disruptiveOps

Establishing Database Resiliency

Conceptual Understanding

Jonathan Jenkins (Jonathan, Jarhead, Gunny)

Professional

- Founder of DataOpus, disruptiveOps, and Optimal Zero Firearms, Jenkins Advisory Group
- Strategic Advisor for Venture Capital Company
- Part-time University Professor (WUST)
- 25+ years software/database development (lie)
- 15+ years project management
- Patent Holder
- Inventor (DataOpus, Iron Pipeline, Lifecycle Sleuth, Digital Continuity Framework)

Principles & Process

- Daily practitioner of XP, Lean, Kanban, & GSD
- Agile Architect / Coach / Trainer
- Certified Coach (Agile, Team Facilitation)
- Kanban Management Professional (I/II)
- Lean (Yellow)
- Certified Scrum Professional (CSP)
- Certified Scrum Master (CSM)
- AgilityHealth Facilitator (AHF)
- ~~Scaled Agile Framework Consultant (SPC)~~

Education

- M.S. Information Technology (Upsilon Pi Epsilon)
- B.S. Computer Science
- A zillion certificates – what 'ev
- #NoExpert

Service

- Gunnery Sergeant of Marines
- Iraq War Veteran
- Air Traffic Controller, Platoon Sergeant, Marine Air Ground Task Force Planner, Lessons Learned

Agenda

1. Common Language
2. Resiliency
3. The Problems
4. Database Resiliency
5. Humans are the Problem
6. Improve
7. Wake Up

disruptiveOps

Common Language

Common Terms

Resiliency The ability to maintain required capability in the face of adversity.

Vulnerability Weakness in an information system, system security procedures, internal controls, or implementation that could be exploited or triggered by a threat source.

Weakness Defect or characteristic that may lead to undesirable behavior.

Flaw A defect that can lead to unexpected consequences.

Smell Surface-level indication that code, data, architecture, etc. might not be as clean or maintainable as it could be.

Common Terms

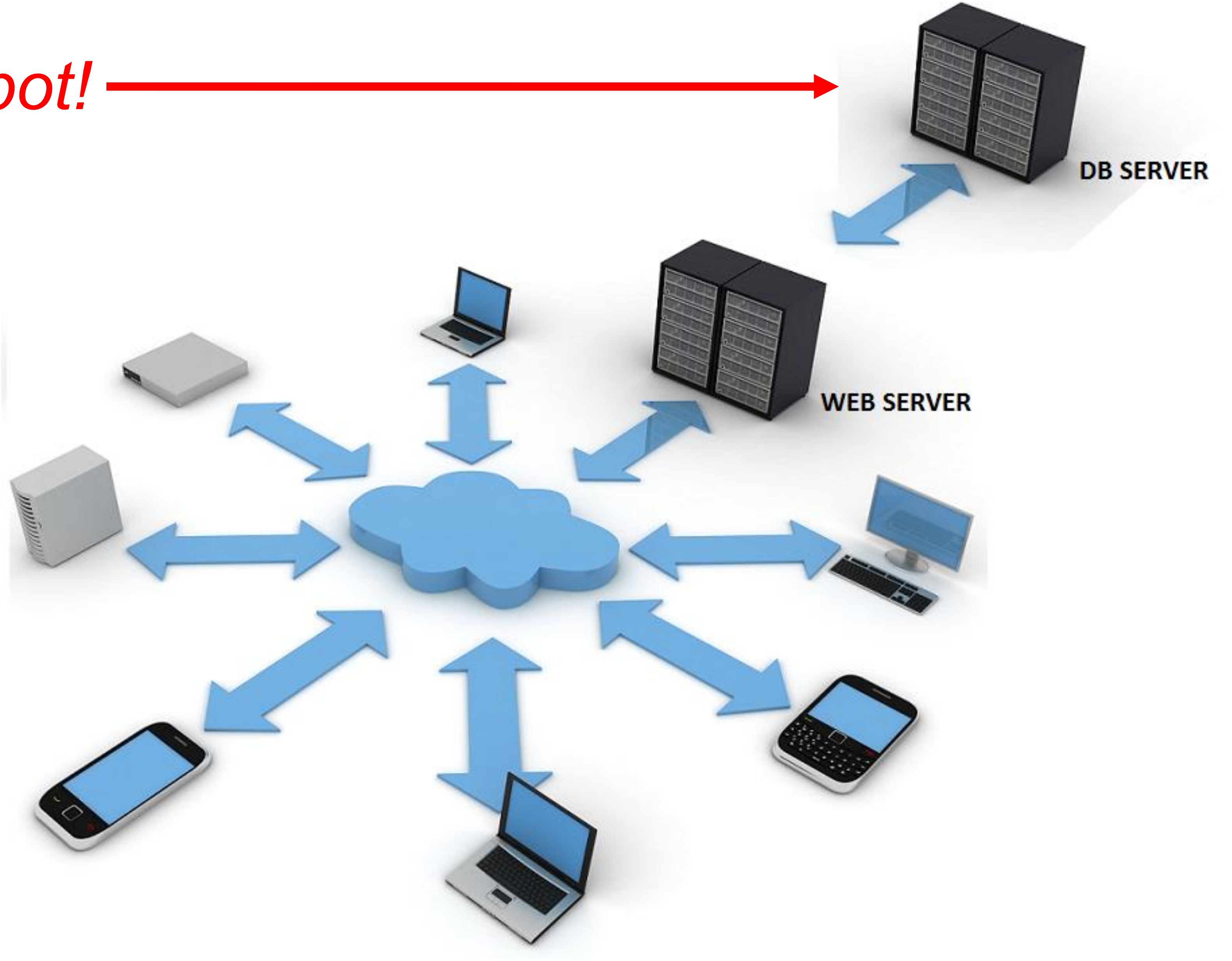
Non-Functional Requirements	Requirement that specifies criteria to judge the operational aspects of a system, rather than specific behaviors.
Risk	Uncertain event or condition that will result in a negative impact **IF** realized.
Issue	Event or condition that **HAS** already occurred and is causing a negative impact.
3rd Party Risk	Risks associated with vulnerabilities, weaknesses, and flaws from external entities, such as vendors, service providers, and partners, that interact with an organization's systems, data, or operations.
ABCs	Ambiguity, Bigness, and Complexity

Why We are Here Today

Every Endpoint is Susceptible

Massive Blindspot!

- *Externally*
- *Internally*
  - *Adversary*
  - *Clown*



Major Problems

25

Average Number of Vulnerabilities per Database (US)

75%

Vulnerabilities at Least 2 Years Old

80%

Public Exploits Occur Before CVE is Published

50%

Known Breaches to Non-Publicly Accessible Databases Occur at the Application Layer (SQLi)

The organizational (data) problem

- ✓Integrity
- ✓Completeness
- ✓Loss

- Loss of Trust
- Financial Loss
- Operational Disruption
- Regulatory Penalties
- Compromised Security
- Recovery Cost
- Decision Making Errors
- Loss of Intellectual Property
- Damage to Brand Image
- Ineffective Artificial Intelligence / Machine Learning

disruptiveOps

Resiliency

Cybersecurity vs Resiliency

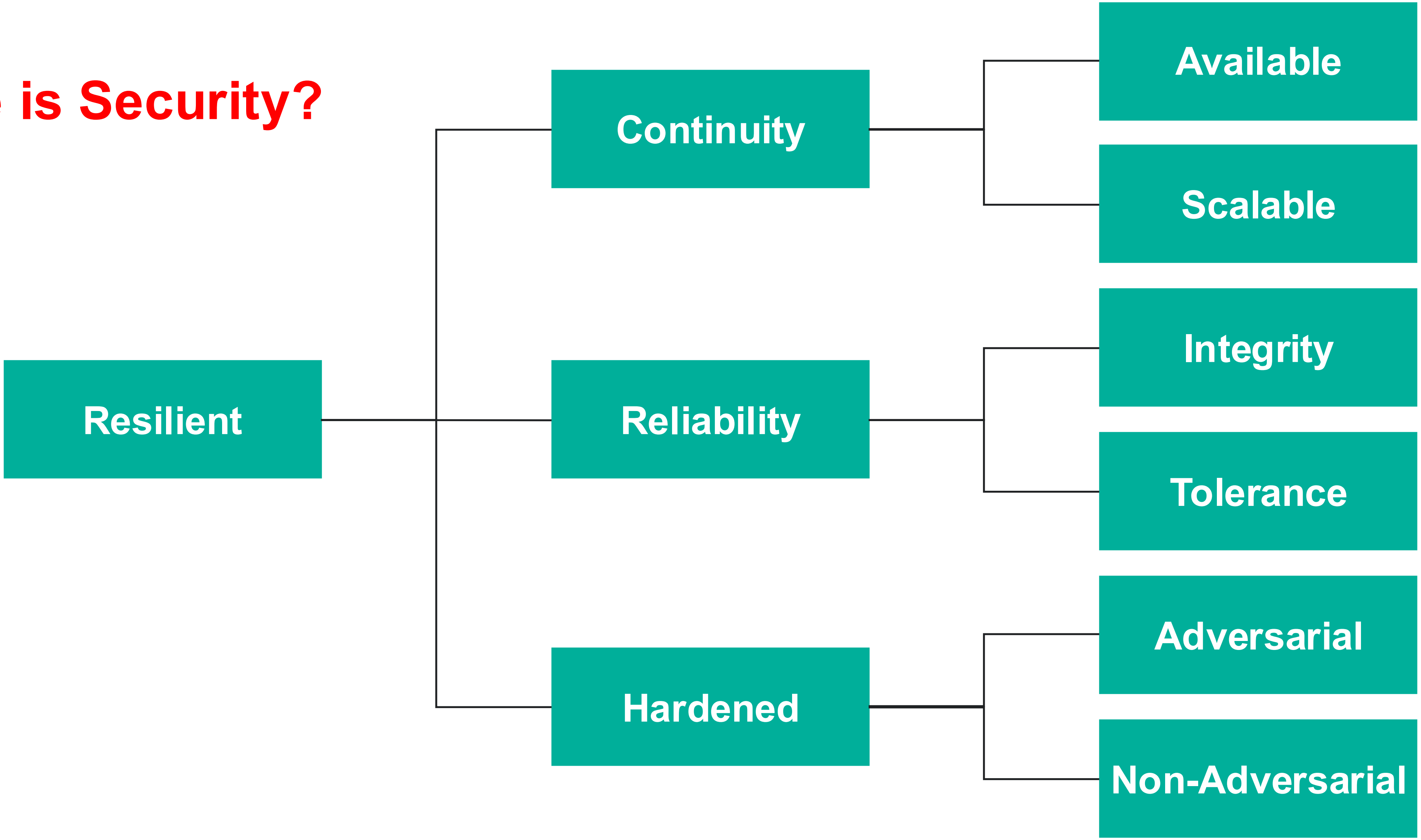
Cybersecurity is NOT resiliency.

Cybersecurity is necessary but not sufficient.

Cybersecurity	The process of protecting information by preventing, detecting, and responding to attacks.
Resiliency	The ability to prepare for and adapt to changing conditions and withstand and recover rapidly from disruption of database services.
	Resilience includes the ability to withstand and recover from deliberate attacks, accidents, **design flaws**, or naturally occurring threats or incidents.

Resiliency Terms

Where is Security?



disruptiveOps

Database Resileincy

Database Resiliency

Is Not

- Just a cool new buzz word
- Just cybersecurity
- Just data recovery
- Policies & Procedures

Is

- Limited Cybersecurity Threats
  - No vulnerabilities
  - Possibly acceptable weaknesses
- Organizational Continuity
  - No data loss
  - Little to no data integrity issues
  - Limited concerns with NFRs

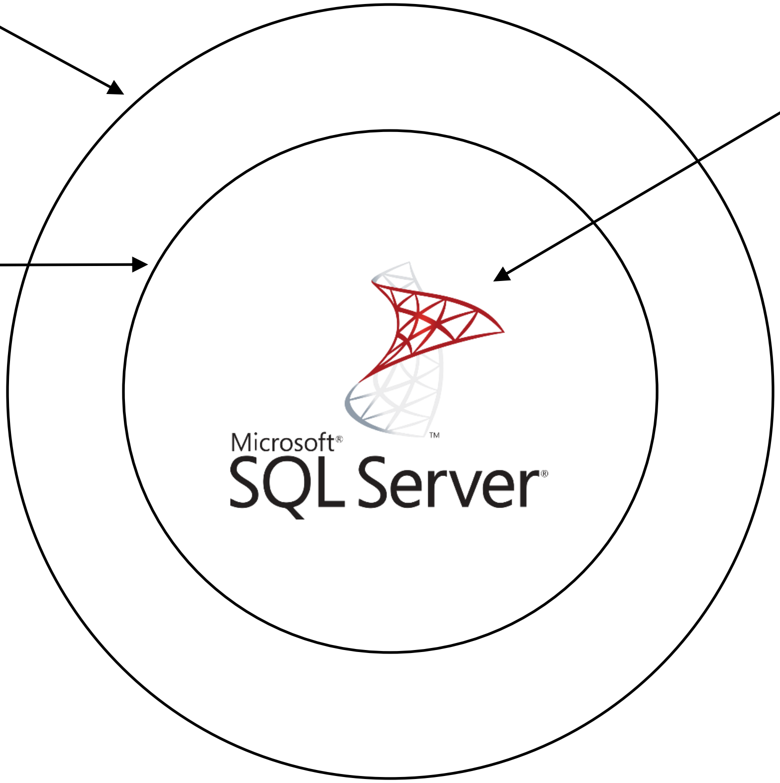
Database Security vs Database Resiliency

Identity & Access Management

Attack Surface of Database

Adversarial

Non-Adversarial

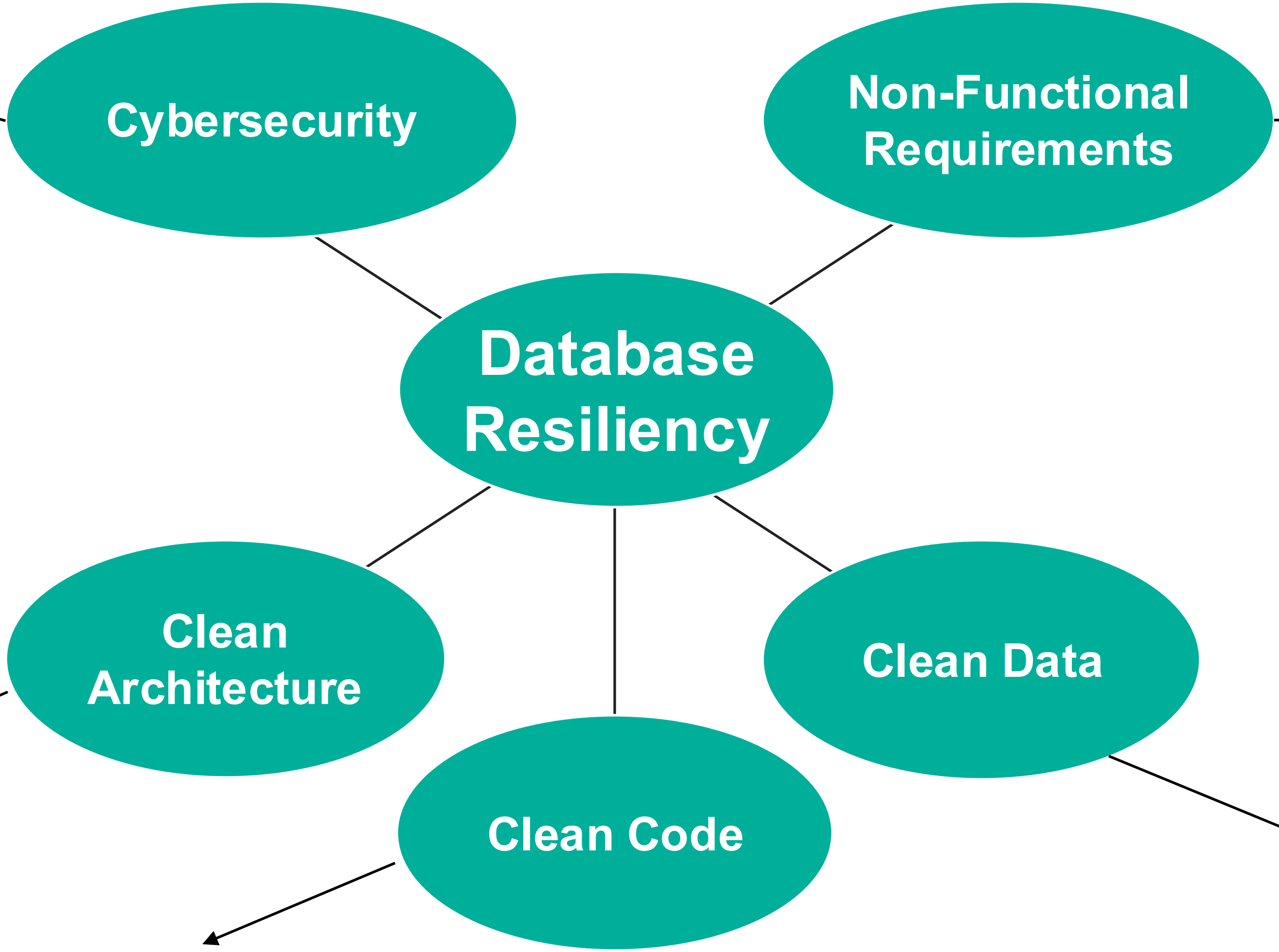


<u>Server / Database Flaws</u>

- Configurations
- Privileges
- Infrastructure
- Architecture
- Modeling / Design
- Data Integrity
  - Tables
  - Columns
  - Indexes
  - Code
- Non-Functional Requirements
  - Hardened
  - Performance
  - Portable
  - Reliable
  - Etc.

Pillars of Database Resiliency

- Access
- Auditing
- Encryption
- External Apps
- Passwords
- Server



- Auditability
- Compliance
- Interoperability
- Maintainability
- Performance
- Portability
- Reliability
- Regulations
- Scalability
- Stability

- Tables
- Columns
- Indexes

- Field Types
- Field Sizes
- Relationships

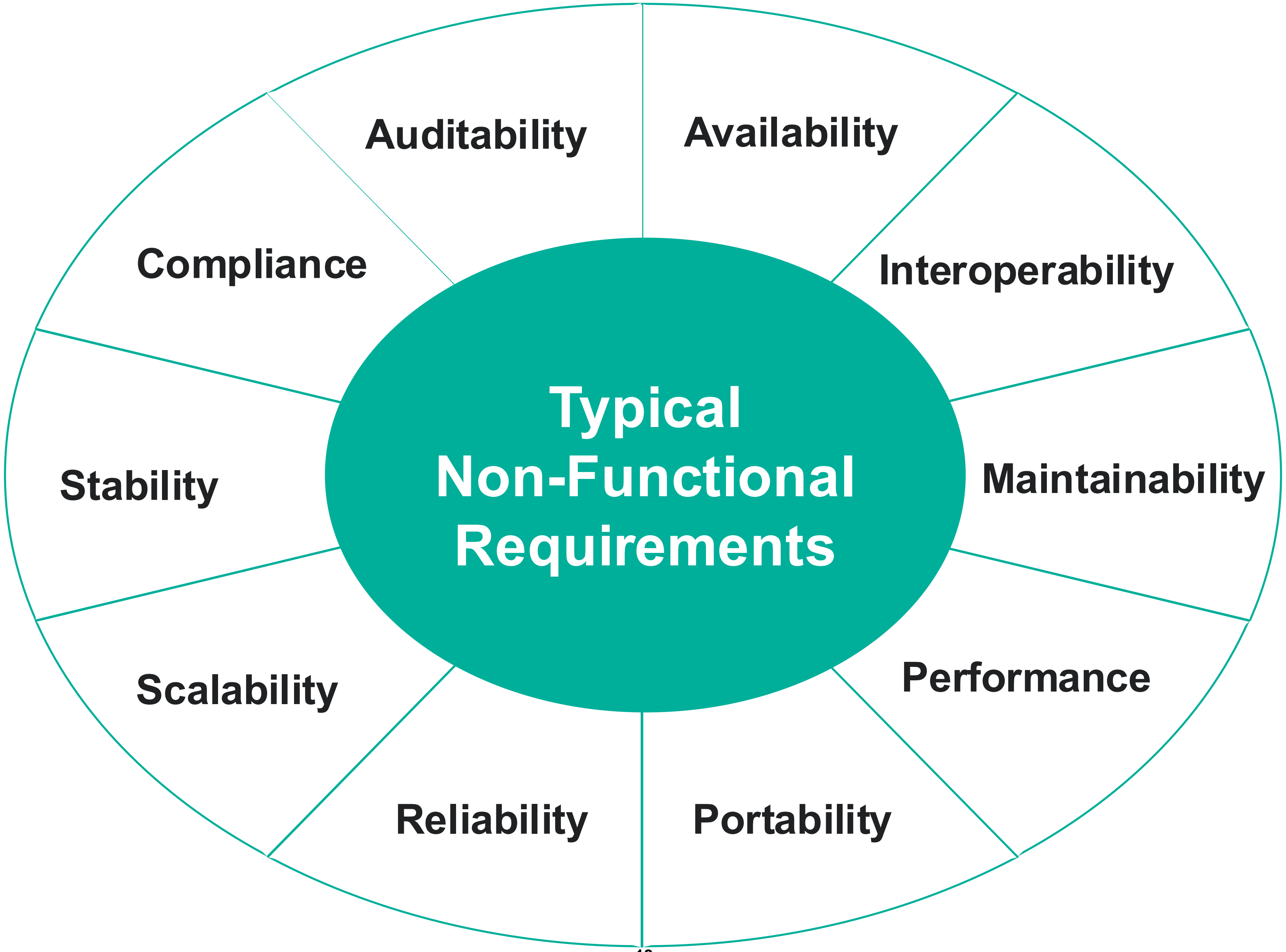
- Stored Procedures
- Triggers
- Views

Pillars of Database Resiliency: Cybersecurity

Vulnerabilities & Weaknesses

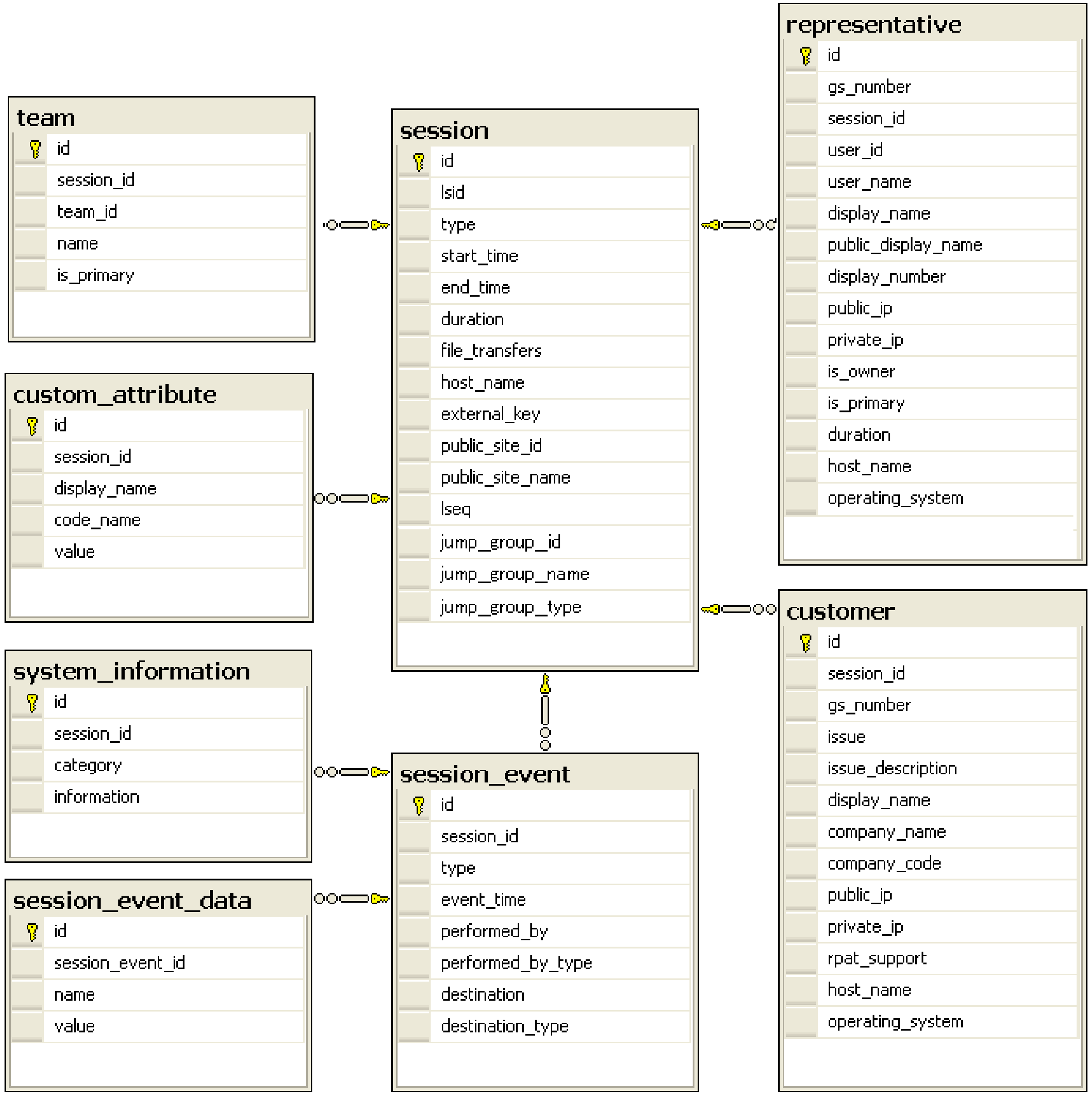
- Access
- Auditing
- Encryption
- Passwords
- Server

Pillars of Database Resiliency: Non-Functional Requirements



Pillars of Database Resiliency: Clean Architecture

- Data integrity (completeness & correctness)
- No data anomalies when updating
- Auditable deletes
- Separation of concerns
- Minimal maintenance
- Limited tables, columns, and indexes
- Appropriate normalization
- Maintain NFR thresholds
- Low ABCs



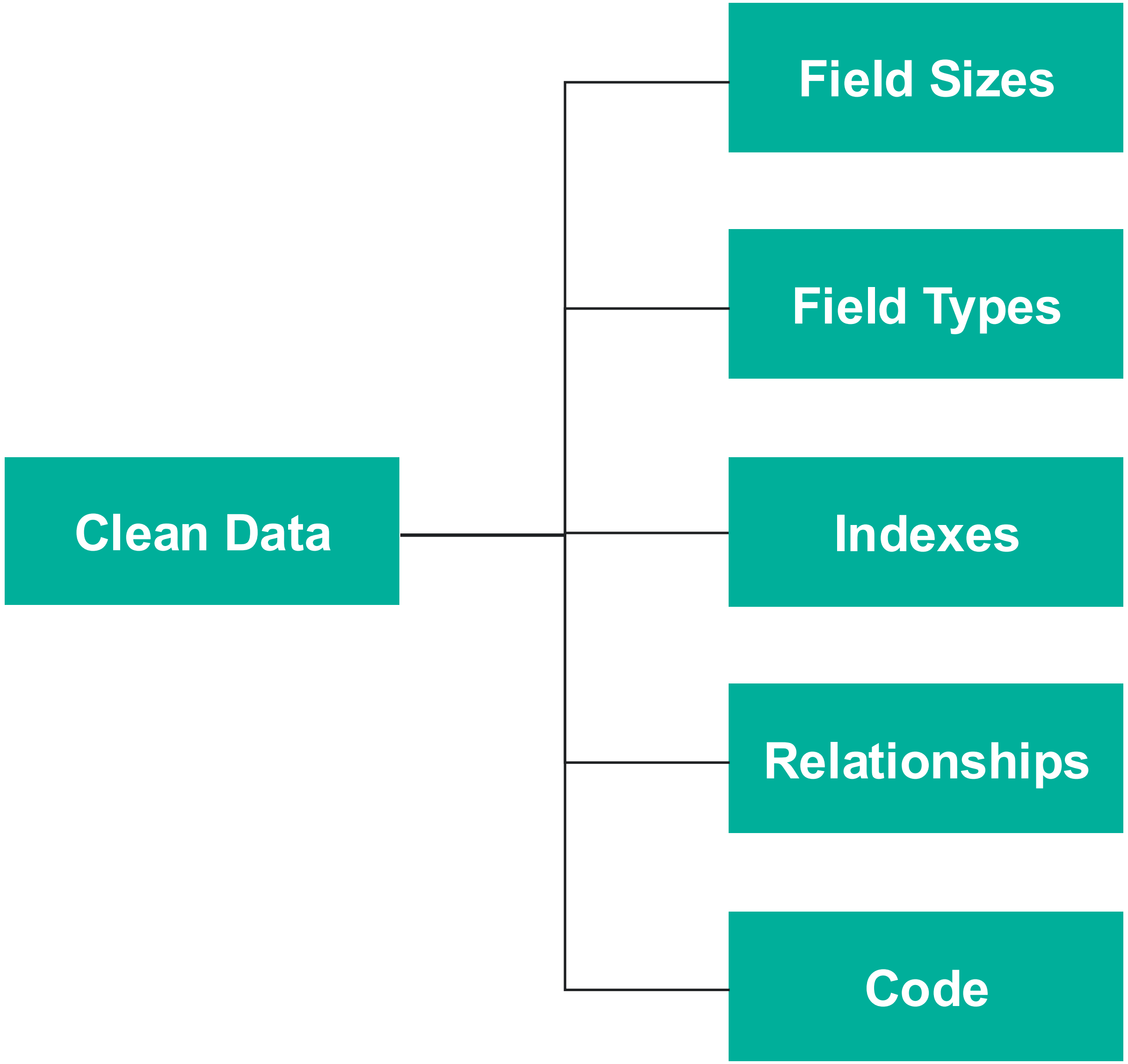
Pillars of Database Resiliency: Clean Code

Stored Procedures | **Triggers** | **Views**

Data Definition (DDL)	Data Query (DQL)	Data Manipulation (DML)	Data Control (DCL)	Transaction Control (TCL)
Alter Create Drop Rename Truncate	Select	Delete Insert Merge Update	Grant Revoke	Commit Rollback Savepoint Set Transaction

- Low ABCs
- Correctness / Completeness
- Consistency
- Readability / Comprehendible
- Maintainability
- Performance

Pillars of Database Resiliency: Clean Data



disruptiveOps

Dang Humans!

Most Observed Myths

1. “They can't get into our systems”
2. “We can secure our systems”
3. Bad guys are focused on intrusion
4. Your data folks know how to build hardened database platforms
5. You know what to look for and when
6. Our data is safe since its encrypted
7. Our testers will keep us honest
8. We have sufficient auditing
9. We can recover from our backups
10. We are doing Agile, Scrum, SAFe, etc so we are building in quality and security

What are your

Assumptions?

Causation?

- Lack of understanding what resiliency is
- Lack of knowledge on where to look for vulnerabilities & weaknesses
- Unsure what standards, controls, compliance, regulation, etc. are relevant
- We do not focus on what matters most (Organizational Continuity & Distributed Systems)
- We believe we can be secure
- Testing makes us feel good
- Thinking clouds make us safer
- We do not maintain and critique relevant artifacts
- We focus on software security while ignoring risks within database platforms

Typical Reasons for a Lack of Resiliency

- You cannot be resilient until you define what resiliency means
- “All things fail, all the time”
- “We will always get disrupted”
- “Your systems are never secure”
- “You are not hardened until you prove you are”
- Zero-Day / X-Year Threats
- Threats are everywhere (Adversarial vs Non-Adversarial)
- You are not conducting continual infrastructure / configuration analysis
- Lack of continual risk / issue management
- You do not know how to address and recover
  - Governance
  - Standards
  - Policies
  - Procedures

disruptiveOps

Shift to a Risk-based Approach

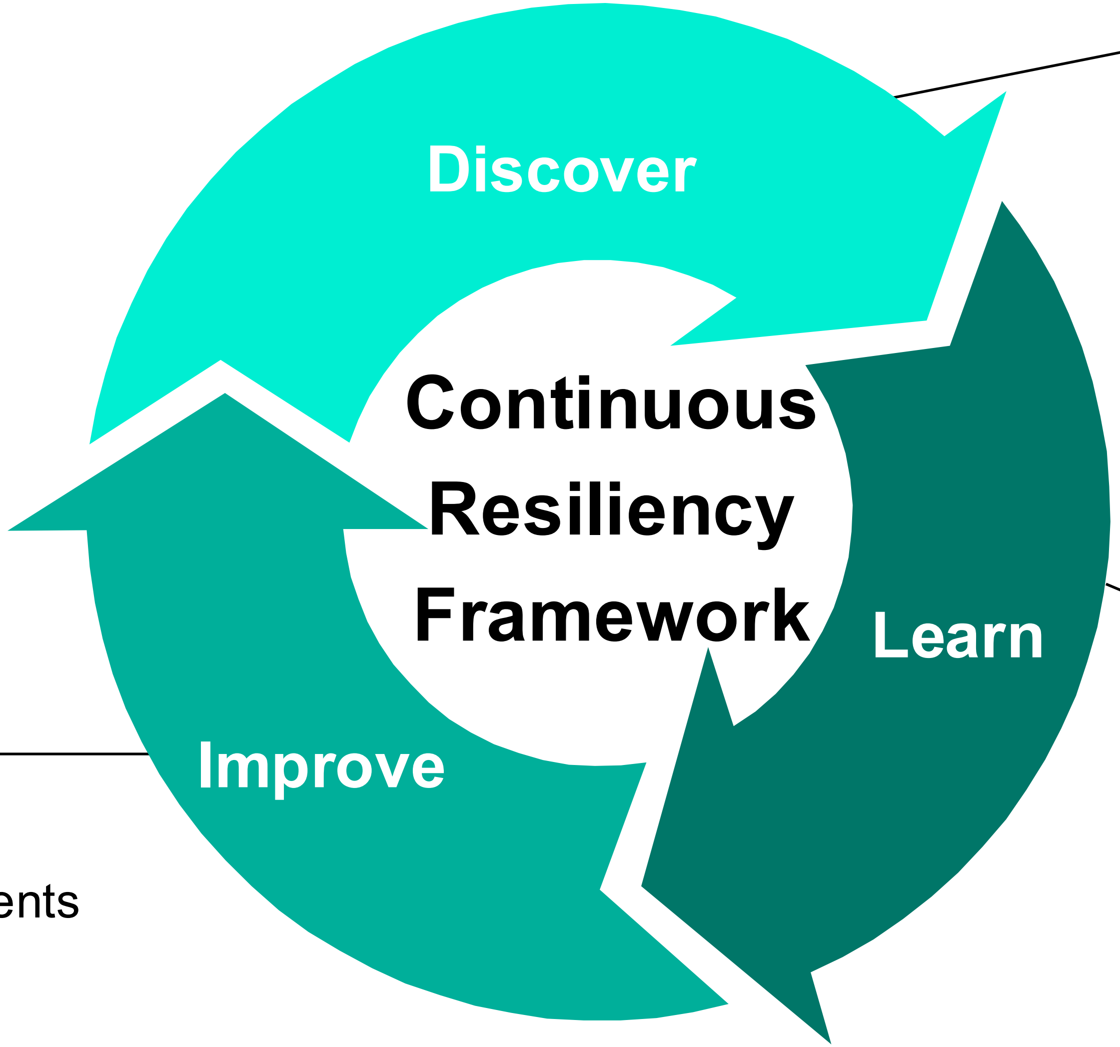
How to Improve: Your 10-Step Program

First, admit you have a problem

1. Get *More* Educated
2. Select Your *Most* At-Risk Solution
3. Define *Minimal* Resiliency
4. Establish *Minimal* Governance
5. Establish *Minimal* Standards
6. Establish *Minimal* Measures & Metrics
7. Conduct *Initial* Analysis
8. Identify *Initial* Waste, Opportunity for Improvements, and Risk
9. *Start* Fixing
10. *Start* the Continuous Resiliency Framework

Continuous Activities for Database Resiliency

- Vulnerabilities & Weaknesses
- Architecture / Design Flaws
- Risks / Issues
- Performance Concerns
- Code Concerns
- Waste
- Opportunities for Improvement



- Behaviors
- Organizational Continuity
- Architecture / Design
- Code
- Management
- Non-Functional Requirements
- Performance
- Planning
- Skills

- Likelihood and Impacts
- Adversarial / Non-Adversarial
- Suggestive Fixes
- Risk-based Plans
- Risk Resolution / Mitigation
- Opportunities for Improvement

Continually "Plan-in" Risk

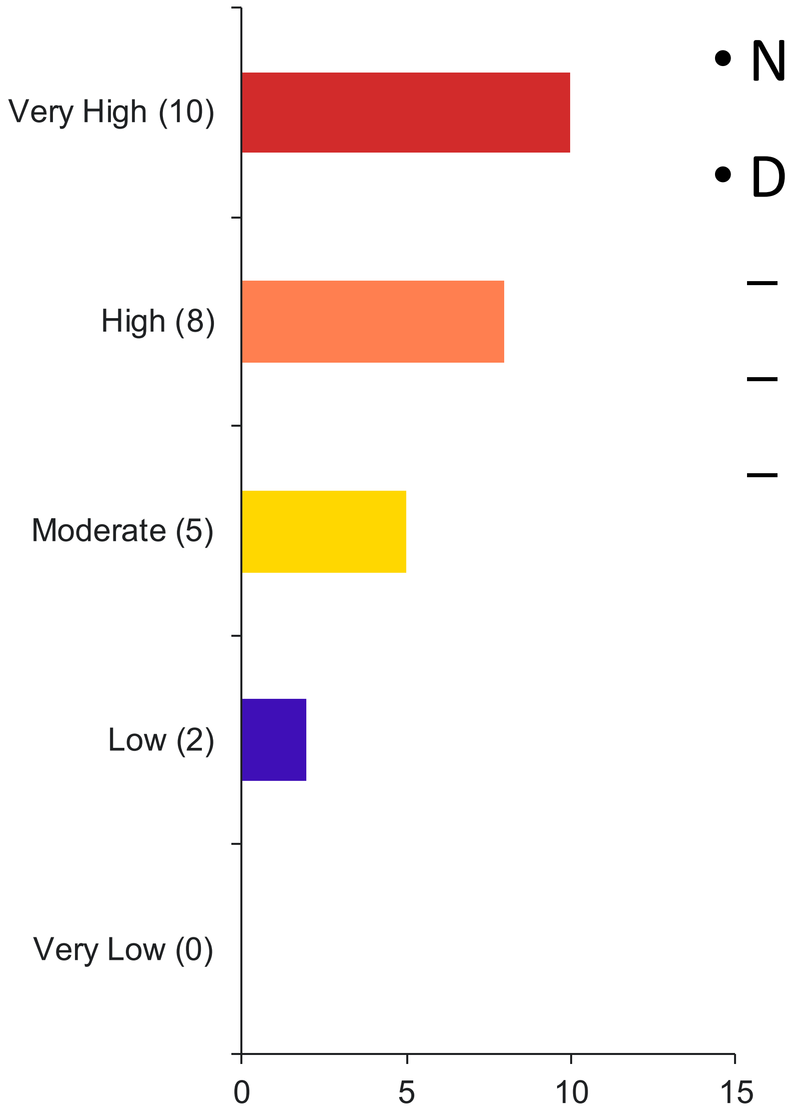
Auto-Generated

Risk Assessments
- Risks & Issues
- Likelihood & Impact
- Risk Score

Plans
- Risks & Issues
- POA&Ms
- Actionable Plan

Artifacts
- Suggestive Fixes
- Corrective Code

Standardize on a Risk Scoring Approach



- NIST-based approach
- Default Scoring
  - ***<u>Likelihood</u>*** of a finding being realized (Adversarial vs Non-Adversarial)
  - ***<u>Impact</u>*** if finding is realized
  - ***<u>Risk</u>*** to organizational continuity

Likelihood * Impact = Risk Score

*Moderate * High = High*

*5 * 8 = 40*

Strive to Maintain Community Alignment

Eliminate. Automate. Reduce.

- Operational frameworks & processes are most typical the impediment to resiliency and efficiency
- Replace (not tweak) approaches that are a bad fit
  - Scaled Agile Framework (SAFe)
  - Scrum
- Eliminate Agile coaches
- Measure everything
- Reduce team autonomy
- Stop the “safe space” mentality

Digital Continuity Framework

3 Key Elements

6 Key Activities

Contact Us

680 S Cache Street,
Suite 100-14672
Jackson, WY 83001

17 West Jefferson Street
Suite 003
Rockville, MD 20850

Info@disruptiveOps.com

www.disruptiveOps.com