Disclaimer

Presentations are intended for educational purposes only and do not replace independent professional judgment. Statements of fact and opinions expressed are those of the presenters individually and, unless expressly stated to the contrary, are not the opinion or position of RSA Conference LLC or any other co-sponsors. RSA Conference does not endorse or approve, and assumes no responsibility for, the content, accuracy or completeness of the information presented.

Attendees should note that sessions may be audio- or video-recorded and may be published in various media, including print, audio and video formats without further notice. The presentation template and any media capture are subject to copyright protection.
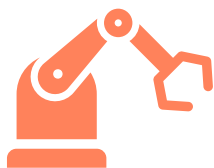
Why this talk, why now?

The problem

- Organizations understand DevSecOps today and plan to (or already are) implement it.
- But... face challenges on how to implement it on a large scale, measure its impact and communicate the outcomes

Dell case study

- Started as small-scale effort. Expanded incrementally on the SDL foundation
- Built with collaboration, experimentation, open mind, and technology
- Now pervasive across nearly all BUs across Dell

Takeaways for you

- Take insights from what worked (and didn't) for us. No need to reinvent the wheel
- Cherry-pick and practice architecture patterns and approach as relevant to your org
- Help you broaden the horizons beyond point solutions and think holistically

Outline of today's talk

1 Dell's scale and context

2 Problems driving requirements

3 Strategy, execution & outcomes

4 Your action plan

By product heterogeneity

Hardware

(personal computers, server, networking, storage…)

Ecommerce

Public facing ecommerce site (dell.com)

Internal apps

Non-Internet facing internal apps

APEX

As-a-service offers

By maturity journey

Long release lifecycles ———●————————◆——————— Agile. Frequent or continuous deployment using CI/CD

Legacy monolithic applications ———————◆————————●— Modern containerized microservices

Risk tolerance: High ———●————————◆——— Risk tolerance: Low
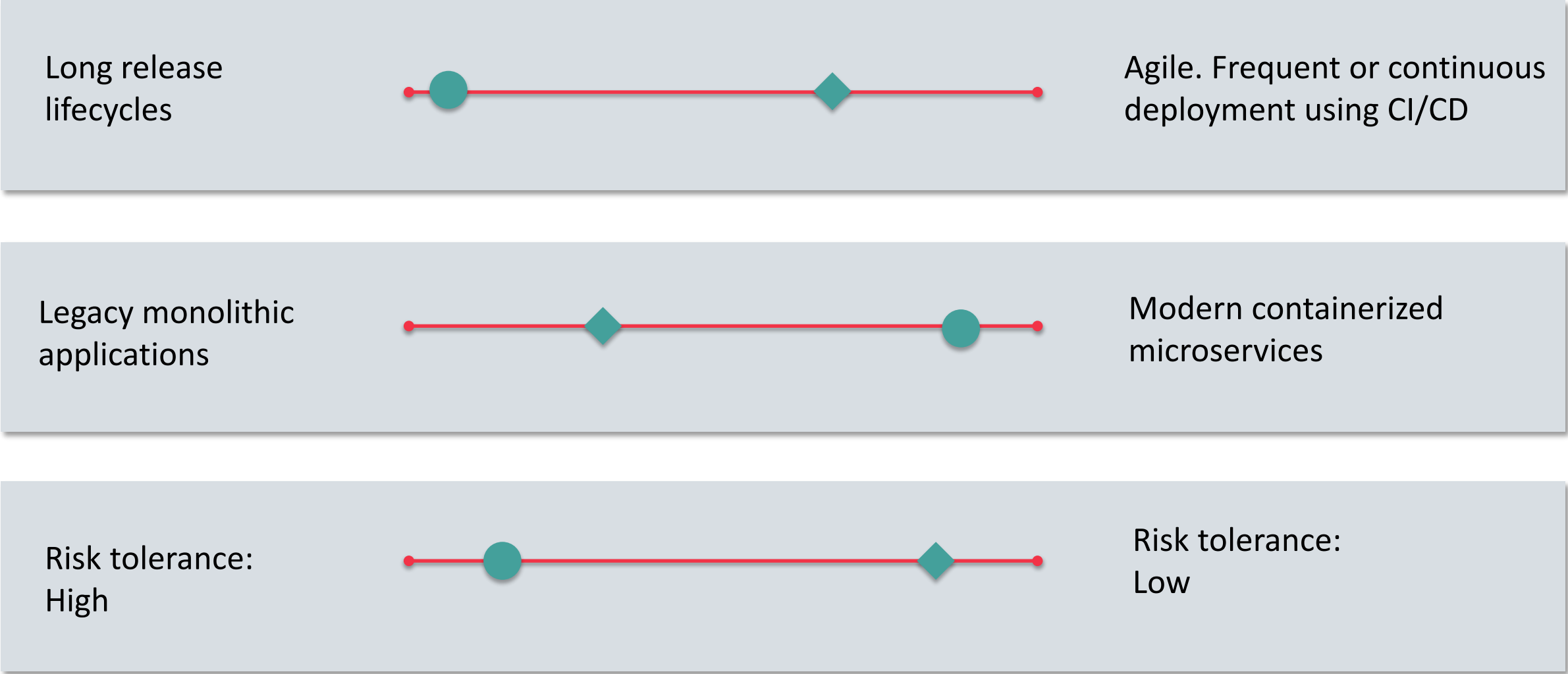
By numbers

of products and applications to be secured

# of code repos
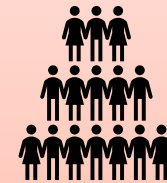
# of engineers

By tech stack and tools
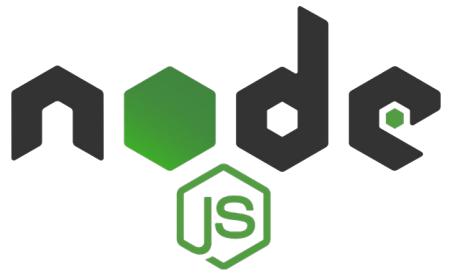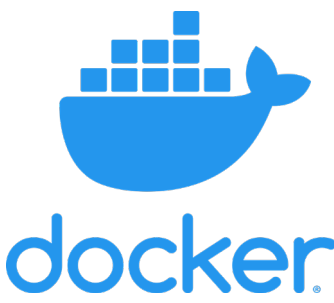
Frustrating dev and ops experience

Feedback to DevOps on security take too long →

Security scan results in PDF, email and not consumable in a pipeline →

Handling of false positives further impedes agility →

Resource intensive

Manual security assessments are...

✓ thorough		and necessary ✓	
✗ but take time		and are expensive ✗	

... so need a multi-option approach

Did you do what the security control says?

Is the security control implemented in the code, app, pipeline?

Is the security control being bypassed?

Are teams deploying to production despite a critical control failure?

Limited observability

1 No instrumentation in the dev environment to collect security data.

2 Fully dependent on security scanning tools

3 Reporting via spreadsheets and slides

4 Manual and time-consuming determination of security posture

And...

the challenge of solving all the above at scale

7-point implementation strategy

1 Solid SDL* foundation

2 Multiple consumption options

3 Customer agnostic architecture

4 Act one team everyday

5 Integrate at LCD**

6 Optimize for DevOps experience

7 Instrument for measurement

*Secure Development Lifecycle
**Lowest common denominator

Build on existing SDL foundation

1

OBJECTIVES

SDL
for Scale

Embrace & Adopt.
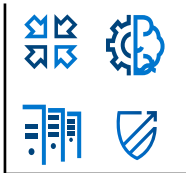
Measure and progress Apps. against four Maturity Practice Levels
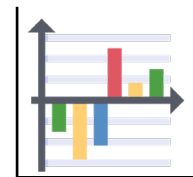
Effectively reduce risk prior to release (GA)

| L1 | L3 |
| L2 | L4 |
| L5 |

Invest and train security champions

SERVICES

Champion

Security Engineer

DevOps & SRE Fostering Continuous & Frictionless Security

Champion

Security Engineer

Threat Modeling

Static Code Analysis

Open Source Component Mgmt.

Container Scanning

Web Security Testing

Network Vulnerability Scanning

Independent Security Testing

SDL Security Assessment

| Design | Development | Testing | Release |

FOUNDATIONS

62+ Controls

One Standard

Compliant

Standard

Cutting Edge

Leading

Four Maturity Practice Levels

L1 Awareness

L2 Knowledge

L3 Knowledge/Skill change

L4 Demonstrate Skill

L5 Demonstrate Skill

Five Levels of Security Training

Offer multiple consumption options

1 SDL Engineer Led	**2** SDL as Self Service	**3** SDL as API

SDL Engineer Led

Led by Security (SDL) Engineer

Manual

For high value applications

Most comprehensive

SDL as Self Service

Led by Security Champion

Manual

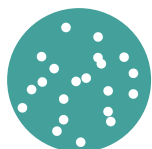For all applications

Comprehensive

SDL as API

Performed pragmatically via Git workflows and CI/CD events

Automated

For apps with frequent & automated deployments

Balanced

RSAConference2022

Customer agnostic automation architecture

A	Maintained by Dell DevOps & SRE teams
B	Provided by Product and Application Security

A

Stages

Security Design and Architecture ⇢ Secure Code

Continuous Build and Integration

Continuous Test

Continuous Delivery and Deployment

Continuous verification, monitoring & redeployment

DevOps and SRE teams' development and deployment environment

On-premise, Cloud deployment env.

Apps, VMs, Containers, Storage, Networking

B1 Security of DevOps processes

B2 On roadmap Security of SRE provisioned infra

Multiple Integration Options for DevOps teams

SEAL – SDL Enforcement and Automation Library

Checkmarx SD ELEMENTS BLACKDUCK BY SYNOPSYS HCL AppScan Fuzzing Tools Twistlock

checkov by bridgecrew terrascan by accurics Twistlock Nessus nexpose Qualys

K8S ANSIBLE vmware Terraform

SDL Foundation

Customer agnostic automation architecture

3

A

Security Design and Architecture

Stages

Secure Code
(IDE & Git repo)

GitHub
GitLab

Continuous Build and Integration

Continuous Test

Continuous Delivery and Deployment

Continuous verification, monitoring & redeployment

On-premise, Cloud deployment env.

Triggers

Jenkins

CI/CD

Static application code scan
IDE/Git PR-MR triggered

Static application code scan
Pipeline triggered

Software composition Scan

Container image Scan

Dynamic app scan

Fuzz testing

Runtime container vuln. defense

Runtime app vuln. defense

Static infrastructure code scan

Static infrastructure code scan

IaC policy validation (pre-deploy)

Infrastructure vuln. scan (on demand)

Continuous Infra. vuln. scan

Runtime infra. cfg drift monitoring

Apps, VMs, Containers, Storage, Networking

B1

Security of DevOps processes

B2   On roadmap

Security of SRE provisioned infra

Integration Options for Dell Products and Dell Digital

API   Container Image   CI/CD Plugins   node Raw Code   python Raw Code   Python Pkg

SEAL – SDL Enforcement and Automation Library

Unified SDL Scanning and Controls Verification Interface

API based SDL enrollment   Scan state management   Taxonomy alignment

Scan Results Interpretation

Security Policy as Code Implementation   Invoke Scans   Push data to Metrics

Unified SRE Security Policy Verification Interface

Dell Security Advisory as code   Policy validation   Patch management

Validation and enforcement of golden images   Configuration drift mgmt

SDL Verification Tool Integration Layer   Dell Products / Infrastructure API

SDL Verification Tools, Dell Systems & Infrastructure

API / CLI

Checkmarx   SD ELEMENTS   BLACKDUCK BY SYNOPSYS   HCL AppScan   Fuzzing Tools   Twistlock

checkov by bridgecrew   terrascan by accurics   Twistlock   Nessus   nexpose   Qualys

K8S   ANSIBLE   vmware   Terraform

SDL controls as code (OSCAL compliant YAML)

System (infrastructure) security policy as code (OSCAL compliant YAML)

4

Act as one team everyday

- Do not "confront" – Build partnerships instead

- Establish joint scrum teams

- Get into a common backlog

- Resource challenged?
  - Bring security champions to the challenge
  - Reinforce through security awareness and training

5
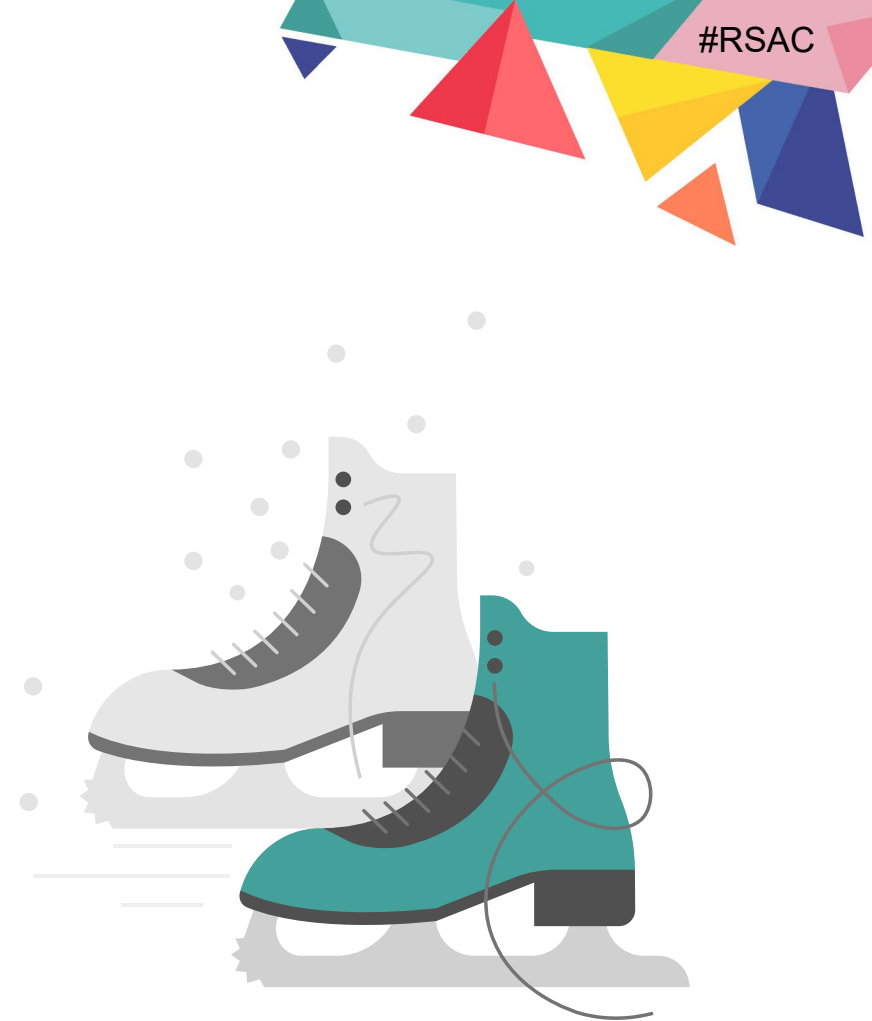
Integrate at the *Lowest Common Denominator*

Security **+** Platforms
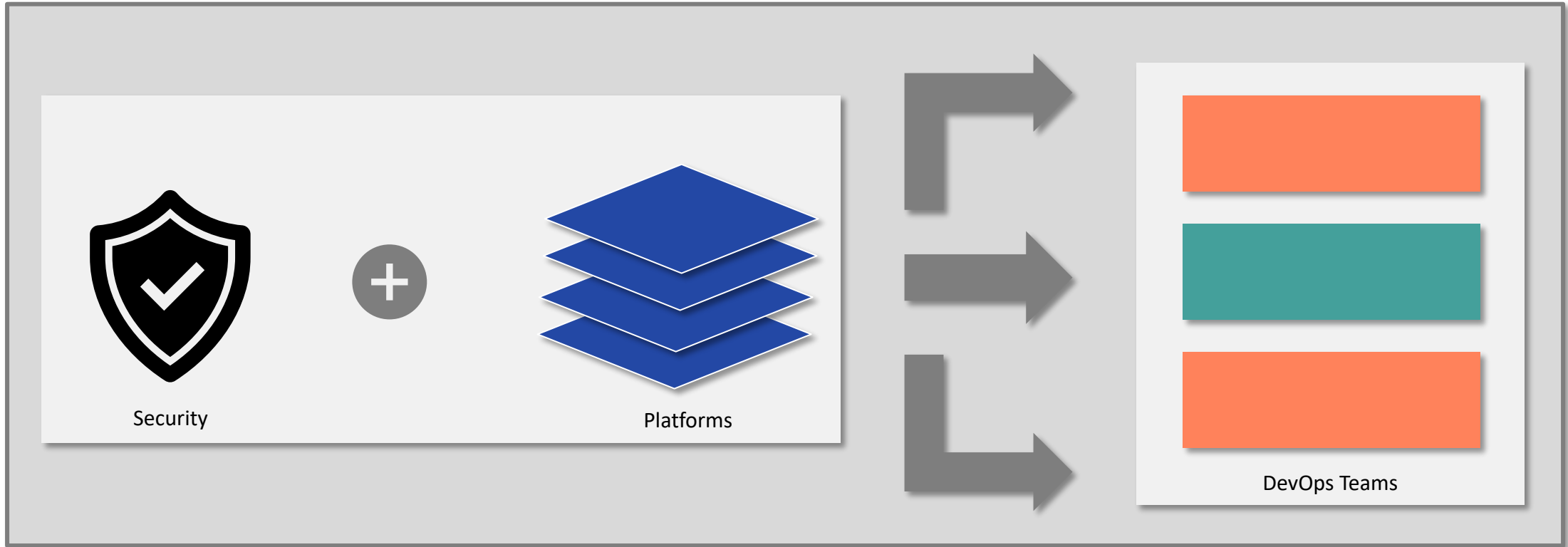
DevOps Teams

Identify opportunities for platform integration...

...without losing sight of the "downstream" DevOps teams

5

Our lowest common denominators

A win-win-win scenario

Delivers

Git + CI/CD platform
Standardized blueprint for commonly used patterns
Automatically invoked security scans
Apply security policy programmatically

Telecom (TSB)

ISG Products, APEX

Dell.com, Internal apps

Dell Technology Services

Accomplishes

platform with security scan built-in and run automatically
Mitigates security findings in code, open-source components, container images & applications

ISG Developer Ready Platform

Dell Digital DevOps Platform

Delivers

Security scanning services
Reusable SDL automation artifacts
Security policy as code
Threat modeling service
Secure architecture guidelines
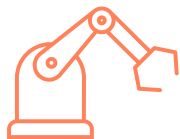
Product & Application Security

Feedback Channels:
- Verbal and subjective
- Programmatic (metrics)

Optimize for developer experience

Embedding a security activity in not enough

Integration must be optimized to reduce friction

N factor model for optimization

Scope	Be precise about "what" needs to security verified
Branch	Not all branches need same treatment for security verification
Stage	Allocate security activities appropriately among CI/CD/CD
Trigger	Security activity triggered with: Time / Git event / Pipeline event
Frequency	The sweet spot: how often to perform a security activity
Enforcement	Audit mode vs. Strict mode (block the merge/build/pipeline)
Packaging	As package / As container image / As code

Instrument for measurement

Power BI — Security KPIs Reporting

Comprehensive Product Specific Maturity Dashboards

Reporting and Visualization Layer

Data Storage & Processing Zone

Jira Software — SDL Issue Mgmt

System Taxonomy & Registry

Security Champions Enrollment

Training Records

Product Incidence and Response Data

SDL Services Data Sources

GitLab / CI/CD — Code Repositories and CI/CD Pipelines

GitHub / Jenkins — Code Repositories and CI/CD Pipelines

DevOps Platforms

Security Tools

SDL (Manual) Control Assessment

Application Architecture Security Requirements

Threat Modelling

Static Code Analysis

Open-source Component Management

Container Image Scan

Dynamic Application Scan

Code Signing

Vulnerability Scan

VIRTUAL SECURITY CONSULTANT

Checkmarx

BLACK DUCK BY synopsys

Twistlock

HCL AppScan

nexpose

Nessus

Qualys

RSA®Conference2022

Outcomes and path forward

We are not done yet

Outcomes

Frictionless Experience

of places to get security feedback

1

Speed of getting security feedback

5 days
5 min

Adoption at Scale

of BUs onboarded

> 90%

of products enabled

> 1000

of projects/repos enabled

> 10,000

Security Effectiveness

of products/apps with security issues mitigated before deployment

> 600

of critical/high security issues discovered and mitigated

> 400

Outcomes

Outcomes

All Applications ▾

Org Comparison   Trending Data   Scan Types   **Scan Type Trending Data**

Scan Types:
- ☑ Static App Security Scan ☑ 3rd Party Code Scan
- ☑ Dynamic Application Scan ☑ Artifact Vulnerability Scan

Fiscal Month/Year
Trend Data is captured monthly at end-of-month
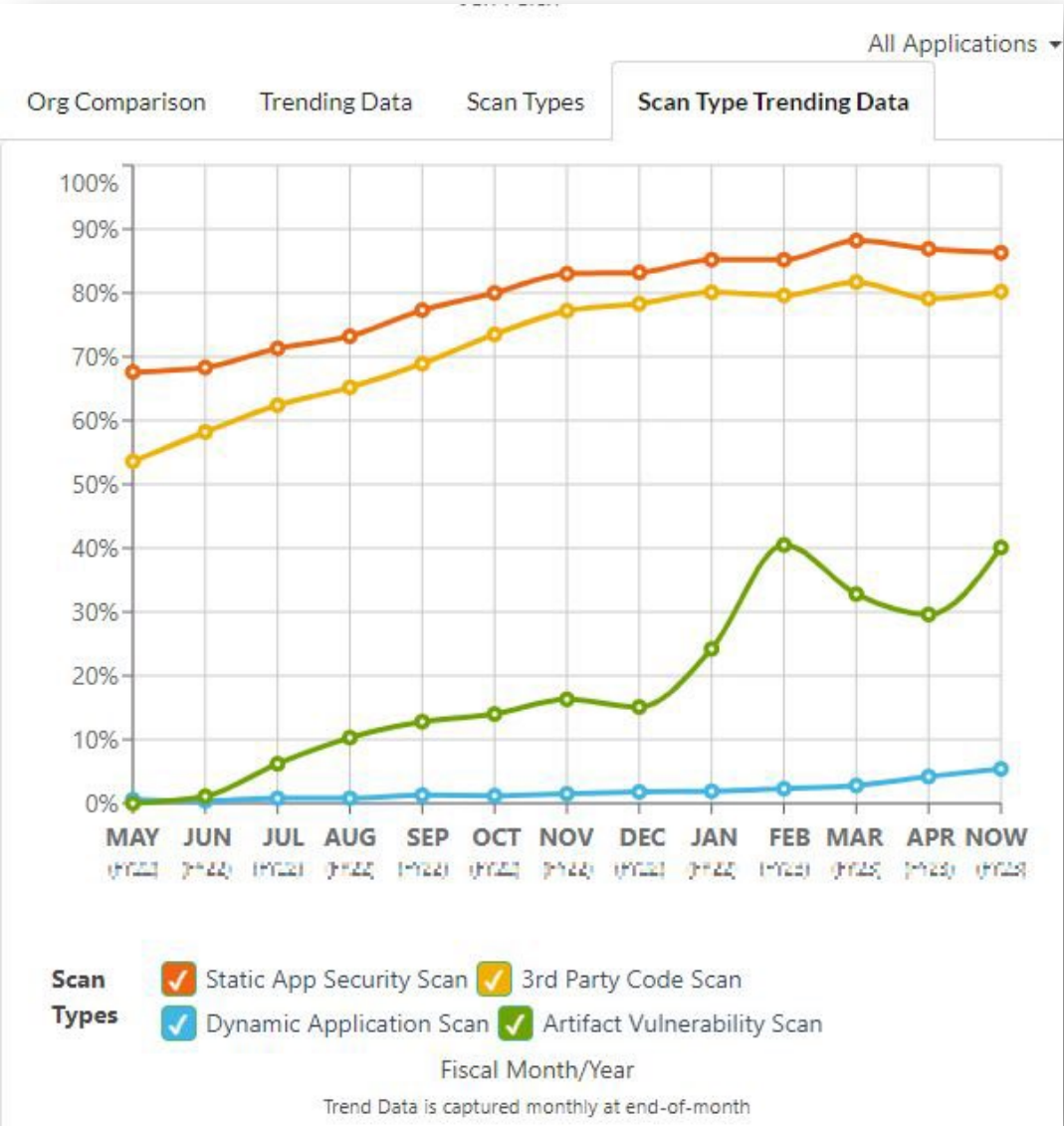
Build & Verify   Non-Prod Deploy & Validation   Scan History

TIP: Click on each Dimension Feature Item in the list below to see the validation's purpose, criteria, validation method, and support details!

Dimension Feature Item Name	Validation	Pipeline Id	
Static Application Security Testing	✓	9797571	⚠
3rd Party Code Scan	✓	9797571	⚠
Artifact Vulnerability Scan	✓	9797571	⚠

Build & Verify   Non-Prod Deploy & Validation   **Scan History**

	Scan run in last		
Security Scan Type	**30 Days**	**180 Days**	**12 Months**
Static Application Security Testing	☑	☑	☑
3rd Party Code Scan	☑	☑	☑
Artifact Vulnerability Scan	☑	☑	☑
Dynamic Application Scan	☑	☑	☑

Outcomes

Outcomes

Reimagining the future

Frictionless enrollment to SDL tools

Slash enrollment time from days to seconds
Superior customer experience
Near-zero human intervention   Focus instead on high impact tasks

Rapid Response Support Model

Conversational support via a chatbot. Instantaneous response to questions/issues

Secure Development Lifecycle

Security Control Verification

Expand beyond security testing automation
Enable DevOps to verify all (qualified) SDL controls with minimal manual effort

Seamless app security testing

Security feedback time reduced from days to minutes
In context feedback to developers

SBOM & EO enablement

Automated generation of software bill of material and recorded in source of truth

Seamless IaC and NW security testing

Enable SRE and Ops team to secure IaC code
Automated network and systems scanning

Legend

★ Available   ☆ Ideation

★ Active new initiatives

D&LL Technologies

Dell – Internal Use – Confidential

RSA®Conference2022

Your action plan

For implementing or scaling DevSecOps

Assess and measure

30 days

- Time taken by DevOps/SRE teams to get security feedback: minutes/days/weeks

- Time and approach taken by Security org to get security posture
  - Security issues discovered, reported, fixed
  - Days of open before remediation

- % of security bugs slipped through SDL and ended up in production
  - Worst case: caught by customers or external entities

- SDLC stages where are security and DevOps interactions happen: (early / mid /later)

Pilot

3 to 6 months

- If offering only manual SDL assessment: pilot with self-service option

- If already offering self-service SDL assessment
  - Start proof of concept with a DevOps team. Ideally with a platform team

- Embed a security engineer or a security champion in the development team's scrum process

Retrospective and scale

1 year

- Remeasure and compare with metrics from "Assess" stage

- Conduct joint retrospective sessions with development and security teams
  - What worked or didn't work

- Plan for scaling across larger number of teams across organization

RSA®Conference2022

Questions