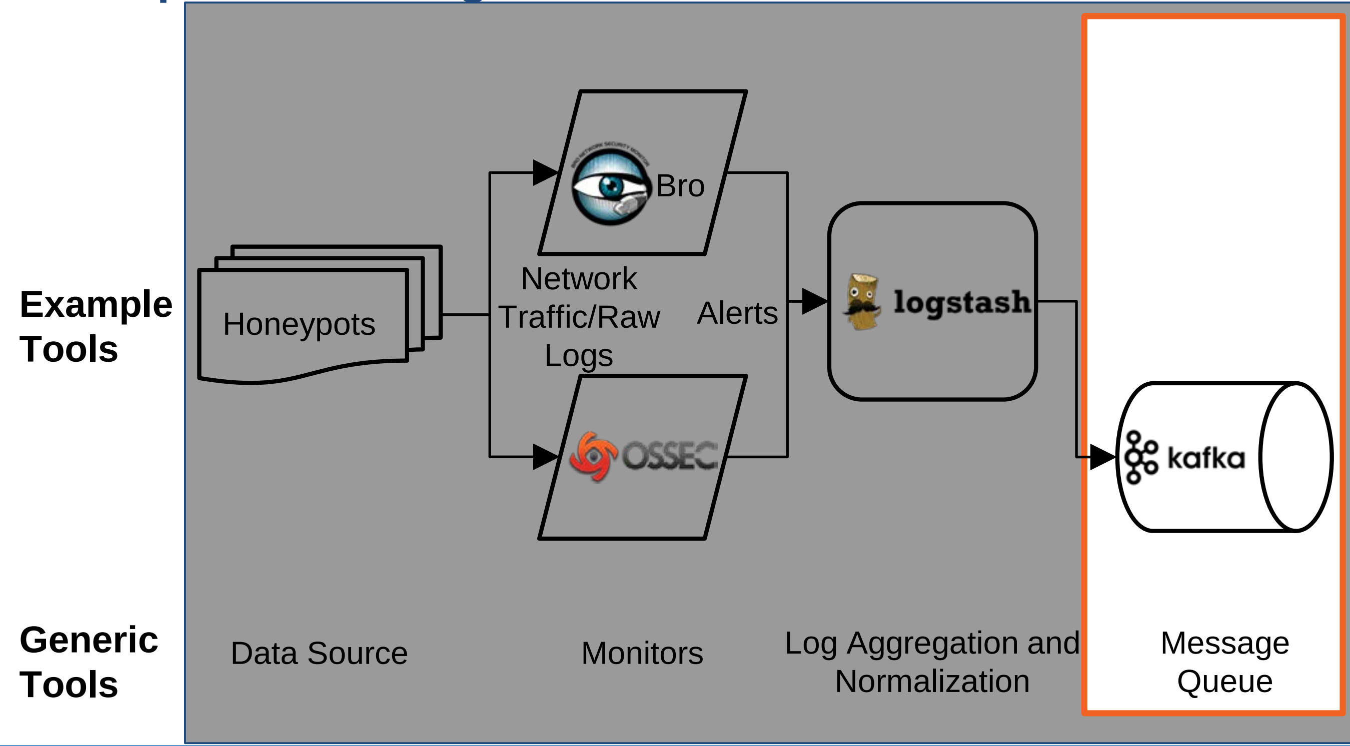
Example Tools

Generic Tools



- The Log Aggregation and Normalization takes in alerts from multiple different inputs and normalizes them to a common format
- We use Logstash as the Log Aggregation tool
- We use Logstash filters to do the Log Normalization
- Logstash has integration with many other tools and has a large community base

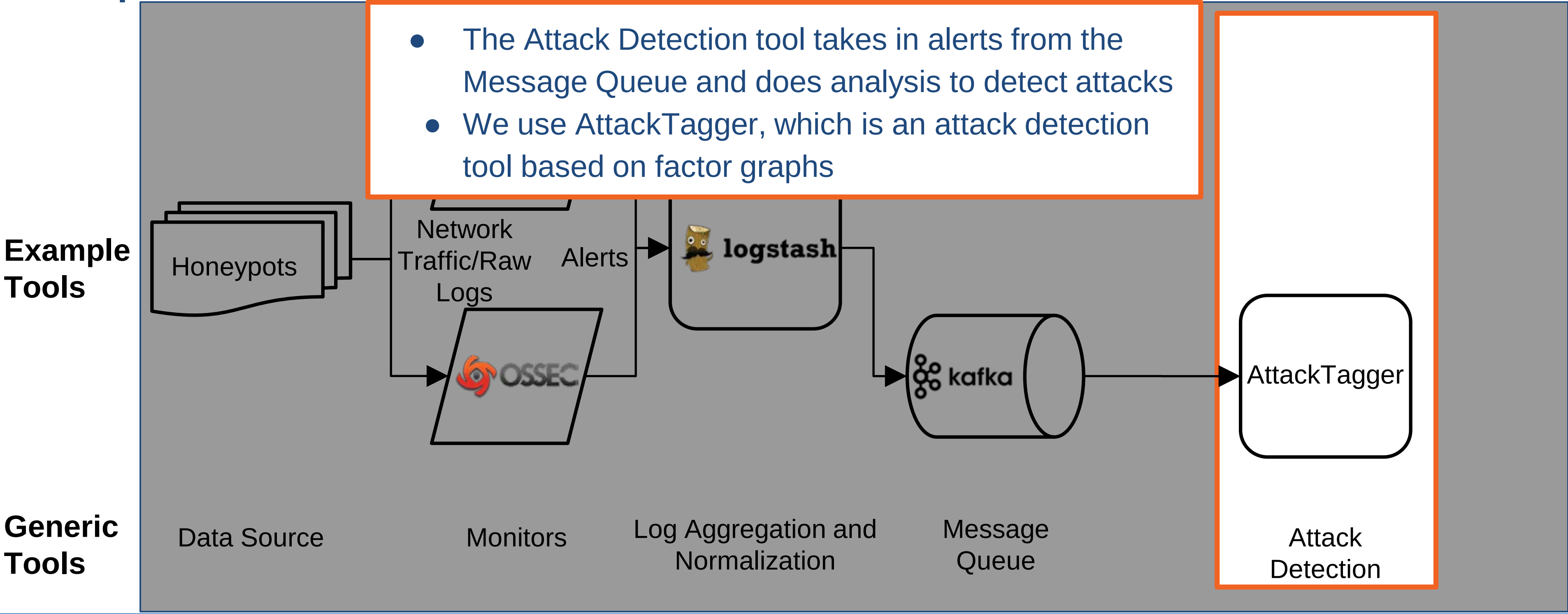
Pipeline Design



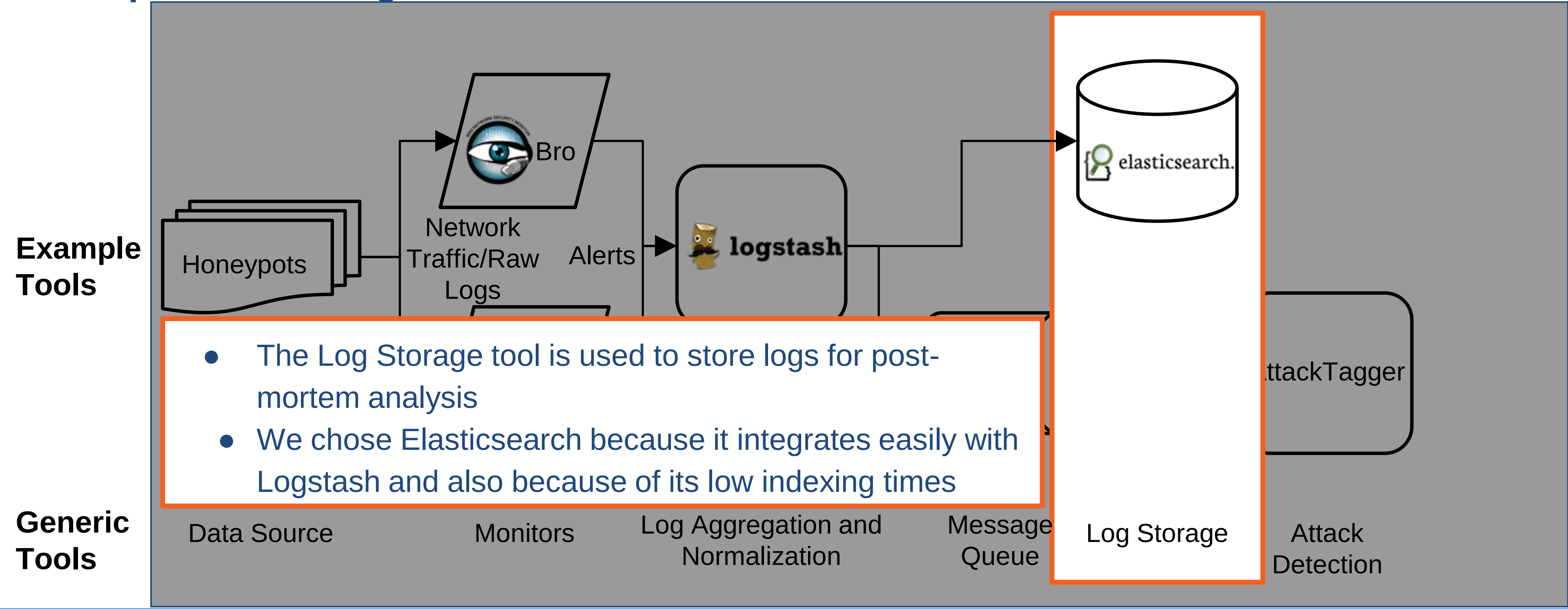
- The Message Queue deals with fluctuations in the throughput of alerts
- This prevents alert loss
- We use Kafka, because it is horizontally scalable and high-throughput

Pipeline Design

- The Attack Detection tool takes in alerts from the Message Queue and does analysis to detect attacks
- We use AttackTagger, which is an attack detection tool based on factor graphs

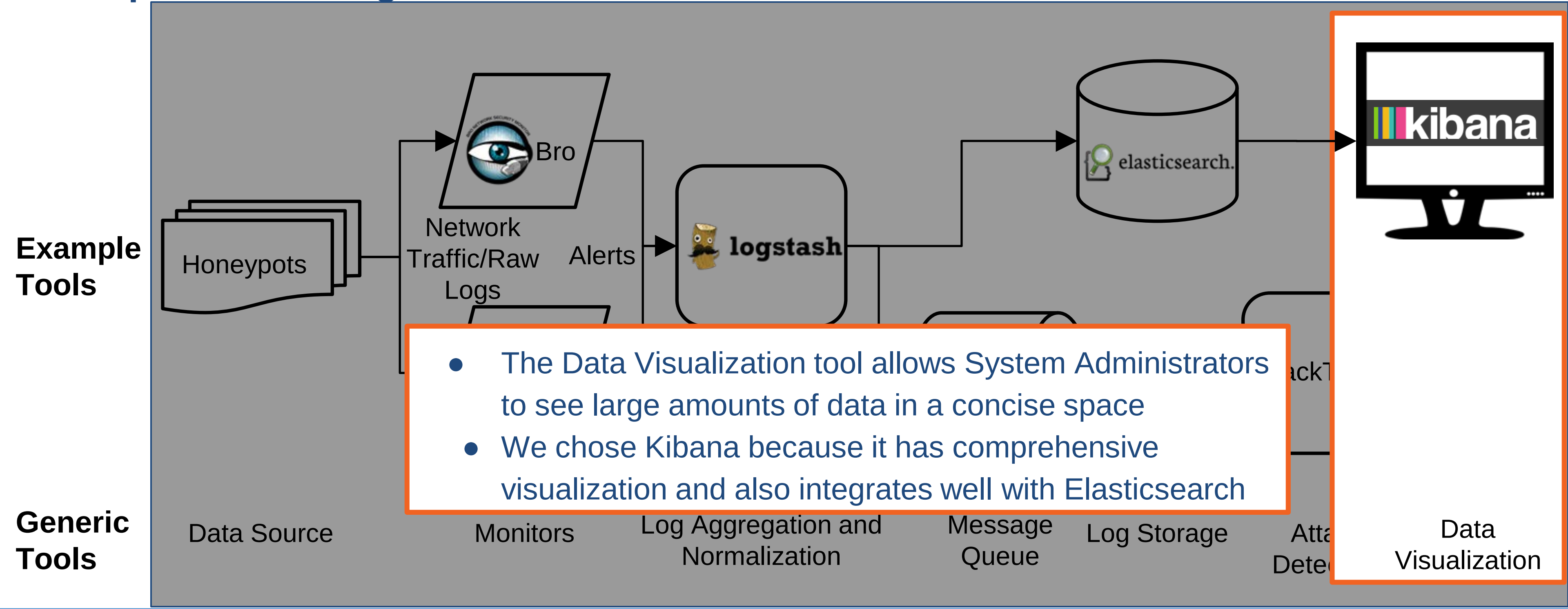


Pipeline Design



- The Log Storage tool is used to store logs for post-mortem analysis
- We chose Elasticsearch because it integrates easily with Logstash and also because of its low indexing times

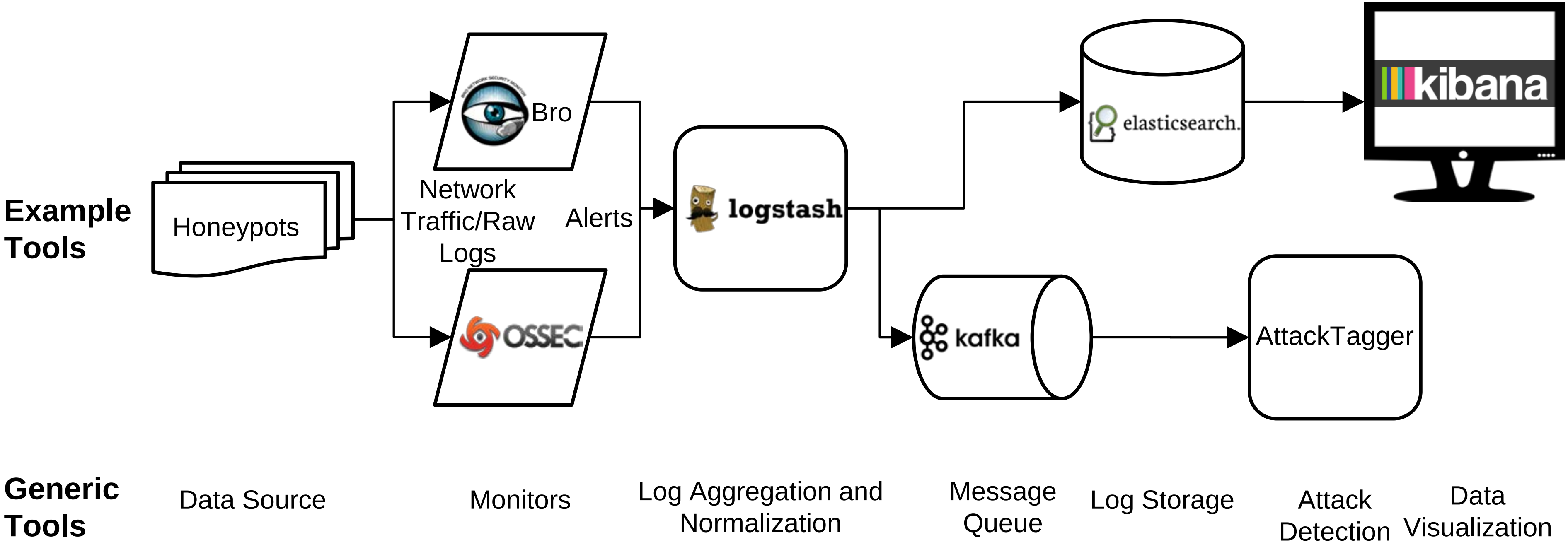
Pipeline Design



- The Data Visualization tool allows System Administrators to see large amounts of data in a concise space
- We chose Kibana because it has comprehensive visualization and also integrates well with Elasticsearch

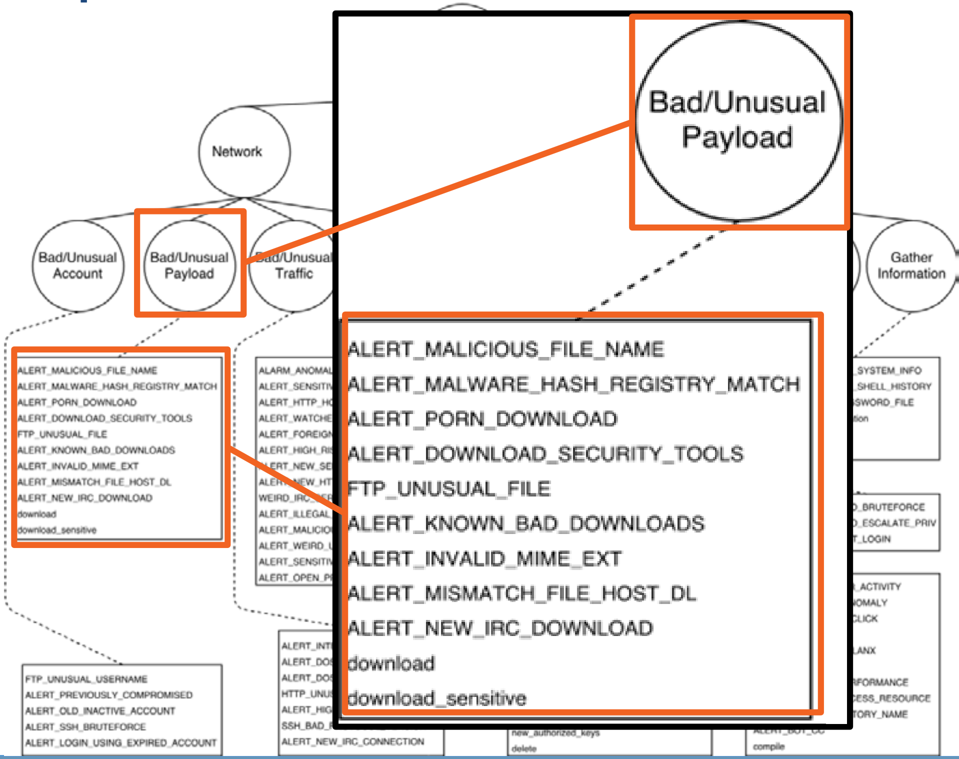

kibana
Discover
Visualize
Dashboard
Settings
November 23rd 2012, 07:33:18.582 to May 17th 2014, 10:37:14.548
LAPD: Crime Dashboard
LAPD: Tile map
Burbank
Glendale
Pasadena
Arcadia
Azusa
San Gabriel
West Hollywood
El Monte
West Covina
Pomona
LOS ANGELES
Brentwood
Culver City
Montebello
Whittier
Chino
Inglewood
Downey
La Habra
Norwalk
Compton
Fullerton
Torrance
1 – 2540
2540 – 5079
5079 – 7618
7618 – 10157
10157 – 12696
Leaflet | Tiles by MapQuest — Map data © OpenStreetMap contributors, CC-BY-SA
LAPD: Crime Type by Area
Legend
77th Street
Southwest
N Hollywood
Pacific
Mission
Southeast
LAPD: Crime Status by Crime Type
Legend
TRAFFIC DR #
BATTERY - SIMPLE...
THEFT PLAIN - PET...
BURGLARY FROM ...
BURGLARY
VEHICLE - STOLEN
LAPD: Monthly histogram by Crime Type
Count
date_rptd per month
2013-03-31
2013-06-30
2013-09-30
2013-12-31
2014-03-31
Legend
TRAFFIC DR #
BATTERY - SIMPLE...
SPOUSAL(COHAB) ...
ASSAULT WITH DE...
THEFT PLAIN - PET...
BURGLARY FROM ...
BURGLARY
VEHICLE - STOLEN
THEFT OF IDENTITY
THEFT-GRAND (OV...
DOCUMENT FORG...
LAPD: Intersections
Count
Top 5 location
WESTERN AV
VERMONT AV
SUNSET BL
6TH ST
SHERMAN WY
Legend
3RD ST
VERNON AV
WASHINGT...
WILSHIRE BL
BEVERLY BL
WESTERN AV
LA BREA AV
GOWER ST
ALVARADO ST
DETROIT ST
SAN PEDRO

Pipeline Design



How Do I Know What Alerts Are Important?

- Research was done in [1] and [2] that studied attacks over a six-year period at NCSA. This research identified important alerts related to these attacks and developed the AttackTagger detection tool
- We utilized and extended a custom set of monitors to create alerts corresponding to the inputs that were used in AttackTagger
- In essence, we brought AttackTagger from a theoretical tool to actual deployment



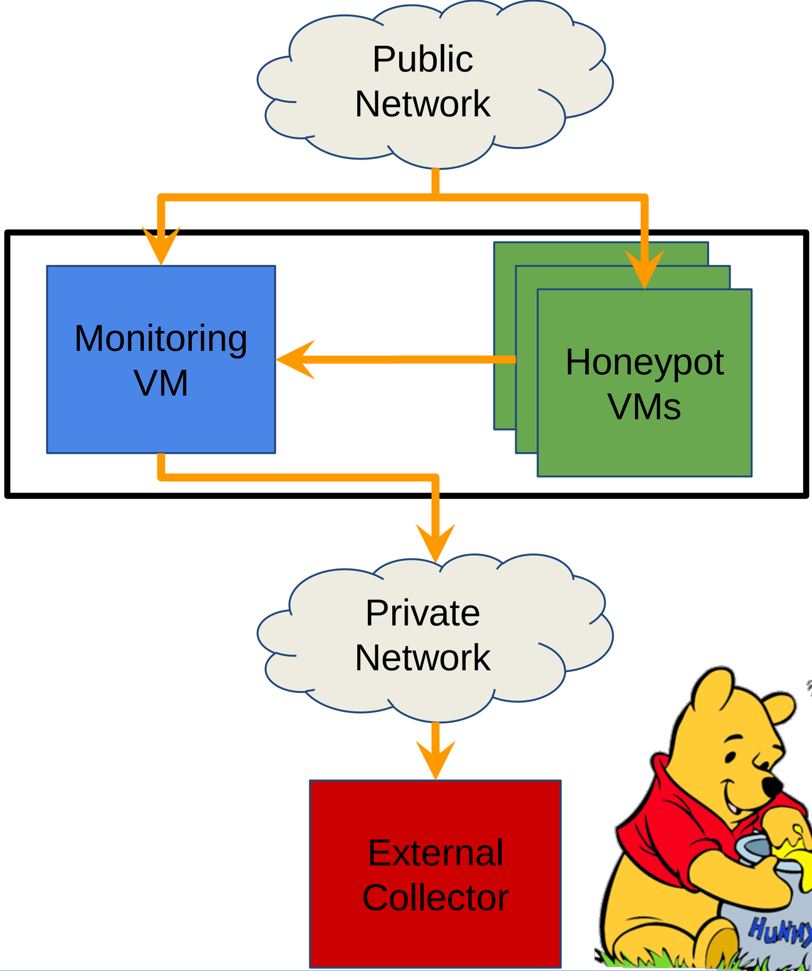
[1] Phuong Cao, Key-whan Chung, Zbigniew Kalbarczyk, Ravishankar Iyer, and Adam J. Slagell. 2014. Preemptive intrusion detection. HotSoS '14.
[2] Phuong Cao, Eric Badger, Zbigniew Kalbarczyk, Ravishankar Iyer, and Adam Slagell. 2015. Preemptive intrusion detection: theoretical framework and real-world measurements. HotSoS '15.

What Can We Do with This Pipeline?

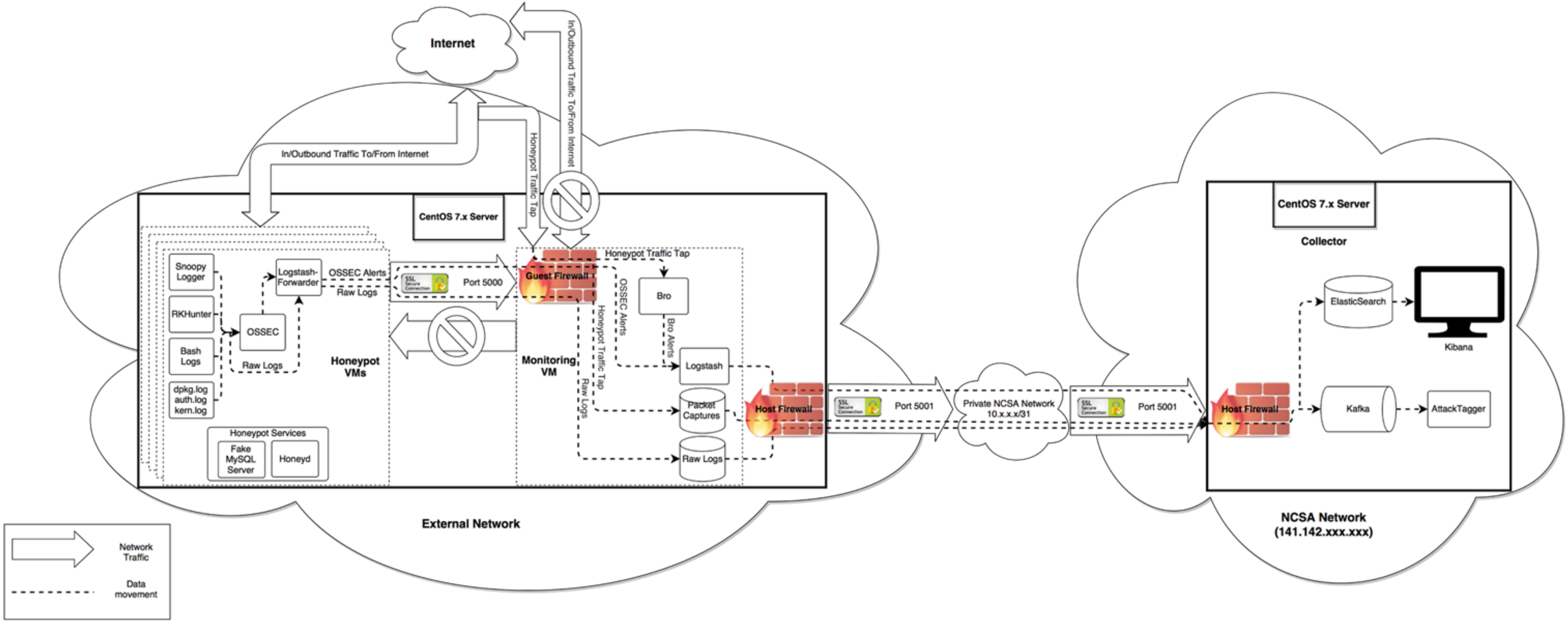
- Both online and offline deployment
- Online
  - Analysis of attacks happening on the infrastructure
  - Analysis of attack detection tools on live data
- Offline
  - Post-mortem log analysis (via Elasticsearch/Kibana)
  - Analysis of old attacks
  - Development of attack detection tools
  - Validation of alerts

Honeypots at NCSA

- NCSA server running several VMs
  - Honeypot VMs
    - Open to public
  - Monitoring VM
    - Allows TCP Port 5000 (Logstash) from honeypots
    - Allows TCP Port 22 from NCSA, UI, and UI wireless
    - Sends logs to Collector via Private Network
- Collector
  - Allows TCP Port 5001 (Logstash) from private network
  - Allows TCP Port 22 from NCSA, UI, and UI wireless



Honeypots at NCSA (2)



Preliminary Honeypot Results

- 3 separate SSH bruteforce attacks successfully compromised one of the honeypots in the first 3 days
- Appeared to download and execute either an open proxy or a DDoS attack through the program “/tmp/squid64”
- They beat my monitors! (Well, sort of...)
  - They pushed their malware from the anomalous host instead of pulling it from the honeypot
  - They deleted the malware immediately after running it, so it was not seen by OSSEC’s file-integrity monitoring

How to Validate Importance of Inputs (Alerts)

- Mix and match which monitors/alerts that we use in our attack detection
- Evaluate the difference in attack detection coverage and accuracy
  - Adding monitors/alerts will likely add detection coverage because of extra data
  - Adding monitors/alerts could possibly decrease detection accuracy because of additional noise
- Determine whether the difference in detection coverage is worth the additional overhead

How to Validate Accuracy of Outputs (Detection Tools)

- Compare and contrast different attack detection tools
  - e.g. Factor Graphs, Bayesian Networks, Markov Random Fields,
    - Signature Detection, etc.
  - Which are most accurate?
  - Which are least complex?
- In the pipeline, attack detection tools are plug and play as long as they can read the normalized alert format
  - If they can't, a translation filter can be added

Future Work

- Validate data pipeline inputs (alerts) and outputs (attack detection tools)
- Add additional data types to data pipeline
  - Netflows
  - Full file-integrity monitoring (e.g. Tripwire)
  - Administrator-generated alerts/profiles
  - Keystroke data (e.g. iSSHD)
- Convert detection model into a stream-processing system
  - Detectors such as AttackTagger are currently batch processing detectors
  - We need to process the data in real-time
- Transition entire pipeline into practice at NCSA production system

Conclusion

- Showed how to transition attack detection software from theory to practice
- Showed how to evaluate the effectiveness of the inputs (alerts) and outputs (attack detection tools) of the pipeline
- Identified challenges and how to overcome them

Special Thanks

- Phuong Cao
- Alex Withers
- Adam Slagell
- NCSA
- DEPEND Research Group

Questions?

Citations

[1] Phuong Cao, Key-whan Chung, Zbigniew Kalbarczyk, Ravishankar Iyer, and Adam J. Slagell. 2014. Preemptive intrusion detection. In *Proceedings of the 2014 Symposium and Bootcamp on the Science of Security* (HotSoS '14). ACM, New York, NY, USA, , Article 21 , 2 pages. DOI=10.1145/2600176.2600197 http://doi.acm.org/10.1145/2600176.2600197

[2] Phuong Cao, Eric Badger, Zbigniew Kalbarczyk, Ravishankar Iyer, and Adam Slagell. 2015. Preemptive intrusion detection: theoretical framework and real-world measurements. In *Proceedings of the 2015 Symposium and Bootcamp on the Science of Security* (HotSoS '15). ACM, New York, NY, USA, , Article 5 , 12 pages. DOI=10.1145/2746194.2746199 http://doi.acm.org/10.1145/2746194.2746199


The Honey Pot and
The Honey Badger