KubeCon + CloudNativeCon 2024

FROM STANDARDS TO PRACTICE:

THE JOURNEY TO CONTAINER MATURITY

yelp

CARMEN CHOW

Software Engineer @ Yelp
ychow@yelp.com

TOM ROBINSON

Software Engineer @ Yelp
trobinso@yelp.com

Table of Contents

THE PROBLEM

Status Quo:

- Thousands of containers
- Over-reliance on alerts
- Inconsistently applied security checks
- No visibility into state of container security per environment

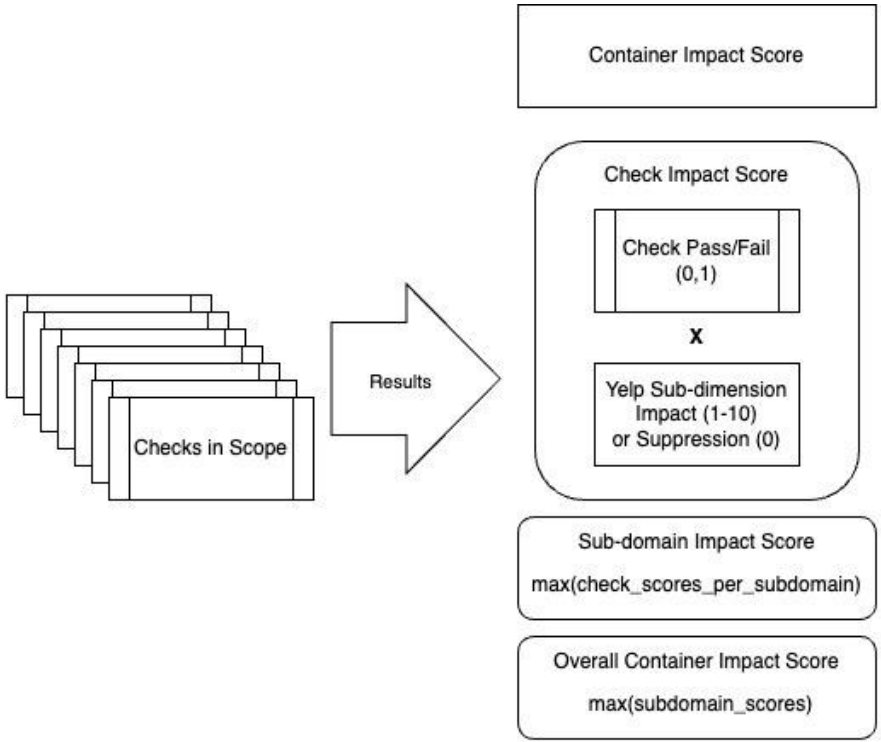
What we want:

- Determine:
  - Environment with the most vulnerable containers
  - How to improve security posture in those environments
- Enforce container security best practices consistently at scale

What we decided to do:

- Create a container maturity model
  - Tiered model measuring the robustness of container images, deployed images, and configuration
  - Roadmap to reach an acceptable level of container security
  - Enforce and track progress over time
  - Align with industry best practices

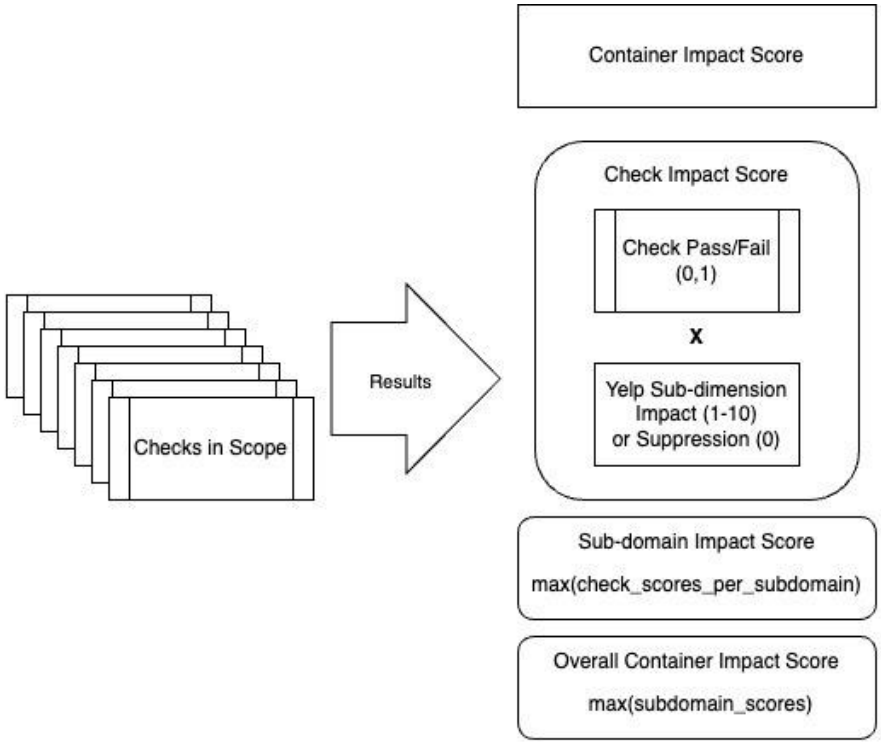
- Goal: Quantify improvements over time



How is this different from a VM Program?

- In a Vulnerability Management program, the focus is on
  - Patch/don't patch
  - Prioritization and triage
  - Risk management

- In Container Maturity:
  - Deploy/Don't Deploy
  - Operational signals within the container space specifically
  - Informs both individual container changes, aggregate, and control plane



THE JOURNEY

yelp

Reviewing Industry Standards

- Redhat, CIS, NIST

- Benefits
  - Build on reputable sources
  - Not reinventing the wheel

- Challenges
  - Some standards are difficult to enforce
  - Some standards are not applicable
  - Few guidelines on prioritization & enforcement at scale
  - Hard to find real world applications

Adapting Industry Standards

- Clearly define the **scope** of the model
- Filter out standards not relevant to us
- Modify/add new standards based on our processes
- **Goal**: Formal list of container security standards tailored to Yelp

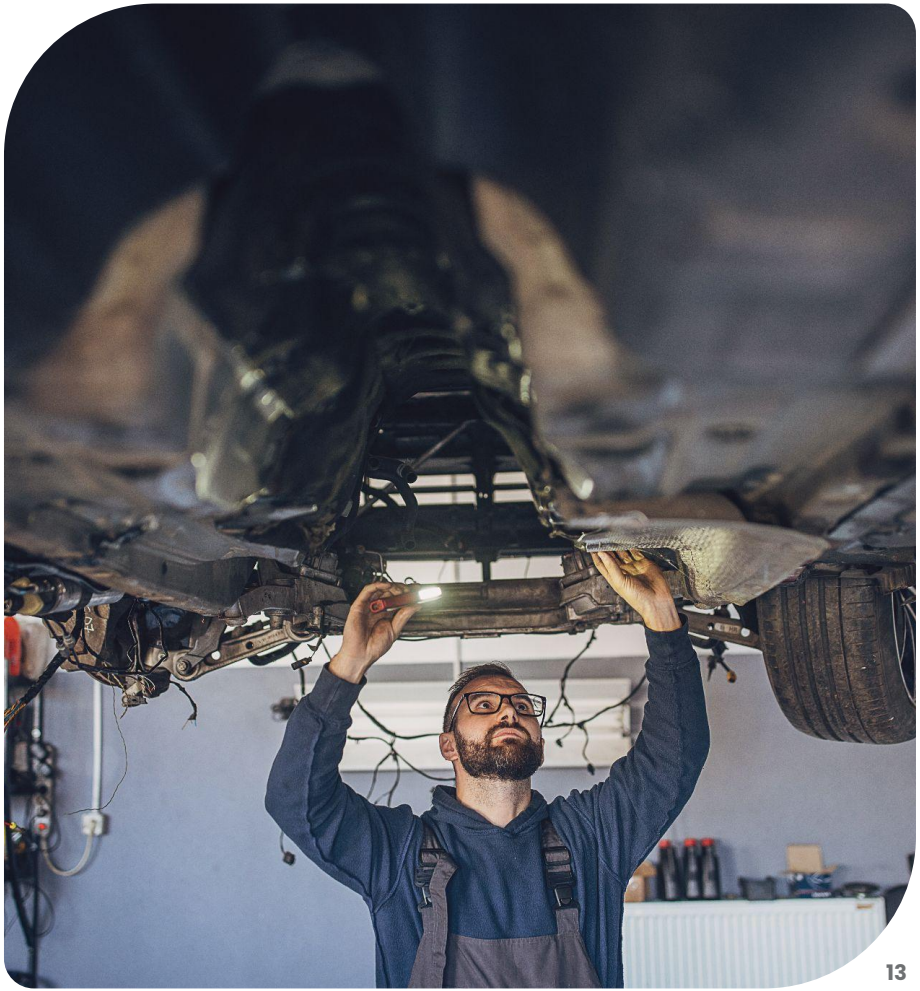
The model so far:

A curated list of container security standards

Considering Risk

- Some containers or security practices are more critical than others

- Container risk factors:
  - Access level
  - Data sensitivity
  - Impact of exploitation

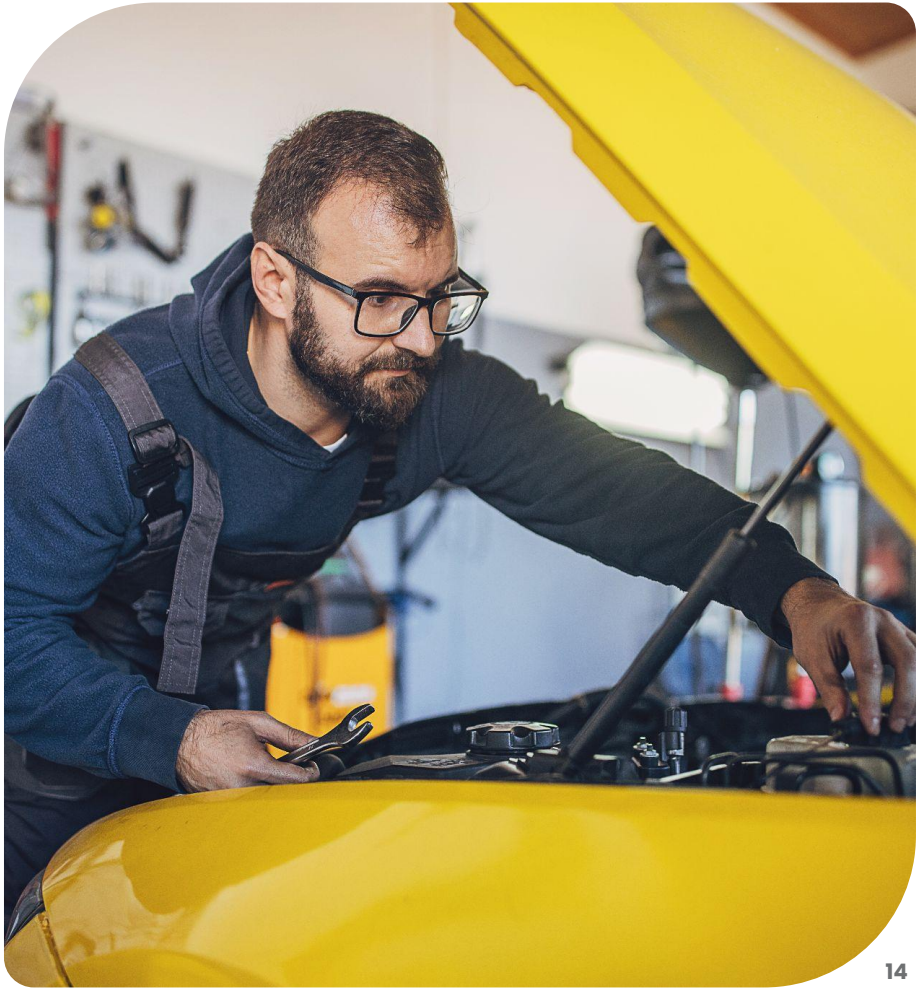


Handling Risk

- Assign **weights** to standards based on impact to:
  - Confidentiality
  - Availability
  - Integrity

- Higher weight = higher impact on container risk

- **Goal**: Prioritize remediation efforts



The model so far:

A curated list of container security standards

Risk management

Considering Scalability

- **Goal:** Enforce the model at scale
- Automation is key
- Assess container compliance via tools
  - → map output to our model
- Create metrics and dashboards from the results

The model so far:

- A curated list of container security standards
- Risk management
- Automated compliance checks

Container Maturity Metrics

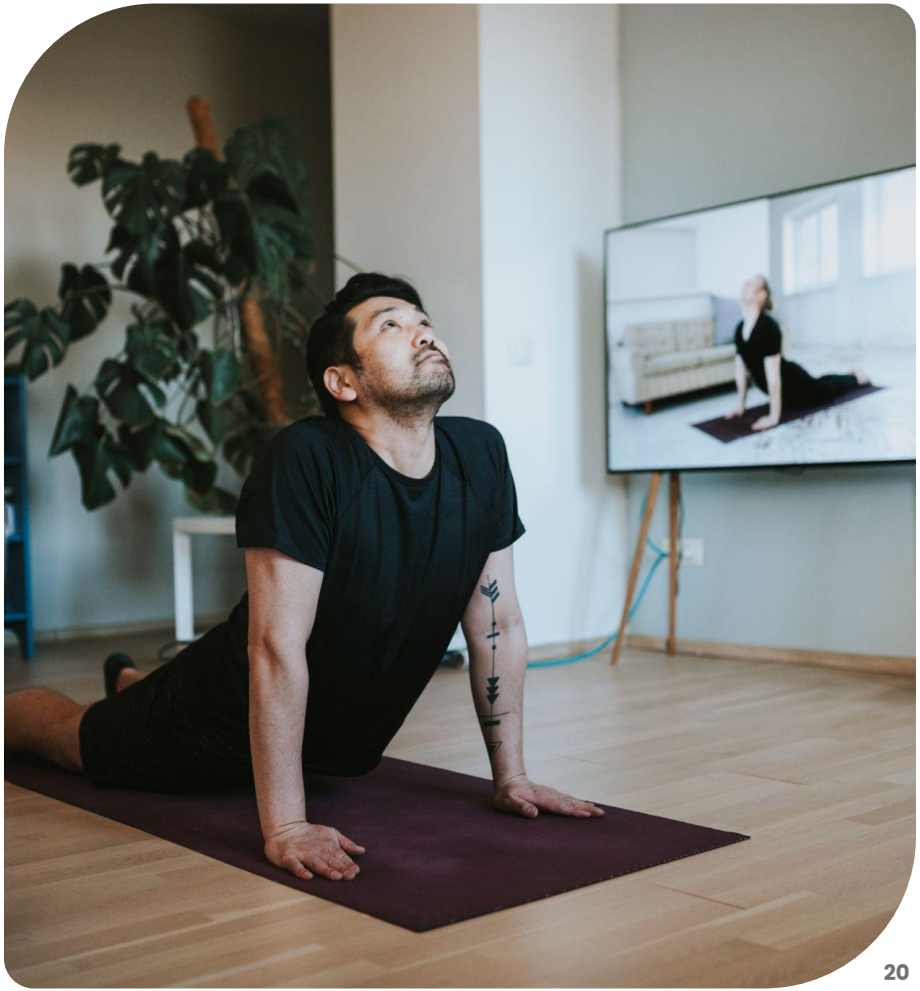
- Each standard graded as weighted pass/fail score
- Quantify container risk factors
- Container impact score - per-standard score & container risk factor scores
- Container maturity level - based on impact scores of all containers in a deploy environment
- **Goal**: Track direction & amount of change over time

The model so far:

- A curated list of container security standards
- Risk management
- Automated compliance checks
- Container scoring mechanism

Considering Our Users

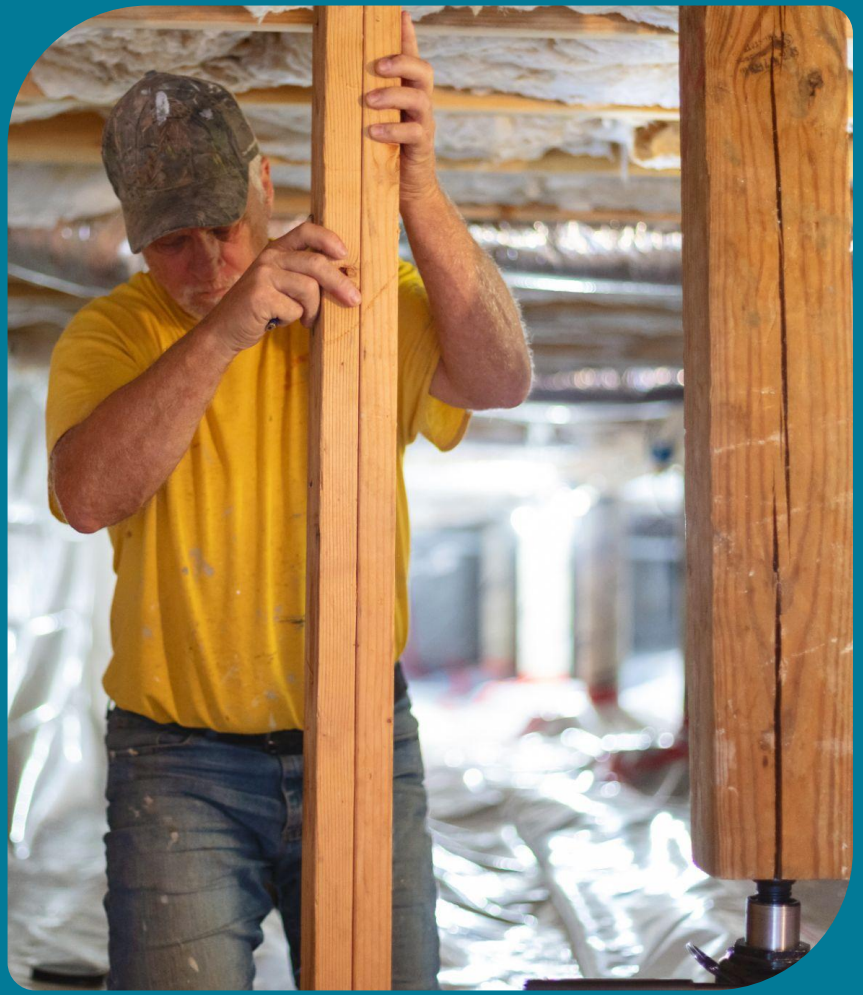
- Success of the model depends on **usability**
- Our users:
  - Developers
  - Leadership
- **Goal:** Make the model as intuitive as possible



User Experience

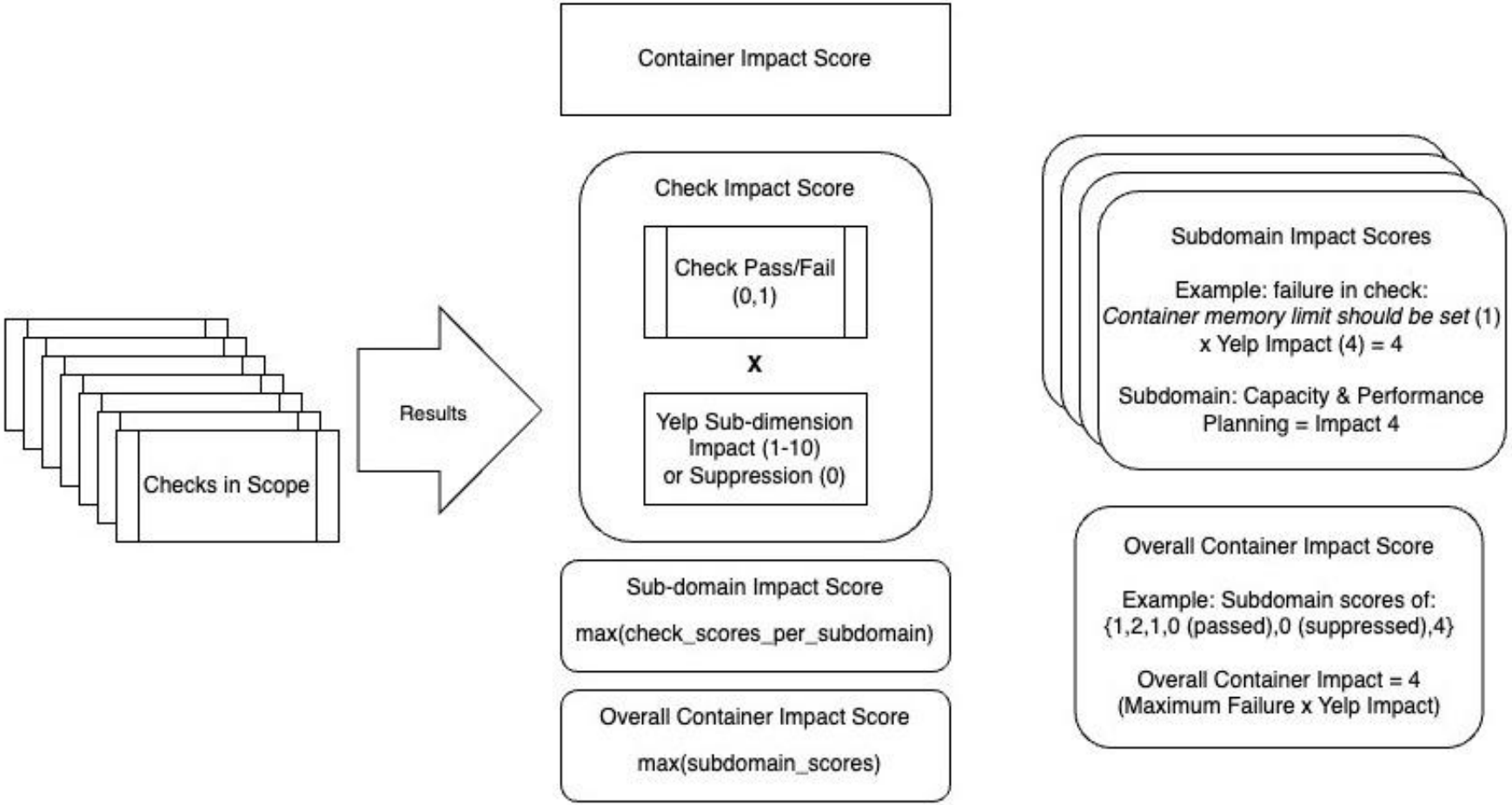
- Build the model into SDLC
  - Pre-commit hooks & linters for early detection
  - Block deployment on critical issues
  - Continuous monitoring with dashboards & alerts
- Clear remediation instructions
- Gradual rollout plan

THE MODEL



Container Maturity Model

- A curated list of container security standards
- Risk management
- Automated compliance checks
- Container scoring mechanism
- Compliance checks integrated into SDLC


Container Impact Score
Check Impact Score
Check Pass/Fail
(0,1)
X
Yelp Sub-dimension
Impact (1-10)
or Suppression (0)
Checks in Scope
Results
Sub-domain Impact Score
max(check_scores_per_subdomain)
Overall Container Impact Score
max(subdomain_scores)
Subdomain Impact Scores
Example: failure in check:
Container memory limit should be set (1)
x Yelp Impact (4) = 4
Subdomain: Capacity & Performance
Planning = Impact 4
Overall Container Impact Score
Example: Subdomain scores of:
{1,2,1,0 (passed),0 (suppressed),4}
Overall Container Impact = 4
(Maximum Failure x Yelp Impact)

Use cases

- Deployment checks for containers
- Risk assessment
- Compliance objectives
- Patch management (though consider a separate metric within a VM program)
- Dashboarding/leadership objectives

THE TAKEAWAYS

yelp

Lessons Learned

- Prioritize security & engineering resources
- Valuable to push for metrics collection–even when it's painful
- Abstraction is a useful tool

How do you apply this?

- Vulnerability & risk management
- Define key concepts in your model
- No one-size-fits-all
- Automate when possible
- Understand the most critical issues
- Remember the model must constantly evolve

THANKS!