$ whoami

- Founder/CEO/CTO The Scale Factory
- Working in hosting/infrastructure for 20 years

Infrastructure / AWS / DevOps
What is Well-Architected?
What is a Well-Architected Review?
Common Review Findings
WHAT IS WELL-ARCHITECTED?
WELL ARCHITECTED ORIGINS

- Catalogue of emergent good practices
- Observed by AWS Field Solutions Architects
- Codified and shared
- Platform agnostic*

@jtopper
Lenses

- Serverless Applications
- High Performance Computing
- IoT (Internet of Things)
USING WELL-ARCHITECTED

- Gap analysis / planning
- Teaching
- Team alignment

@jtopper
WHAT IS A WELL-ARCHITECTED REVIEW?
WELL ARCHITECTED REVIEW

- Foundational questions
- Up to 4 hours
- Qualitative

@jtopper
<table>
<thead>
<tr>
<th>Category</th>
<th>Operational Excellence</th>
<th>Security</th>
<th>Reliability</th>
<th>Performance Efficiency</th>
<th>Cost Optimisation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Well Architected Core</td>
<td>9</td>
<td>11</td>
<td>9</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>Serverless Applications</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>High Performance Computing</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>IoT (Internet of Things)</td>
<td>4</td>
<td>11</td>
<td>6</td>
<td>10</td>
<td>4</td>
</tr>
</tbody>
</table>

@jtopper
How do you determine what your priorities are?

- Evaluate external customer needs
- Evaluate internal customer needs
- Evaluate compliance requirements
- Evaluate threat landscape
- Evaluate tradeoffs
- Manage benefits and risks
- None of these
How do you determine what your priorities are?

- Evaluate external customer needs
- Evaluate internal customer needs
- Evaluate compliance requirements
- Evaluate threat landscape
- Evaluate tradeoffs
- Manage benefits and risks
- None of these
How do you determine what your priorities are?

- Evaluate external customer needs
- Evaluate internal customer needs
- Evaluate compliance requirements
- Evaluate threat landscape
- Evaluate tradeoffs
- Manage benefits and risks
- None of these
QUESTION
OPS 1

How do you determine what your priorities are?

- Evaluate external customer needs
- Evaluate internal customer needs
- Evaluate compliance requirements
- Evaluate threat landscape
- Evaluate tradeoffs
- Manage benefits and risks
- None of these

Medium Risk

@jtopper
QUESTION
OPS 1

How do you determine what your priorities are?

- Evaluate external customer needs
- Evaluate internal customer needs
- Evaluate compliance requirements
- Evaluate threat landscape
- Evaluate tradeoffs
- Manage benefits and risks
- None of these
How do you determine what your priorities are?

- Evaluate external customer needs [✓] WA
- Evaluate internal customer needs [✓] WA
- Evaluate compliance requirements [✓] WA
- Evaluate threat landscape [✓] NI
- Evaluate tradeoffs [✓] NI
- Manage benefits and risks [✓] NI
- None of these [ ] CI
COMMON REVIEW FINDINGS
How do you determine what your priorities are?

- Evaluate external customer needs
- Evaluate internal customer needs
- Evaluate compliance requirements
- Evaluate threat landscape
- Evaluate tradeoffs
- Manage benefits and risks
- None of these
How do you select your storage solution?

- Understand storage characteristics and requirements
- Evaluate available configuration options
- Make decisions based on access patterns and metrics
- None of these
How do you implement change?

- Deploy changes in a planned manner
- Deploy changes with automation
- None of these
THE BAD

@jtopper
How do you plan for disaster recovery?

- Define recovery objectives for downtime and data loss
- Use defined recovery strategies to meet the recovery objectives
- Test disaster recovery implementation to validate the implementation
- Manage configuration drift on all changes
- Automate recovery
- None of these

<table>
<thead>
<tr>
<th>Option</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>WA</td>
<td>33%</td>
</tr>
<tr>
<td>WA</td>
<td>33%</td>
</tr>
<tr>
<td>WA</td>
<td>25%</td>
</tr>
<tr>
<td>NI</td>
<td>39%</td>
</tr>
<tr>
<td>NI</td>
<td>16%</td>
</tr>
<tr>
<td>CI</td>
<td>31%</td>
</tr>
</tbody>
</table>

@jtopper
How do you respond to a [security] incident?

- Identify key personnel and external resources
- Identify tooling
- Develop incident response plans
- Automate containment capability
- Identify forensic capabilities
- Pre-provision access
- Pre-deploy tools
- Run game days
- None of these

High Risk
75%
[93%]
HRI Rank: 2

@jtopper
How do you classify your data?

- Define data classification requirements
- Define data protection controls
- Implement data identification
- Automate identification and classification
- Identify the types of data
- None of these

High Risk
75% (88%)
HRI Rank: 3

WA 61%
WA 39%
WA 17%
NI 4%
NI 59%
CI 23%
How do you evaluate new services?

- Establish a cost optimisation function
- Develop a workload review process
- Review and implement services in an unplanned way
- Review and analyse this workload regularly
- Keep up to date with new service releases
- None of these
How do you test resilience?

- Use playbooks for unanticipated failures
- Conduct root cause analysis and share results
- Inject failures to test resiliency
- Conduct game days regularly
- None of these
How do you reduce defects, ease remediation, and improve flow into production?

- Use version control
- Test and validate changes
- Use config management systems
- Use build/deploy systems
- Perform patch management
- Share design standards
- Implement practices to improve code quality
- Use multiple environments
- Make frequent, small, reversible changes
- Fully automate integration and deployment
- None of these

Well Architected: 14%  
@jtopper
How do you understand the health of your workload?

- Identify key performance indicators (WA 53%)
- Define workload metrics (WA 62%)
- Collect and analyse workload metrics (WA 72%)
- Establish workload metric baselines (NI 51%)
- Learn expected patterns of activity for workload (NI 54%)
- Alert when workload outcomes are at risk (NI 40%)
- Alert when workload anomalies are detected (NI 34%)
- Validate the achievement of outcomes and the effectiveness of KPIs and metrics (NI 37%)
- None of these (CI 14%)
How do you control human access?

- Define human access requirements
- Grant least privileges
- Allocate unique credentials per person
- Manage credentials based on lifecycle
- Automate credential management
- Grant access through roles or federation
- None of these
How do you control programmatic access?

- Define programmatic access requirements
- Grant least privileges
- Automate credential management
- Allocate unique credentials per component
- Grant access through roles or federation
- Implement dynamic authentication
- None of these

High Risk: 57% (89%)
HRI Rank: 15
MAJOR THEMES

@jtopper
TEAMS ARE OK AT CHOOSING CORRECT SERVICES

- Database choices match workload
- Storage choices match workload
- Compute choices sometimes not right-sized.
Teams are OK at making software changes.

- Automation tools are being used
- Full CD remains out of reach
- Change batch sizes need to be smaller

@jtopper
<table>
<thead>
<tr>
<th>Aspect of Software Delivery Performance*</th>
<th>Elite</th>
<th>High</th>
<th>Medium</th>
<th>Low</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Deployment frequency</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>For the primary application or service you work on, how often does your organization deploy code to production or release it to end users?</td>
<td>On-demand (multiple deploys per day)</td>
<td>Between once per day and once per week</td>
<td>Between once per week and once per month</td>
<td>Between once per month and once every six months</td>
</tr>
<tr>
<td><strong>Lead time for changes</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>For the primary application or service you work on, what is your lead time for changes (i.e., how long does it take to go from code committed to code successfully running in production)?</td>
<td>Less than one day</td>
<td>Between one day and one week</td>
<td>Between one week and one month</td>
<td>Between one month and six months</td>
</tr>
<tr>
<td><strong>Time to restore service</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>For the primary application or service you work on, how long does it generally take to restore service when a service incident or a defect that impacts users occurs (e.g., unplanned outage or service impairment)?</td>
<td>Less than one hour</td>
<td>Less than one day</td>
<td>Less than one day</td>
<td>Between one week and one month</td>
</tr>
<tr>
<td><strong>Change failure rate</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>For the primary application or service you work on, what percentage of changes to production or released to users result in degraded service (e.g., lead to service impairment or service outage) and subsequently require remediation (e.g., require a hotfix, rollback, fix forward, patch)?</td>
<td>0-15%</td>
<td>0-15%</td>
<td>0-15%</td>
<td>46-60%</td>
</tr>
</tbody>
</table>

TEAM ARE BAD AT THINKING ABOUT FAILURE MODES

- Not considering business requirements
- No risk analysis of failure modes
- Poor documentation
- Almost no attempt to rehearse outages

@jtopper
<table>
<thead>
<tr>
<th>Reference</th>
<th>Component</th>
<th>Risk</th>
<th>Likelihood</th>
<th>Impact</th>
<th>Observation (Cold/ Impleneted)</th>
<th>Mitigation</th>
<th>Runback action</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 R01</td>
<td>AWS account</td>
<td>Malicious use (e.g. crypto-mining) using AWS resources up to account limit</td>
<td>Low = Medium</td>
<td>Use GuardDuty alerts (eg with Stack integration) to detect suspicious misuse. Consider subscribing to AWS Security Hub. Use Sentry to detect application failures.</td>
<td>- Follow recommended practices for AWS account security</td>
<td>- Automate application deployment</td>
<td>Address breach</td>
<td>GuardDuty on, but not Tansformed</td>
</tr>
<tr>
<td>4 R03</td>
<td>API Lambda (Django / Zappp)</td>
<td>Cold start delay on scale-out event</td>
<td>High = High</td>
<td>Use CloudWatch / X-Ray metrics</td>
<td>- Ensure good retry/backoff logic in front-end (code changes)</td>
<td>- Adjust warning event frequency</td>
<td>Cards:</td>
<td></td>
</tr>
<tr>
<td>5 R04</td>
<td>API Lambda (Django / Zappp)</td>
<td>Cold start delay after idle</td>
<td>Medium = Medium</td>
<td>Use CloudWatch / X-Ray metrics</td>
<td>- Ensure good retry/backoff logic in front-end (code changes)</td>
<td>- Move application components into Fargate</td>
<td>Cards:</td>
<td></td>
</tr>
<tr>
<td>6 R05</td>
<td>API Lambda (Django / Zappp)</td>
<td>Lambda concurrency limit reached through load</td>
<td>High = High</td>
<td>Use CloudWatch metrics to monitor and alarm on Lambda concurrency.</td>
<td>- Reserve Lambda execution for Django API lambda</td>
<td>- Request increased account-wide Lambda execution limit</td>
<td>Cards:</td>
<td></td>
</tr>
<tr>
<td>7 R06</td>
<td>API Lambda (Django / Zappp)</td>
<td>Denial-of-service attack via backend API gateway, exceeding account Lambda limit</td>
<td>Low = High</td>
<td>Use CloudWatch metrics to monitor and alarm on Lambda concurrency. Use Sentry to detect failed calls to external APIs. Use CloudWatch metrics to monitor and alarm on Lambda concurrency.</td>
<td>- Reduce Django Lambda execution time (code changes)</td>
<td>Cards:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 R07</td>
<td>API Lambda (Django / Zappp)</td>
<td>API misuse, eg another party wishing to access past APIs using a gateway</td>
<td>Low = High</td>
<td>Use CloudWatch metrics to monitor and alarm on Lambda concurrency.</td>
<td>- Configure AWS WAF for CloudFront distribution</td>
<td>Cards:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9 R08</td>
<td>API Lambda (Django / Zappp)</td>
<td>IP address exhaustion (Lambda subnets)</td>
<td>Low = Medium</td>
<td>Use CloudWatch metrics to monitor and alarm on Lambda concurrency, which is a proxy for IP address use.</td>
<td>- Redesign VPCs</td>
<td>Cards:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 R09</td>
<td>API Lambda (Django / Zappp)</td>
<td>ENI exhaustion</td>
<td>Low = Medium</td>
<td>ENI exhaustion will manifest as (unexplained) Lambda call failures.</td>
<td>- Request increased ENI limit for account</td>
<td>Cards:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11 R10</td>
<td>SDR Lambda</td>
<td>Manual deployment error</td>
<td>Medium = Medium</td>
<td>Use Sentry to detect application failures. Monitor ENI use, publish a custom CloudWatch metric (based on querying the EC2 API). Optionally, set alarms.</td>
<td>- Automate application deployment</td>
<td>Cards:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12 R11</td>
<td>SDR Lambda</td>
<td>Lambda concurrency limit reached through load</td>
<td>Medium = High</td>
<td>Use CloudWatch metrics to monitor and alarm on Lambda concurrency.</td>
<td>- Reduce Django Lambda execution time (code changes)</td>
<td>Requests increased account-wide Lambda execution limit</td>
<td>Cards:</td>
<td></td>
</tr>
<tr>
<td>13 R12</td>
<td>SDR Lambda</td>
<td>Cold start delay on scale-out event</td>
<td>High = High</td>
<td>Use CloudWatch / X-Ray metrics</td>
<td>- Warm front end Lambda</td>
<td>Cards:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14 R13</td>
<td>SDR Lambda</td>
<td>Cold start delay after idle</td>
<td>High = Medium</td>
<td>Use CloudWatch / X-Ray metrics</td>
<td>- Warm front end Lambda</td>
<td>Cards:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15 R14</td>
<td>AWS SES</td>
<td>Sending quota exceeded</td>
<td>Low = High</td>
<td>Use Sentry to detect failed message send events</td>
<td></td>
<td>Cards:</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

@jtopper
Teams are bad at monitoring for failure modes.

- Monitoring happening
- Data not used for much
- Tracing almost non-existent

@jtopper
TEAMS NEED TO DO BETTER AT SECURITY

- Poor hygiene around patching
- Limited data classification
- Mediocre human access control
- Bad programmatic access control
- Low adoption of security monitoring tools
TOP BREACH CAUSES

- Using components with known vulnerabilities
- Security misconfiguration
- Injection
- Weak auth / session management
- Missing function access control

@jtopper

https://snyk.io/blog/owasp-top-10-breaches/
EVERYONE IS BETTER AT BUILDING PLATFORMS THAN THEY ARE AT SECURING OR RUNNING THEM.
WHAT NEXT?

- Read the white papers:
  https://aws.amazon.com/architecture/well-architected/

- Run your own review(s)
  https://aws.amazon.com/well-architected-tool/

- Consider engaging an AWS Well-Architected partner
  https://scalefactory.com/services/well-architected/
  (funding available)
http://www.scalefactory.com/
https://github.com/scalefactory
@scalefactory
jon@scalefactory.com