Endpoint Detection and Response (EDR) Ontology Considerations -= Track 2

Update 3/9/2022

Public Domain Dedication

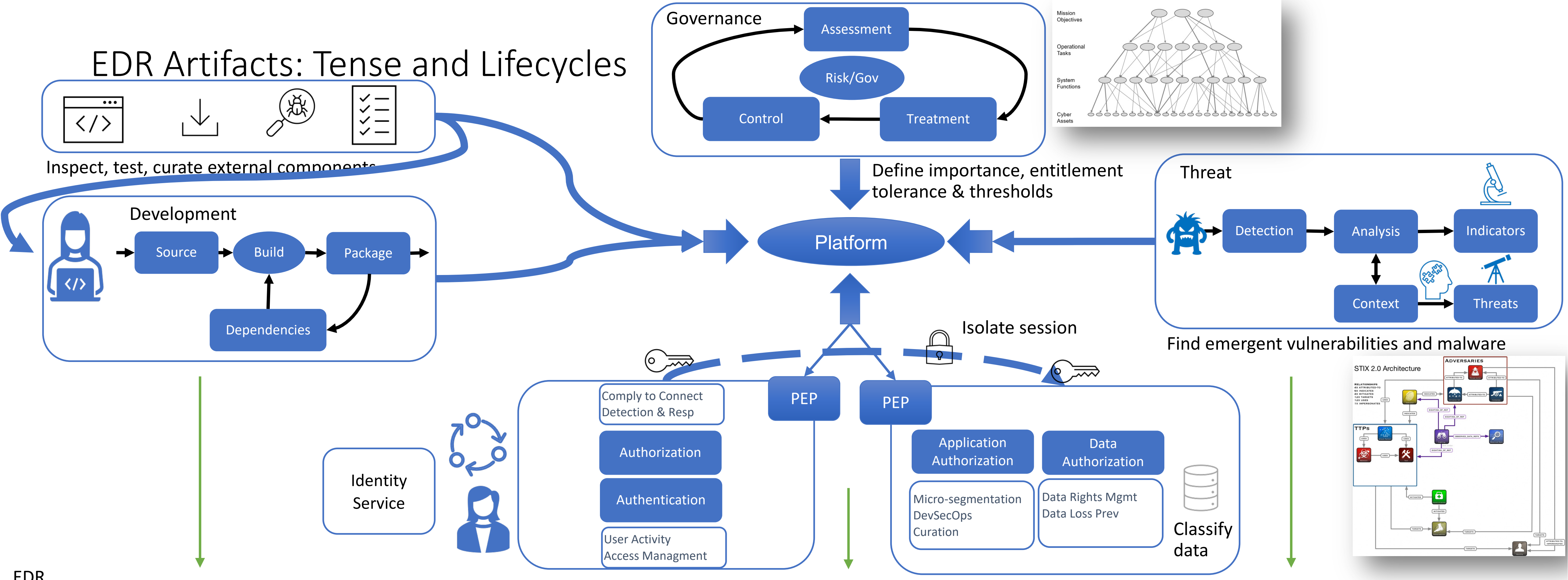
Per Last Meeting

- Parking interactions with external groups until we are ready to form a project/sub-project.
- Proceeding with analysis for normalization – cataloging the “gotchas” across *DR tooling (tooling capable of supporting *DR (EDR, NDR, XDR, …) required functionality (emergent threat detection & response, intelligence ingestion, mal/anomalous detection, hunting, analysis, response…) :
  - Comodo – widely used
  - GRR - cloud scale
  - BlueSpawn – academic
  - Question 1: Others?
  - Question 2: How to include non-OSS products (proprietary integrations (commercial API), normalized mappings (ATT&CK), or only at the “indicator sharing” level (STIX)…)

Track 2 - Objective 1 - Status

- Indication and Behavior Normalization
- Indications and Behaviors are potentially invariant attributes across *DR tooling, so are important in normalizing across different *DR implementations.
- As Indications and Behaviors get exposed, across the security, dev and system disciplines, some implicit context around indications and behavior, need to be made more explicit.

EDR Artifacts: Tense and Lifecycles



Inspect, test, curate external components

Development

Source → Build → Package

Dependencies

Governance

Assessment

Risk/Gov

Control

Treatment

Mission Objectives

Operational Tasks

System Functions

Cyber Assets

Define importance, entitlement tolerance & thresholds

Platform

Threat

Detection → Analysis → Indicators

Context → Threats

Find emergent vulnerabilities and malware

STIX 2.0 Architecture

ADVERSARIES

TTPs

Isolate session

PEP

PEP

Comply to Connect Detection & Resp

Authorization

Authentication

User Activity Access Managment

Identity Service

Application Authorization

Data Authorization

Micro-segmentation DevSecOps Curation

Data Rights Mgmt Data Loss Prev

Classify data

EDR Artifacts

Dev Context & Telemetry

- IDs: SW Component ID: SWID, CycloneDX, SPDX, GUID, "string"…
- **Intended** SW Component Configuration: Settings, Privilege, Dependencies, secrets, obj hashes, policy …
- **Expected** SW Component Configuration (Test): DLLs Used,
- **Intended** SW Component Behavior: OpenAPI, RAML, … (L7)
- **Expected** SW Component Behavior (Test): SysCall/ Res Profile, Memory, Network, Data, CPU

Operational Telemetry & Context

- IDs: Instance MAC, IP, SysSID, GUIDs, … (stack)
- Provisioning Decisions (operationalization)
- Provisioned Configuration (deployment baseline)
- Mapping from SW Component Manifest -> Instance:
- **Observed** SW Component Configuration (Instances):
- **Observed** SW Component Behavior (Instances):
- **Observed** Telemetry Inconsistency:

Curated Intelligence

- IDs: Relevant Component ID Types specified, Malicious IDs identified, …
  - "strings", SWID, SPDX, CycloneDX, (SCAP: CPE, CVE, …)
- **Relevant** Indicators (Instances):
- **Relevant** Behaviors (Instances):
- **Relevant** Inconsistencies (anomaly): (telemetry)
- Mitigation **Verification**: State & behavior restriction
- Remediation **Verification**: Sustained resilience to repeat exploit
- Cleanup considerations

Ontology Consideration 1: Tense

- “Tense” is conventionally implicit in EDR ecosystem, and inconsistently selected & represented across EDR tools/svcs.
- EDR relevant artifacts are produced by different processes, at different times with different implications
    - Intended – By design or decision. May be coverage tested. Ex. Supported API
    - Expected – Observed under test. Cannot be coverage tested. Good automate-able baseline. Test Platform sensitive. Ex. SysCall pattern demonstrated during coverage testing.
    - Observed – Runtime or forensic telemetry. Relevant by difference from Intended, Expected or by association with Vuln* or Mal* via Intelligence feed.
    - Verification – What should I see post mitigation/remediation?
- Maybe be the same attribute across all tenses: Intended, Expected, Relevant, Verification – Registry setting, DLL Hash
- But, may be different. May only be indirectly associated: Patch level vs. Patch Level Indication
- Recommendation: make provision to capture Intended, Expected, Observed, Verification, … on Configuration and Behavior
- * Process , Ontologies (upper vs lower) - Patrick