Did You Do Your Homework?

Use Case-Driven SIEM Deployments

SIEM SUMMIT 2019

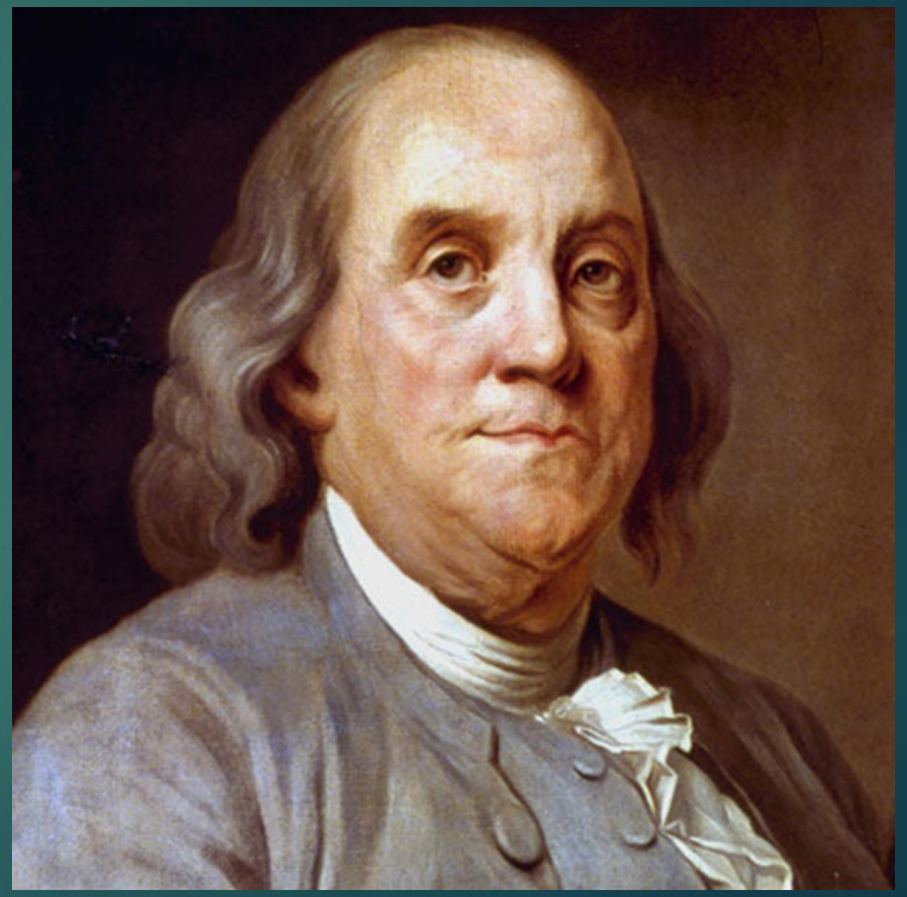
Who am I?

Scott Lynch @PacketEngineer

- Security Operations Manager @SSCSpace
- Former US Navy EW
- Up and coming SANS Instructor SEC555

Did you do your homework?

"If you **fail to plan**, you are **planning** to **fail**.", Benjamin Franklin or Spiderman….



Importance of Planning

- Identify the purpose
- Expected Outcome
- Timeline
- Resources
- Importance and Priority
- Impact of failing

Define base requirements

- Determine the need
  - What problems are you solving?
- Business case
  - Compliance, Customer requirement
- Storage and Log Volumes
- Events Per Second (EPS)
- Budget
- Time

Lessons learned as a guide

Use lessons learned from previous security events

Example: User infects machine with PowerShell malware from an email

Lesson Learned:

- A/V did not stop infection and execution
- Host logging was limited
- Malware succeeded in executing and infection

After Action:

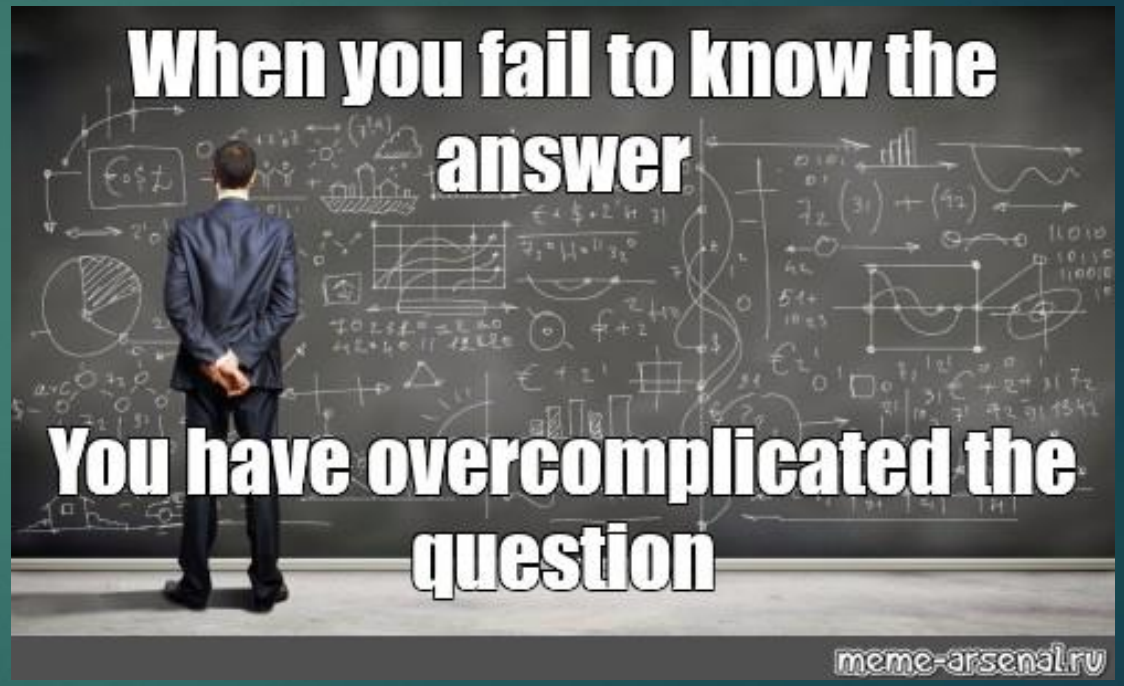
- Install Sysmon and collect additional log data
- Enable PowerShell Logging
- Update A/V policy to detect PowerShell execution
- Send logs to centralized logging solution for alerting and monitoring

Requirements we've all heard before

- Identify all the threats on my network
- Let me when things I don't want to happen, happen
- Let me know when a user clicks on a phishing email
- Let me know when a treat actor tries to break into my network

Its complicated…

- Log sources are not your only problem
- Standards vary, leading to customization or modification
- Enrichment may be needed to add value

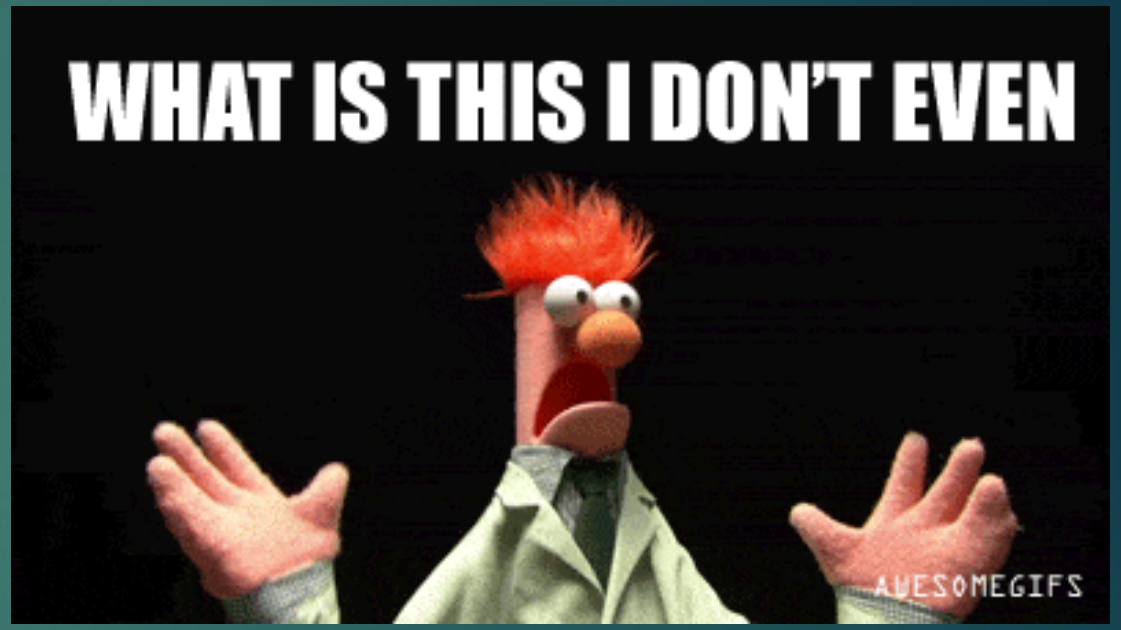


Examples of real requirements

- User interface and analyst experience
- Correlation
- Log source coverage
- Dashboards and analyst views
- Reporting
- Search and query
- Escalation, shift and analyst collaboration support
- Ability to gradually expand storage on demand
- Complete log categorization and normalization for cross-device correlation
- New log source integration technology and process: ability to either quickly integrate new log sources or have vendor do it promptly (days to few weeks) upon request

How does my network work?

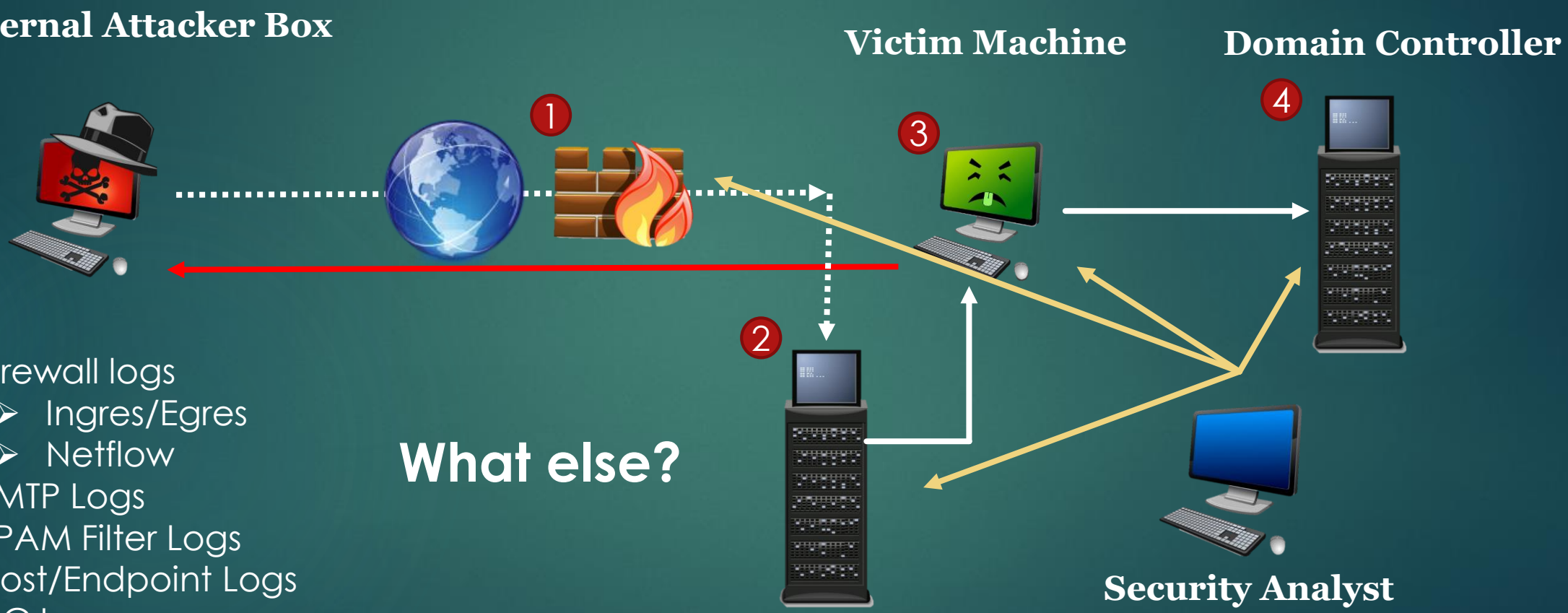
- Simple question with not so simple answer
- CIS 1 - Inventory of hardware
- CIS 2 - Inventory of software
- Network diagrams
- Entity relationship diagrams (ERD)



Use case : Unauthorized logins to internet facing asset

1. External attacker phishes victim machine
2. Attacker exploits system and attempts to pivot to DC
3. DC logs failed login attempt

Expanded Use Case



- Firewall logs
  - Ingres/Egres
  - Netflow
- SMTP Logs
- SPAM Filter Logs
- Host/Endpoint Logs
- DC Logs
- Observation

What else?

Write it down!

- Requirements
- Test plans
- Implementation plans
- Change control docs
- Training
- …

Custom SIEM Use Cases/Scenarios

Problem: User may have opened a malicious email attachment

Possible Detections:

- Event logs
  - What Event ID's?
- Sysmon
  - Additional log detail
- Network Traffic
  - Host or LAN/WAN, Netflow
- End Point Protection
  - A/V or EDR
- DNS Queries

Standardization - Sigma

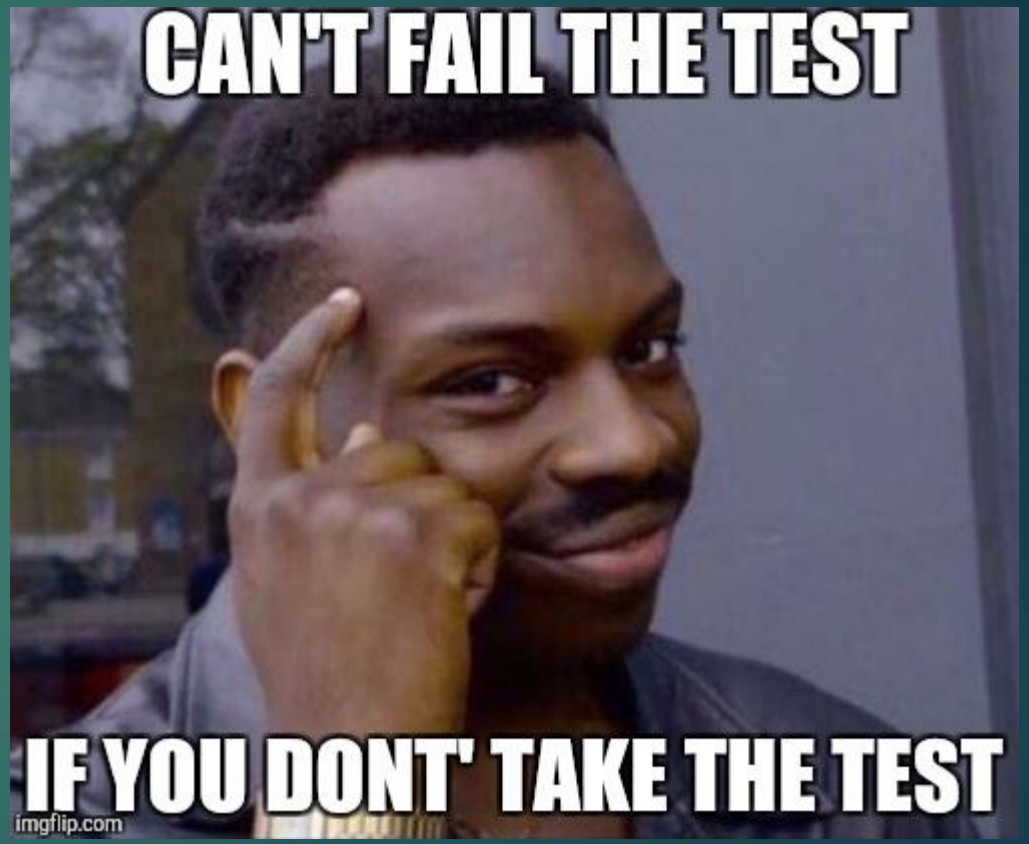
- Sigma is a generic and open signature format
- The rule format is
  - flexible
  - easy to write
  - applicable to any type of log fil.
- The main purpose to provide a structured form shareable with others.
- Sigma is for log files what Snort is for network traffic and YARA is for files.

Developed by
Florian Roth @cyb3rops and
Thomas Patzke @blubbfiction

https://github.com/Neo23x0/sigma

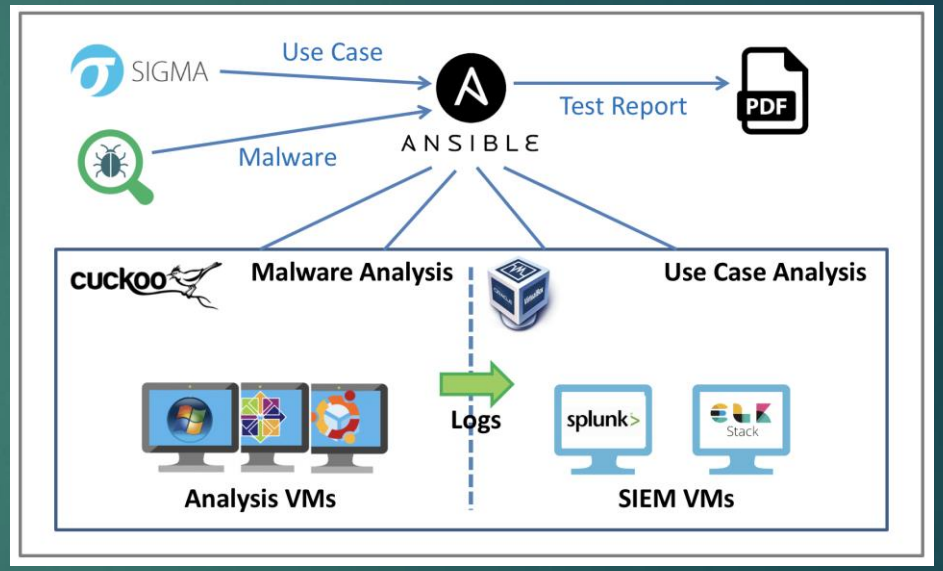
Testing

- Need to validate your use cases
- Test case to follow the use case
- Document your findings



Ypsilon: Automated Use Case Testing

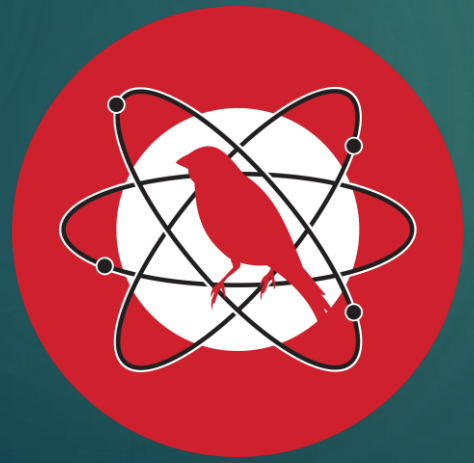
- Ypsilon is an Automated Security Use Case Testing Environment using real malware to test SIEM use cases in an closed environment.
- Different tools such as Ansible, Cuckoo, VirtualBox, Splunk and ELK are combined to determine the quality of a SIEM use case by testing any number of malware against a SIEM use case.
- Finally, a test report is generated giving insight to the quality of an use case.



https://github.com/P4T12ICK/ypsilon

Atomic Red Team

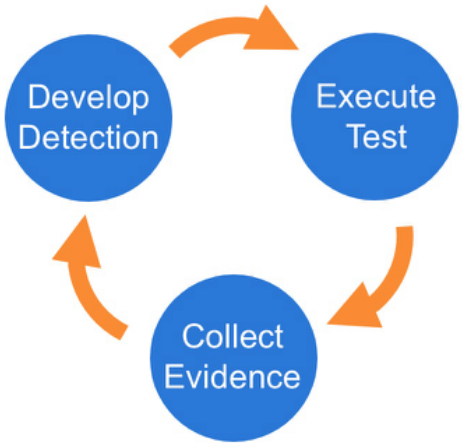
- Select a Test
- Execute a Test
- Collect Evidence
- Develop Detection
- Measure Progress



Getting Started Testing with Atomic Tests

We suggest a phased approach to running a test and evaluating your results:

1. Select a test
2. Execute Test
3. Collect Evidence
4. Develop Detection
5. Measure Progress



Best Practices

- Be sure to get permission and necessary approval before conducting tests. Unauthorized testing is a bad decision and can potentially be a resume-generating event.
- Set up a test machine that would be similar to the build in your environment. Be sure you have your collection/EDR solution in place, and that the endpoint is checking in and active.
- Spend some time developing a test plan or scenario. This can take many forms. An example test plan could be to execute all the Discovery phase items at once in a batch file, or run each phase one by one, validating coverage as you go.

https://github.com/redcanaryco/atomic-red-team

Caldera and Brawl

- Automated adversary emulation system
- BRAWL seeks to create a compromise by creating a system to automatically create an enterprise network inside a cloud environment

- https://github.com/mitre/caldera

https://github.com/mitre/brawl-public-game-001

Additional resources

- Blue Team handbook: SOC, SIEM and Threat Hunting Use Cases. Notes from the the Field (v1.02), Don Murdoch
- Security Information/Event Management Security Development Life Cycle v5 https://www.sans.org/media/score/esa-current.pdf
- (NIST) Special Publication (SP) 800-64, Security Considerations in the System Development Life Cycle https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-64r2.pdf
- SPECIFYING SYSTEM SECURITY REQUIREMENTS, Paula A. Moore https://csrc.nist.gov/csrc/media/publications/conference-paper/1999/10/21/proceedings-of-the-22nd-nissc-1999/documents/papers/t03.pdf
- **Red Teaming/Adversary Simulation Toolkit** https://github.com/infosecn1nja/Red-Teaming-Toolkit