FIRSTCON25

Detection Engineering 101

Establishing a Structured Approach to Detection Engineering

2025.6.26

Tomohisa Ishikawa, Ph.D., CISSP, CSSLP, CCSP, CISA, CISM, PMP
scientia.admin@gmail.com
@scientia_sec

packt

1ST EDITION

Practical Threat Detection Engineering

A hands-on guide to planning, developing, and validating detection capabilities

MEGAN RODDIE
JASON DEYALSINGH | GARY J. KATZ

Tomo

$ whoami

Name :	Tomo (Tomohisa Ishikawa)
Affiliation:	Distinguished Cyber Security Architect @ Tokio Marine HD
Certifications:	Ph·D·, CISSP, CSSLP, CCSP, CISA, CISM, PMP etc·
Expertise Area:	Ex-Red Teamer and Current Blue Teamer • Security Strategy, Security Architecture, Cyber Threat Intelligence, Security Operation…
External Activity:	• Member of National IT Exam Committee • Cybersecurity Experts in Gov Agency (part-time) • Speaker @ DEFCON24 SE Village, LASCON 2016, BSide Philadelphia 2016, FIRSTCON23, GCC 2024-2025… • Author of book related to Cyber Thereat Intelligence • Translate 7 cybersecurity books into Japanese

Today's Goal

Today's Goal :

Sharing structured approach of Detection Engineering &

Understand how to operationalize Detection Engineering Process

Agenda :

- **Part I: Defining Detection Engineering**
- **Part II: Detection Engineering Process Deep Dive**
- **Part III: "Successful" Detection Engineering Program**

Part I : Defining Detection Engineering

What is Detection Engineering？

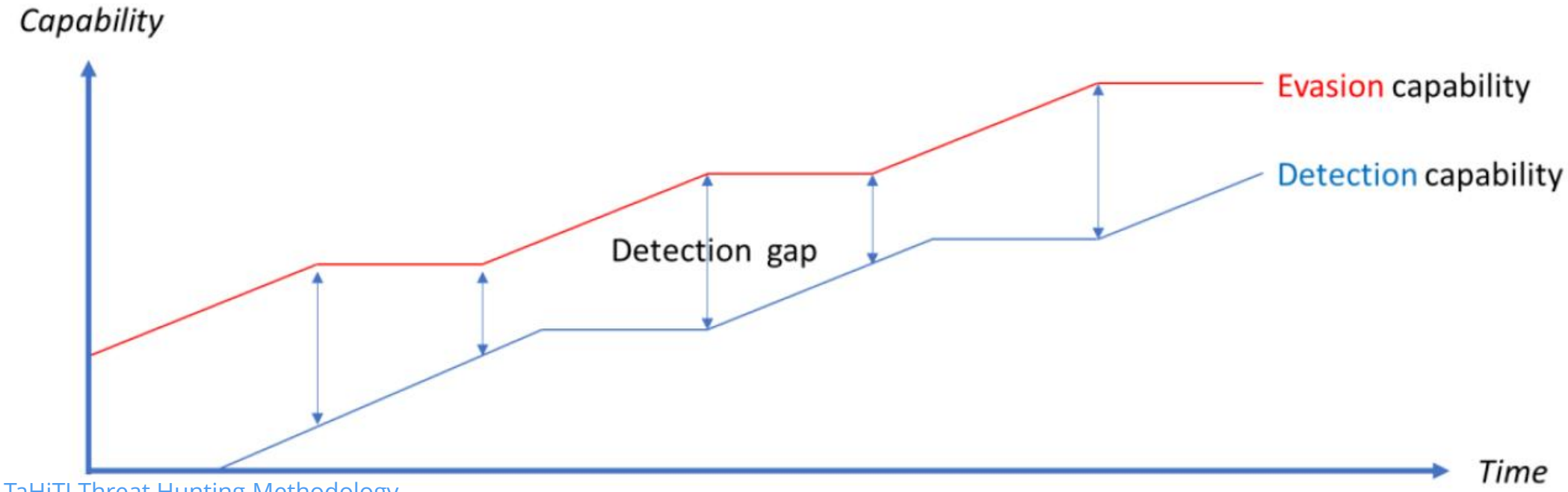
- Many definition is available, but definition from SentinelOne says:
  - *“Detection engineering is a structured approach to developing, optimizing, and managing rules, alarms, and processes to detect threats or suspicious activity in real-time”*

Ref : https://www.sentinelone.com/cybersecurity-101/threat-intelligence/detection-engineering

What is Detection Engineering?

- **Propose Another Definition of "Detection Engineering"**
  - *"Systematic approach to mind Detection Gap"*
- **Detection Gap :**
  - "Cat & Mouse Game" btw "Threat Actor" and "Blue Team"

< What is Detection Gap?>



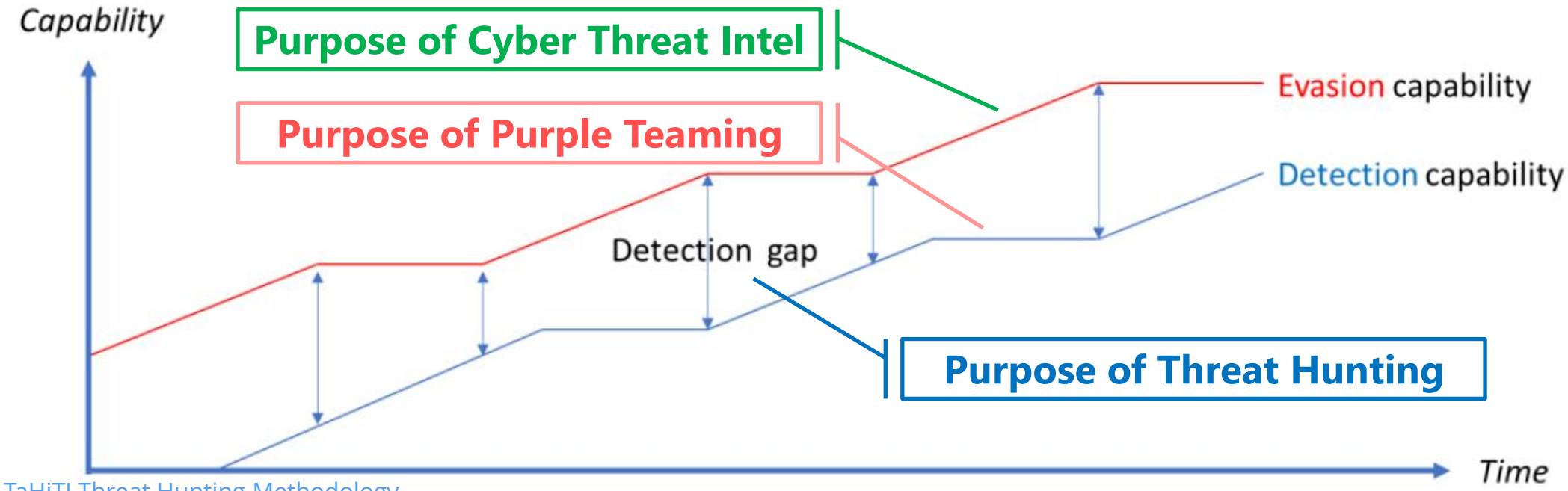
Ref : TaHiTI Threat Hunting Methodology

What is Detection Engineering?

- **Three way to "Identify" Detection Gap**
  - **Cyber Threat Intel** : Identify latest TTPs
  - **Purple Teaming** : Identify existing Prevention & Detection capability
  - **Threat Hunting** : Identify unknown threat evading existing security mechanisms

- **Detection Engineering is to "Mind" Detection Gap**

< What is Detection Gap?>



Ref : TaHiTI Threat Hunting Methodology

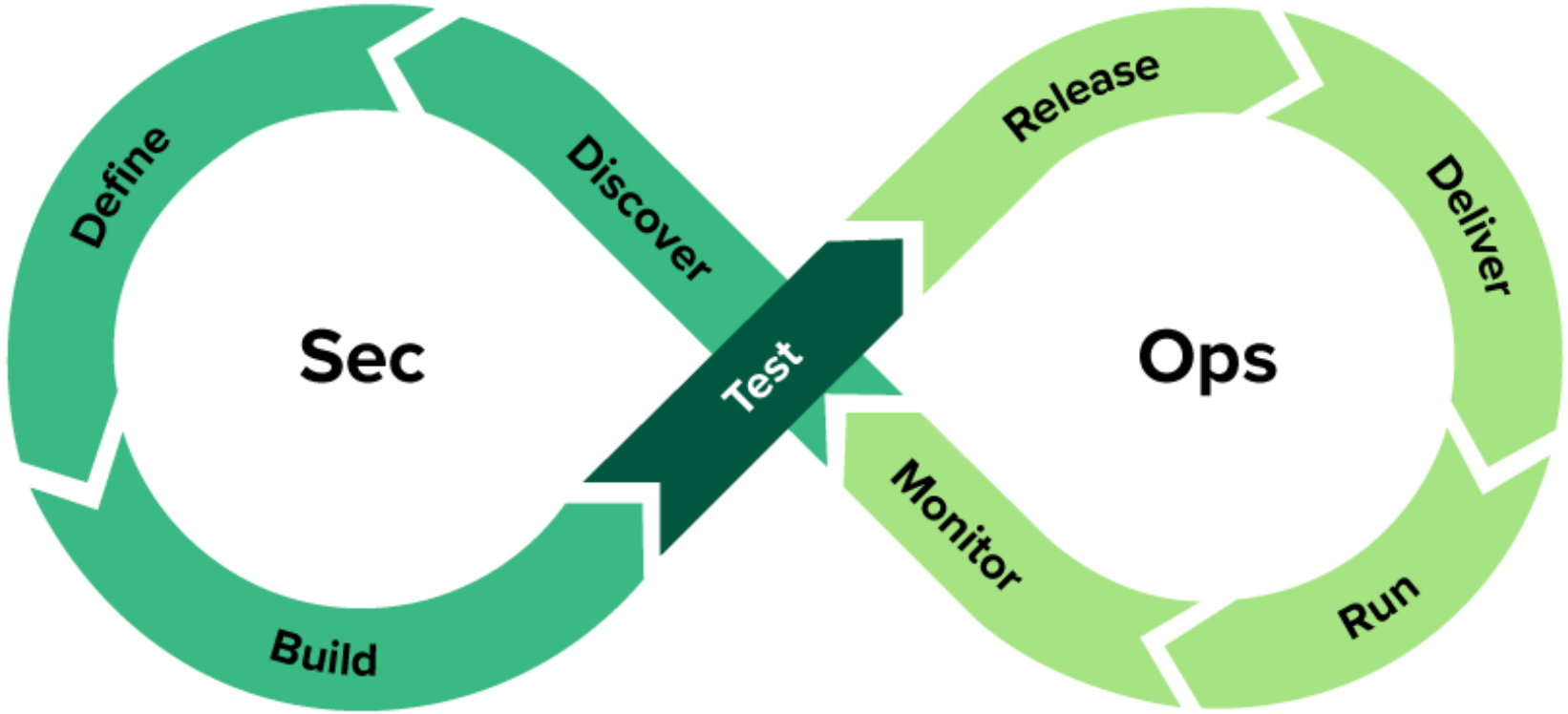
What is Detection Engineering?

- **Three Characteristics of "Detection Engineering"**
  - Proposed by Forrester Principal Analyst (Allie Mellen)
  - ***#1 : Detection as Code***
    - Use and Manage detection rule as code (YARA, SIGMA etc.)
  - ***#2 : Applying Software Engineering***
    - *Structured Approach = Software Engineering*
    - *"The application of a systematic, disciplined, quantifiable approach to the development, operation and maintenance of software" (definition of Software Engineering by IEEE)*
    - Use metrics for evaluating and managing detection engineering
  - ***#3 : "Agile" Approach***
    - Start Small, and Continuous Improvement

Ref : https://www.forrester.com/blogs/announcing-the-detection-and-response-development-lifecycle-dr-dlc-for-detection-engineering/

DR-DLC : Detection & Response Development Life Cycle

- Originally proposed by Forrester Principal Analyst (Allie Mellen)



https://www.forrester.com/blogs/announcing-the-detection-and-response-development-lifecycle-dr-dlc-for-detection-engineering/

Part II : Detection Engineering Process Deep Dive

DR-DLC : Detection & Response Development Life Cycle

< Overview of Each Phase>

Step	Description
Discover	Identify detection rules that need to be created or improved.
Define	Define the requirements for detection rules.
Build	Create detection rules that meet the defined requirements.
Test	Test whether the detection rules function as expected.
Release	Review and approve the created detection rules.
Deliver	Document, manage, and store the detection rules in the appropriate repository
Run	Deploy and execute the detection rules in the production environment.
Monitor	Manage the accuracy and quality of detections based on KPIs and functional metrics.

Detection Engineering Process

Discover | Define | Build | Test | Release | Deliver | Run | Monitor

Phase 1 – Discover

- Identify detection rules that need to be created or improved
- 4 Major Request Path

< 4 Major Request Path in Discover Phase >

Request from…	Request Overview
CTI Analyst	• By using CTI input for encountering new threat
SOC Analyst	• SOC analyst is working on the cyber attack frontier, and their insight will be helpful • SOC analyst also request DE based on the results of **purple teaming** and **threat hunting**
Security Policy	• Business and security policy require additional control for securing the environments • Ex) prohibiting specific services
"Monitoring phase"	• "Monitoring" phase is evaluation phase of detection rule quality

Detection Engineering Process

Discover | **Define** | Build | Test | Release | Deliver | Run | Monitor

Phase 2 – Define

- Based on “Discover” phase, we will define/describe the requirement for detection rules
- Two activities is required.
  - *Step 2-1 : Detection Requirement Definition*
  - *Step 2-2 : Triage (prioritization)*

Detection Engineering Process

Discover | **Define** | Build | Test | Release | Deliver | Run | Monitor

Phase 2 – Define

- **2-1 : Detection Requirement Definition**
  - Define the requirement for creating “detection rule”

< RESCUE framework for Detection Requirement Definitions >

Components	Details
Requester	Person to request to create detection rule (Need to satisfy requester’s expectation)
Evidence	Provide CTI and log data that form the basis for creating detection rules
Scope	Define the scope for detection rules (target scope and time period to be applied)
Contents	Define the technical details of detection rule
Utility / Urgency	The reason that this specific detection rule needs to be created (used for deciding priority)
Exception	Describe exceptions of detection rule for false positives prevention

Detection Engineering Process

Discover | **Define** | Build | Test | Release | Deliver | Run | Monitor

Phase 2 – Define

- **2-2 : Triage (Prioritization)**
  - Based on defined “requirement”, we decide the priority
  - Triage criteria in addition to “Utility / Urgency” of RESCUE framework is as follows.

< Triage Criteria >

Viewpoint	Details
Severity	Does this requirement bring a significant impact if detection rules are not created?
Consistency	Does the detection requirement align organization profile with the attack targets, intent, and capabilities of each threat actors?
Coverage	Does the requirement enhance the organization's detection coverage based on identified "detection gaps" from purple teaming or threat hunting?

Detection Engineering Process

Phase 3 – Build

- After “Define” phase, we will build “detection code” that meet the defined requirements.
- Two activities is required.
  - *Step 3-1 : Design*
  - *Step 3-2 : Development*

Detection Engineering Process

Discover | Define | **Build** | Test | Release | Deliver | Run | Monitor

Phase 3 – Build

- **3-1 : Design**
  - ***(A) Research***
    - Understanding necessary components to create “detection code”
      - *Source* : What kind of log source are necessary?
      - *Logic* : What kind of detection logic are required?
      - *Conditions* : What kind of conditions are required to avoid “false positive”?
  - ***(B) Describe “Technical Specification”***
    - Use **H.O.P.E. framework** for concise “Technical Specification”
  - ***(C) Create Validation Criteria***
    - Prepare test data and validation criteria for “Test” phase

Detection Engineering Process

Discover | Define | **Build** | Test | Release | Deliver | Run | Monitor

Phase 3 – Build

- **3-1 : Design**
  - **H.O.P.E. framework** can help to build “Technical Specification”

< H.O.P.E framework for Technical Specification>

Components	Examples
Hypothesis	”Threat actor create suspicious Domain Account”
Object of Investigation	Windows Event Log in Domain Controller Server
Procedure	Filter by Event ID (4720) and search log entries of account creation when helpdesk is closed
Evaluation Criteria	If expected entry is found, code generate alerts

Detection Engineering Process

Discover | Define | **Build** | Test | Release | Deliver | Run | Monitor

Phase 3 – Build

- **3-2 : Development**
  - Based on “design”, we will develop actual “detection rule”.
    - SIGMA
    - YARA

Detection Engineering Process

Phase 4 – Test

- Based on Detection Code created in “Build” phase, we test whether the detection rules function works as expected.

- **The purpose of “Test” phase**
  - To ensure that the detection code is implemented appropriately by aligning with the requirements definition and returns the expected behavior and results

- **Use Two Types of Test Data**
  - Known Good : Confirming no false positive
  - Known Bad : Confirming no false negative

Detection Engineering Process

Discover | Define | Build | Test | **Release** | **Deliver** | Run | Monitor

Phase 5 – Release

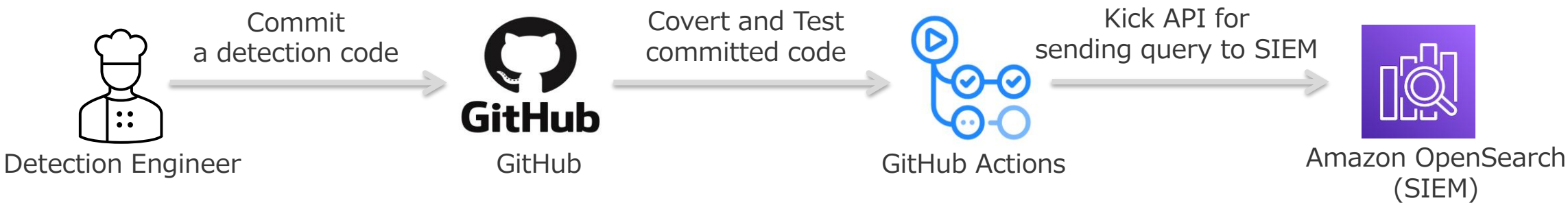
- Review and approve the created detection rules.

Phase 6 – Deliver

- Document, manage, and store the detection rules in the appropriate repository

For release and deploy management, Detection Engineering best practice typically recommends to use CI/CD pipeline.

< Sample workflow of CI/CD pipeline for Detection Engineering>



Detection Engineering Process

Discover | Define | Build | Test | Release | Deliver | **Run** | Monitor

Phase 7 – Run

- Deploy and execute the detection rules in the production environment.
- Use “Deployment Tag” on sharing confidence level of detection rule to manage the commitment level of SOC analyst.

< Deployment Tag >

Viewpoint	Details
Experimental	Detection rules that have not undergone extensive testing in a live environment. It is desirable to limit the scope of application and restrict alert reviews to only the rule creator, excluding other SOC analysts.
Testing	Detection rules that have progressed from the "Experiment" phase and are now available for broader use. From the perspective of detection quality, all SOC analysts review alerts at a lower priority compared to "Stable" rules.
Stable	Rules that have been deemed stable and are fully deployed in the production environment. SOC analysts review these rules with high priority. If false positives or improvements are needed after deployment, consultation with a Detection Engineer is required.

Detection Engineering Process

Discover | Define | Build | Test | Release | Deliver | Run | **Monitor**

Phase 8 – Monitor

- Manage the accuracy and quality of detections based on KPIs and functional metrics.
- Use 3M+C framework originally proposed by "*Practical Threat Detection Engineering*"

< 3M +C Framework for "Monitor" phase>

Metrics	• Evaluate "detection rules" performance from statistical KPI
Monitoring	• Continuously analyze actual "alerts" from detection rule to avoid false positive
Maintenance	• Update "detection rules" based on updated threat intelligence and additional insight from security operation including threat hunting
Continuous Validation	• Conduct purple teaming continuously to identify "detection gap"

Detection Engineering Process

Discover | Define | Build | Test | Release | Deliver | Run | **Monitor**

Phase 8 – Monitor

- Deep Dive : "**Metrics**"
  - Evaluate "detection rules" performance from statistical KPI (**Top-Down approach**)

< Example of "Metrics" alerts >

Metrics	Meaning
Number of Detection Alerts	The number of detection alerts that need to be reviewed by analysts.
Analysis Results	The number of cases where analysts reviewed alerts and determined whether they were false positives.
Average Time to Close	The average time required to analyze a alert and close it.
Standard Deviation of Time to Close	Time deviation to analyze and close alerts.
Change in Time to Close	The difference in the average time to close before and after updating detection rules.

Detection Engineering Process

Discover | Define | Build | Test | Release | Deliver | Run | Monitor

Phase 8 – Monitor

- Deep Dive : “**Monitoring**”
  - Continuously analyze actual “alerts” from detection rule (**Bottom-Up approach**)
  - Even we have various validation in test phase, we might have “false positive”

  - Ex) “nc” command detection => “rsync” or “sync”
  - Ex) “Emotet” detection of Excel Macro execute “powershell.exe”

Detection Engineering Process

Discover | Define | Build | Test | Release | Deliver | Run | **Monitor**

Phase 8 – Monitor

- Deep Dive : "**Maintenance**"
  - "Agile" Approach - *"Start Small, and Continuous Improvement"*
  - Various Approach are available:
    - Use External Threat Intelligence
    - Analyze Similar Malware (i.e. VirusTotal Commercial License)
    - Utilize Deception Techniques for Detections

Detection Engineering Process

Discover > Define > Build > Test > Release > Deliver > Run > **Monitor**

Phase 8 – Monitor

- Deep Dive : "**Continuous Validation**"
  - Same as **Purple Teaming** as below, but apply "atomic" purple teaming concept

Information Gathering > Threat Scenario > Evaluation > Analysis & Report

< VECTOR framework for Making Scenarios>

Components	Details
Validation Scope	Define the direction of validation (which TTPs/Techniques?)
Exception	Describe exception for this validations
Control Scope	Define the scope of detection mechanism, technical control, and logs
Testing Procedure	Define step-by step procedure of testing
Outcome Expected	Define expected results by proceeding "testing procedure"
Review Criteria	Define the review criteria whether or not existing "detection code" correctly works.

Part III : ”Successful” Detection Engineering Program

What is the “Successful” Detection Engineering Program

– 3 critical KPI will define what is “successful” in Detection Engineering Program

< 3 Critical KPI for “successful” detection engineering program>

Metrics	Viewpoint	Overview
MTTD + MTTR	*Time*	*Explain the resilience capability from time-basis*
Precision & Recall	*Efficiency*	*Explain the “false positive” ratio and “false negative” ratio as the efficiency of detection*
Detection Coverage	*Coverage*	*Covering entire MITRE ATT&CK*

”Successful” Detection Engineering Program

KPI #1 : MTTD + MTTR

- Detection Engineering shorten MTTD + MTTR
  - MTTD（Mean Time To Detect）　　: Average time between attack start and have detected
  - MTTR（Mean Time To Response）　　: Average time between detection and response

- **Background Theory : Time-Based Security (by Winn Schwartau)**
  - System are secured if it satisfy following equation
  - **MTTA > MTTD + MTTR**
    - MTTA（Mean Time To Attack）　　: Average time between attack start and end

KPI #2 : Precision + Recall

- Precision + Recall is defined as follows.

< Confusion Matrix >

		Alert Generation?	
		YES	**NO**
Malicious Attack?	**YES**	*True Positive*	*False Negative*
	NO	*False Positive*	*True Negative*

Recall (False Negative)

Precision (False Positive)

< Precision + Recall>

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

Understanding False Positive ratio

An indicator of how many malicious attacks are correctly detected out of the total # of alerts generation

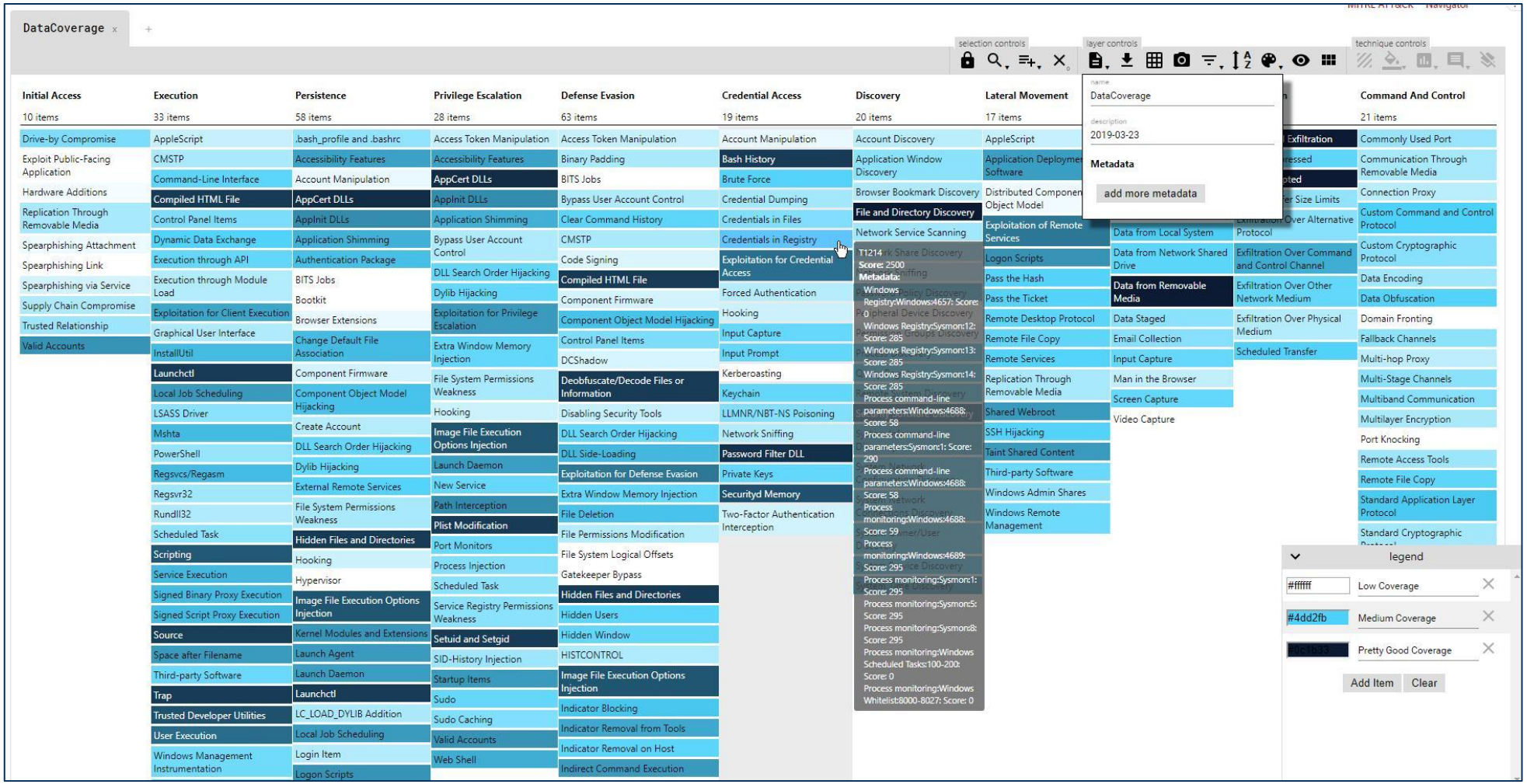
$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

Understanding False Negative ratio

An indicator of how many malicious attacks are correctly detected out of the total # of attacks that actually occur.

KPI #3 : Detection Coverage

- Detection Coverage is as follows.



Source : https://twitter.com/olafhartong/status/1109569799863091201

”Successful” Detection Engineering Program

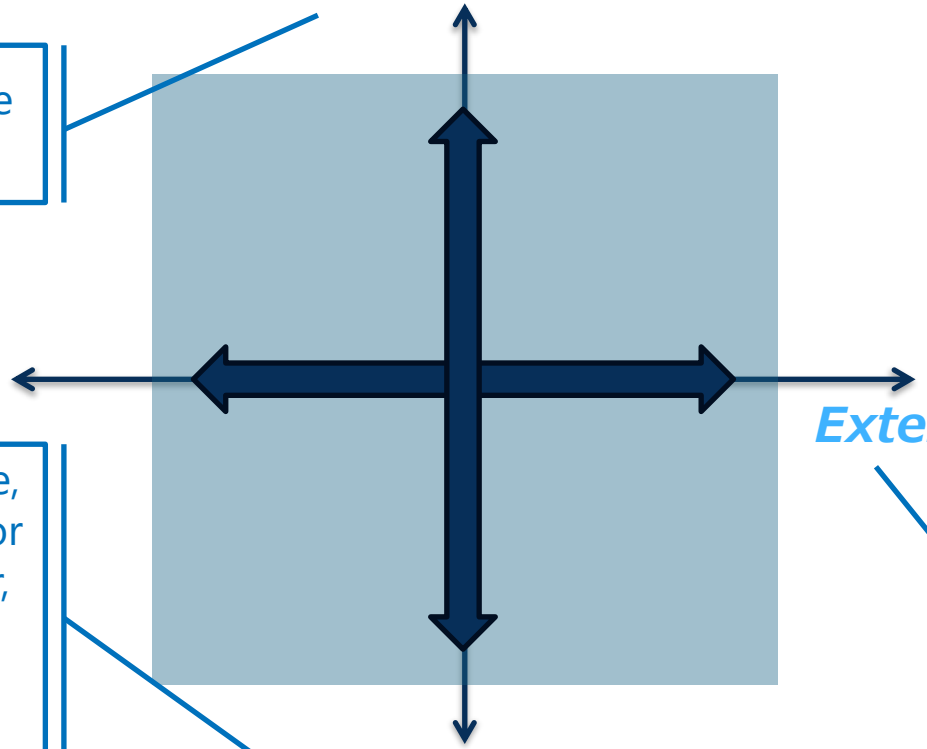
- “3M + C framework“ can improve “3 Critical KPIs”.
- “3M + C framework” makes Detection Engineering Program “successful”.

< 3M + C framework & 3 Critical KPIs>

	Metrics	Monitoring	Maintenance	Continuous Validation
Evaluation Viewpoint	Top Down	Bottom-Up	Bottom-Up	Top Down
Approach	Reactive	Reactive	Proactive	Proactive
Evidence	Quantitative	Qualitative	Qualitative	Quantitative
↑ MTTD + MTTR	✅	-	-	✅
↑ Precision + Recall	Precision	Precision	Recall	Recall
↑ Detection Coverage	-	-	✅	✅

How to Expand the Detection Capability

- Extension : Extend the detection capability to similar attack methods or tools
- Sophistication : Deep-dive into technical details for further detections
- Abstractions : Generalize detections (Follow the Idea of MITRE “***Summiting The Pyramid***”)



By generalizing detections, we increase the detection coverage.

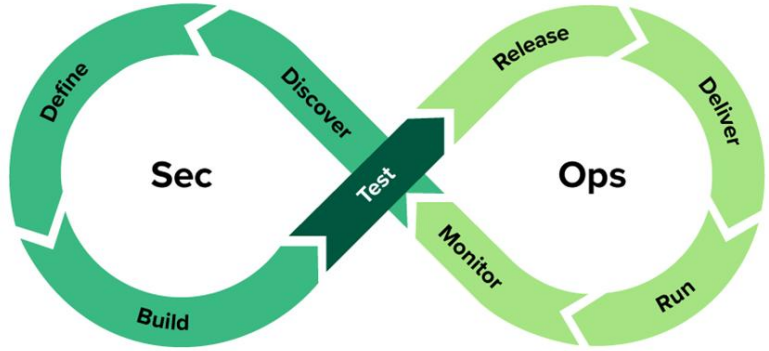
For malware detections, in initial phase, we will use hash or extracted strings for detections in “static analysis”. However, after conducting code analysis, we will have a technical details such as binary code.

For Kerberoasting attack, this extends the tools to detect NOT ONLY mimikaz BUT ALSO Rubeus and Invoke-Kerberoast.ps1.

Wrap-Up

Wrap-Up

- ***Part I: Defining Detection Engineering***
  - "Systematic approach to mind Detection Gap"
  - Three Characteristics of Detection Engineering
  - Detection Engineering Process (DR-DLC Model)



- ***Part II: Detection Engineering Process Deep Dive***
  - Detailed Explanations of DR-DLC Process
  - Sharin several framework/idea for structured approach such as RESCUE, HOPE, 3M+C, VECTOR

- ***Part III: "Successful" Detection Engineering Program***
  - High-Level KPIs for "Successful" Detection Engineering Program
  - Discuss the relationship between 3 Metrics and 3M+C "Monitor" framework
  - 3 Strategies for further improvement

Thank You!


@scientia_sec


https://www.linkedin.com/in/tomohisaishikawa/


scientia.admin@gmail.com