Raytheon

Leaning Into Large Ground System Vulnerabilities with Machine Learning

Raytheon Intelligence, Information and Services

David A Wilson, Addy Moran, Joshua Welch

February 26, 2019

Current Situation

Raytheon
Intelligence, Information and Services

Modern ground systems have complex combinations of COTS/FOSS products

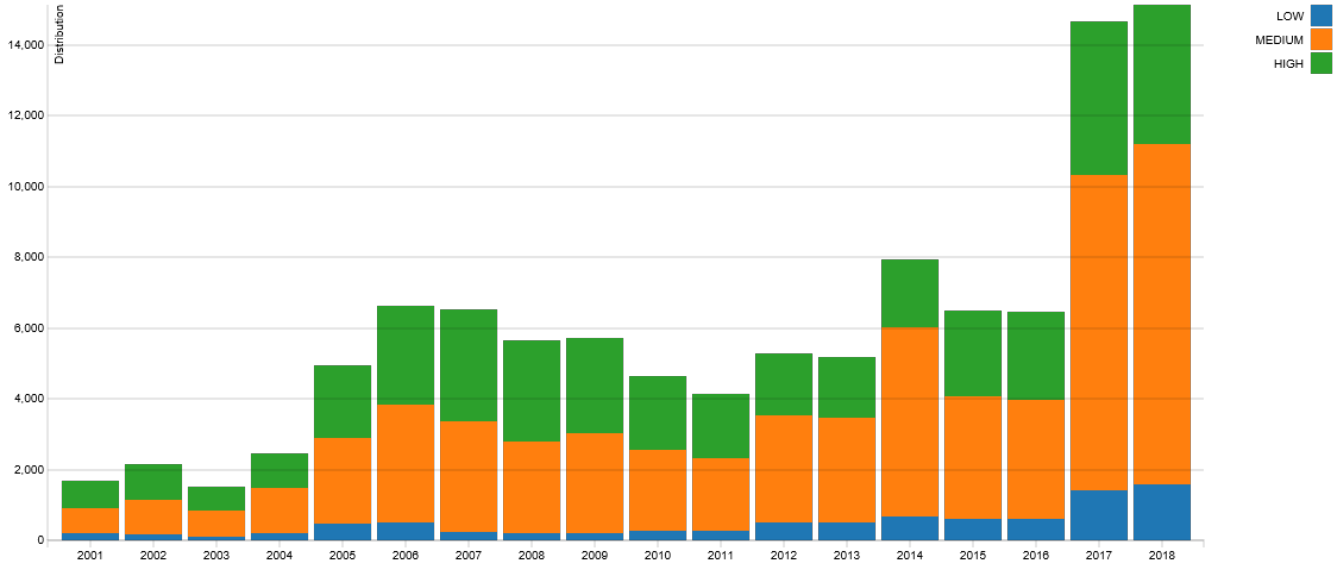
Vulnerability updates can become an unmanageable amount of work for system administrators.

Potential for:

- Alert fatigue
- Updates unsuccessfully processed

4,800+ open source vulnerabilities were reported in 2017[1]

15,130 Security Vulnerabilities published in 2018 (**3,944** High Priority)[2]



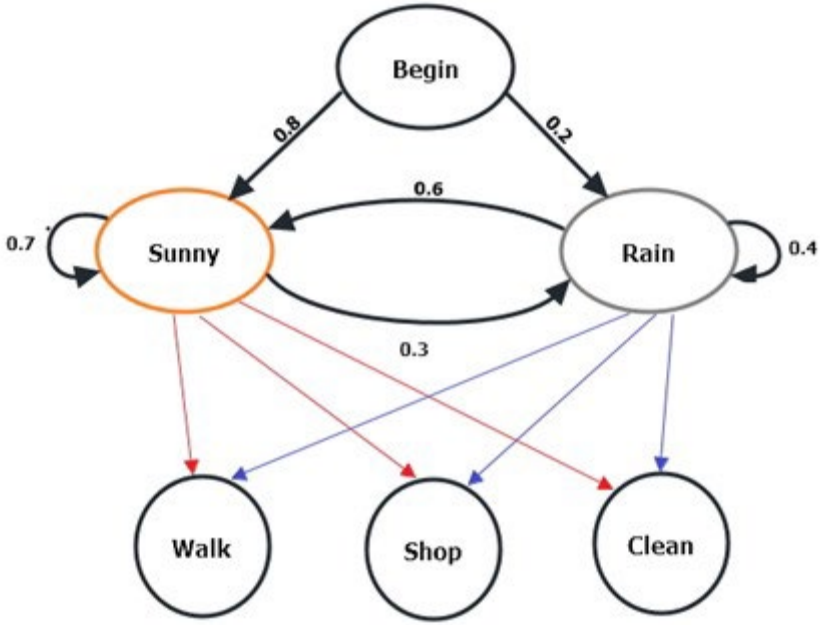
Low/Medium/High Security Vulnerabilities Published in 2018

(1) 2018 Open Source Security and Risk Analysis, Synopsys Center for Open Source Research & Innovation
(2) As of 12/31/2018, https://nvd.nist.gov/general/visualizations/vulnerability-visualizations/cvss-severity-distribution-over-time

Approach

Create a risk model that provides risk posture of sum of COTS/FOSS vulnerabilities and provide suggested patches to improve risk

Use Markov chains, Human-Interactive Machine learning, and data-mining to prioritize system patches



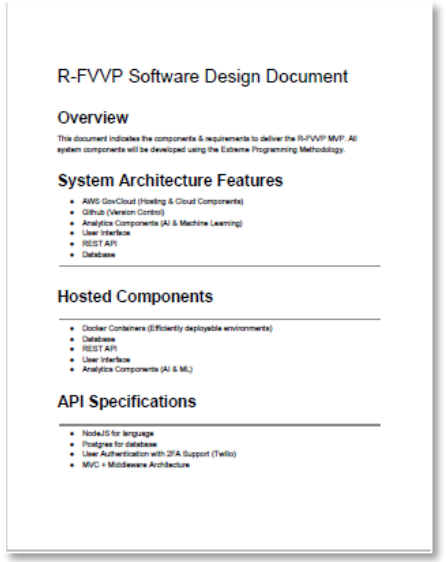
Example Markov Chain Model

Administrator can easily improve overall system risk posture by applying suggested prioritized patches

Approach

Raytheon
Intelligence, Information and Services

- Use a Markov chain model to simulate the movements of a hacker inside a system.
- Quantitative security data such as Operating System (OS) scans, network scans, and network topologies to classify the severity of each vulnerability.
- Takes into account the number of connections per each component and classifies a weight per vulnerability, and uses this to rank patches.
- This vulnerability rank is used as a prioritization scheme.
- The algorithm uses vulnerability data and integrates the data into the network topology and builds an absorbing Markov model to predict which systems an attacker is most likely to attack (and therefore patching priorities).


R-FVVP Software Design Document

Overview

This document indicates the components & requirements to deliver the R-FVVP MVP. All system components will be developed using the Extreme Programming Methodology.

System Architecture Features

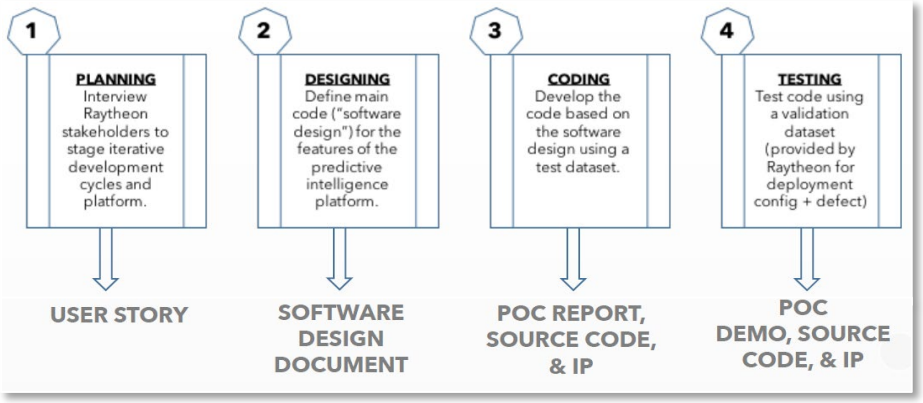
- AWS GovCloud (Hosting & Cloud Components)
- Github (Version Control)
- Analytics Components (AI & Machine Learning)
- User Interface
- REST API
- Database

Hosted Components

- Docker Containers (Efficiently deployable environments)
- Database
- REST API
- User Interface
- Analytics Components (AI & ML)

API Specifications

- NodeJS for language
- Postgres for database
- User Authentication with 2FA Support (Twilio)
- MVC + Middleware Architecture



Vulnerability Data

- The Common Vulnerability Scoring System 3.0 (CVSS) provides an open framework for communicating the characteristics and impacts of IT vulnerabilities.
- CVSS 3.0 metrics for risk assessment

Exploitability Metrics: Reflect the characteristics of the thing that is vulnerable. It has Attack Vector (AV), Attack Complexity (AC), Privileges Required (PR) and User Interaction (UI).

Scope (S): Scope refers to the collection of privileges defined by a computing authority when granting access to computing resources. When the vulnerability of a software component governed by one authorization scope is able to affect resources governed by another authorization scope, a Scope change has occurred.

Impact Metrics: Refer to the properties of the impacted component. It has Confidentiality Impact (C), Integrity Impact (I) and Availability Impact (A).

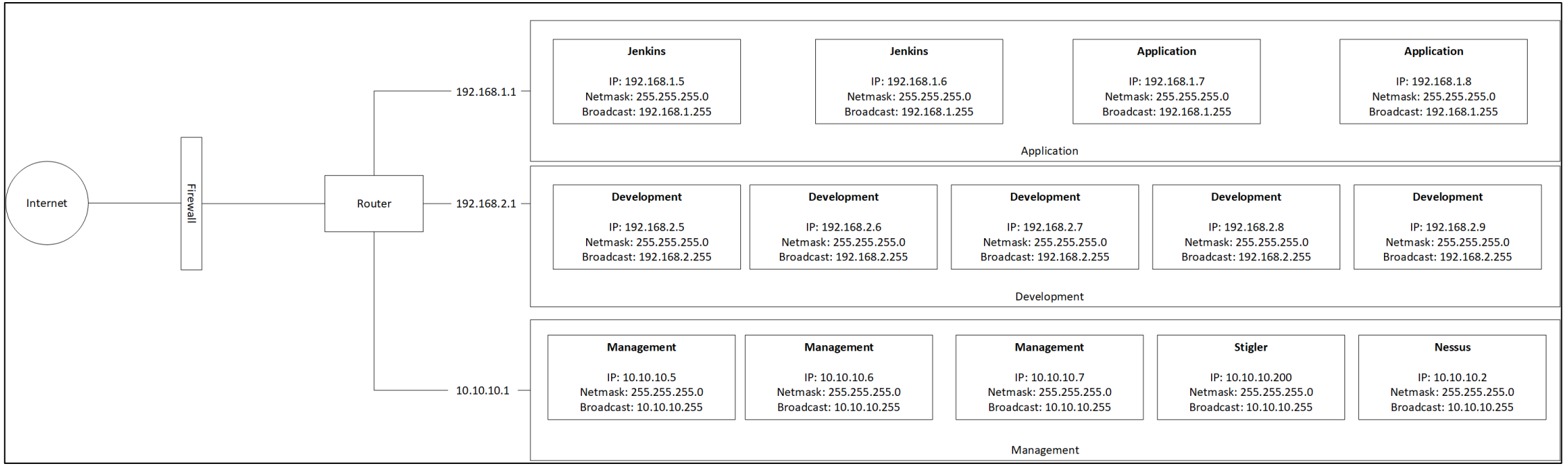
Temporal Metrics: Measure the current state of exploit techniques or code availability, the existence of any patches or workarounds, or the confidence that one has in the description of a vulnerability: Exploit Code Maturity (E), Remediation level (RL), Report Confidence (RC).

Environmental Metrics: Enable the analyst to customize the CVSS score depending on the importance of the affected IT asset to a user's organization, measured in terms of complementary/alternative security controls in place, Confidentiality, Integrity, and Availability.

Representative System

Raytheon
Intelligence, Information and Services

- Secure DevOps Environment instantiated on Raytheon's internal Cyber Range
- ~50 FOSS Products
- VMWare, STIG Hardened RHEL 7.5, Containers, etc.



Decision Support Process

Raytheon
Intelligence, Information and Services

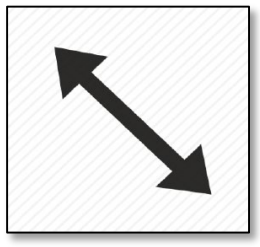
Assess System Risk

Baseline System Patches

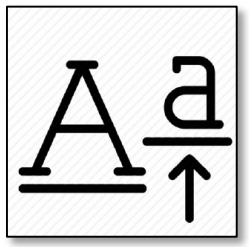
Rank Vulnerabilities / Patches

Provide Decision Logic for Validation

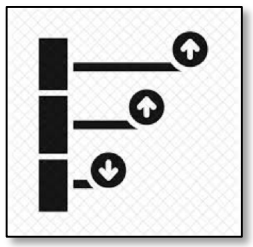
Provide Upgrade Path



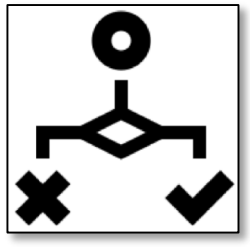
Using known vulnerabilities (e.g. NVD), display a risk assessment of a subject FOSS stack and network topology "system".



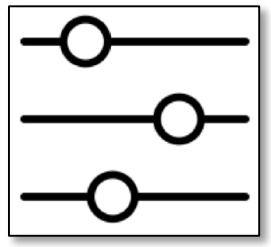
Utilizing the subject system, create a baseline of the patches required for a fully qualified (patched) system.



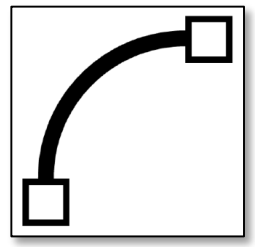
Using a graphing or data visualization interface, rank the most critical patches to the least critical.



Provide decision logic for the ranking utilizing text and visual components.



Create decision logic parameters to display tangible evidence that supports the ranking (severity, exploitability, etc.).

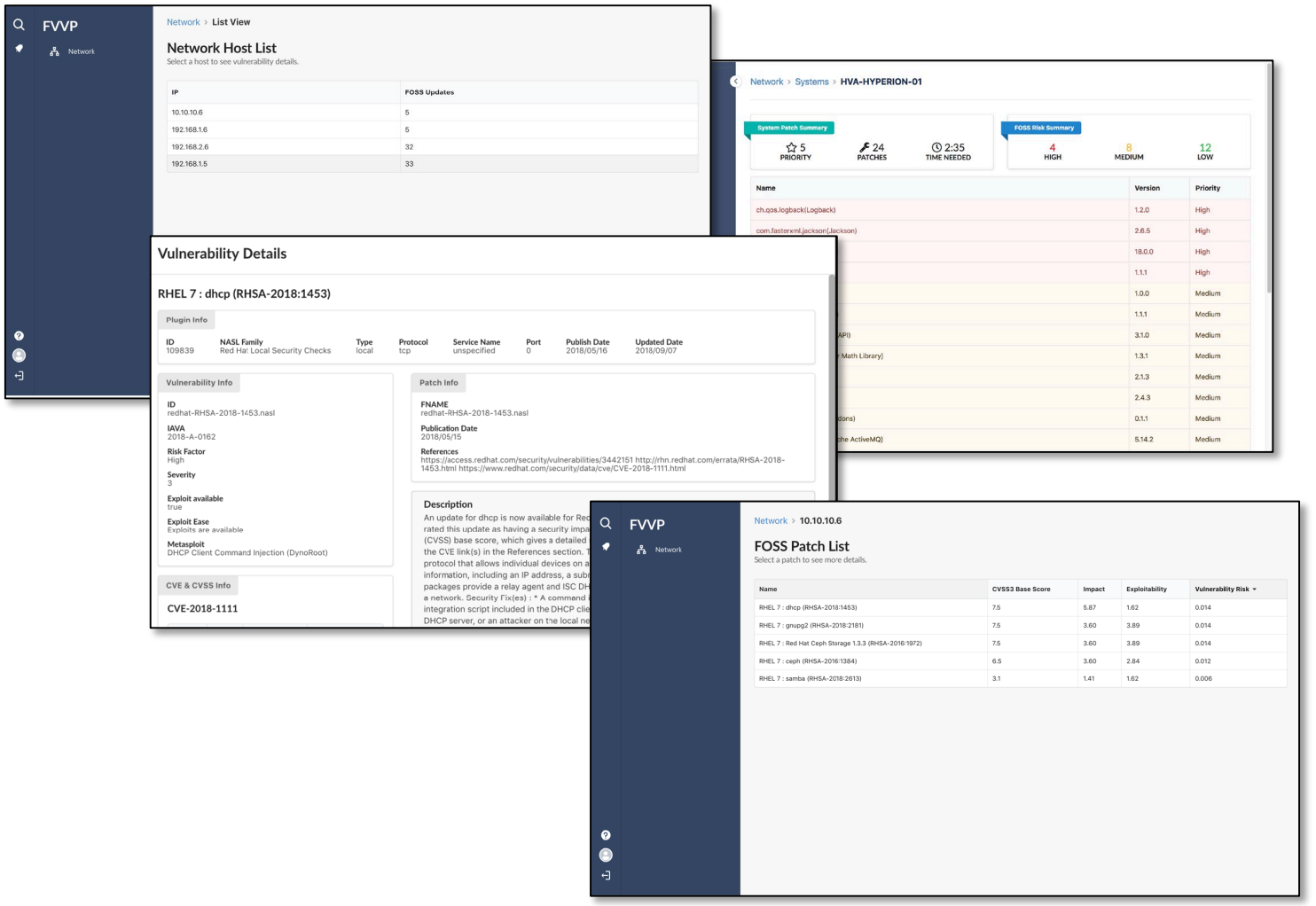


Provide the most optimal upgrade path (prescriptive) for the patch.

Decision Support System

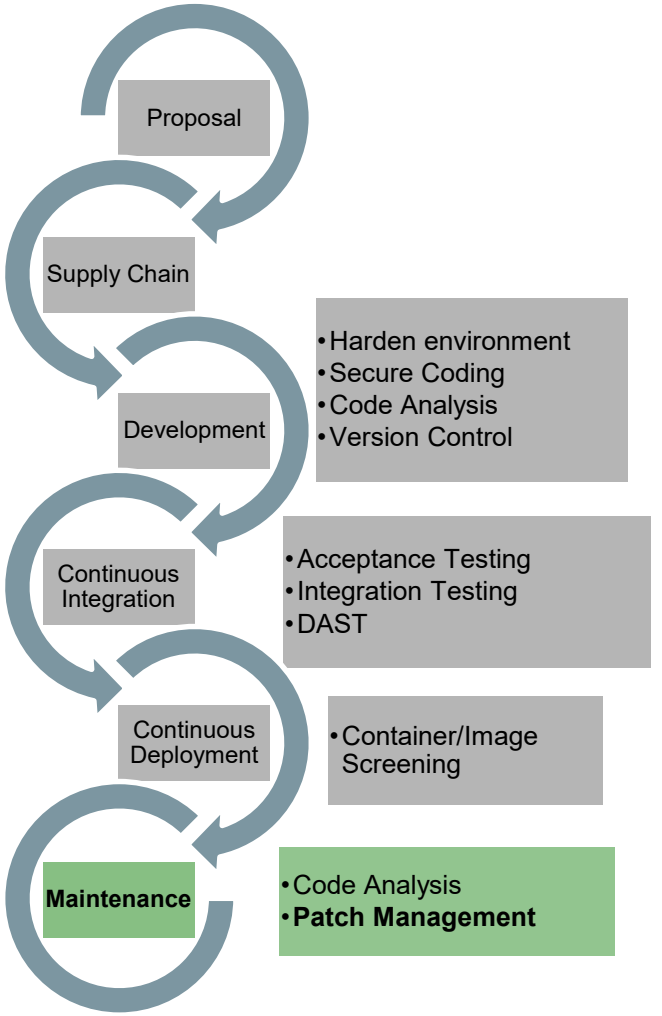
Raytheon
Intelligence, Information and Services

- Provides a comprehensive, prioritized list of patches
- Patches are organized in the order that is suggested the patch administrator apply them
- Can be deployed in AWS or as a containerized service



Future Work

- Algorithm Improvements: integrate additional vulnerability data sources
- UI/UX Optimization: Develop system status visualization
- Develop “patching pipeline” for use in Secure DevOps by automating:
  - the gathering of information
  - the screening process
  - patch testing and deployment



Example of a Secure DevOps Pipeline