CHAPTER 7

Tearing Down and Retraining Muscle Memory

by Tracy Bannon

CHAPTER 7

Tearing Down and Retraining Muscle Memory

Tracy Bannon

Raging Against Muscle Memory!

If you Google “muscle memory”, .56 seconds later you will get a listing of “about 63,900,000” entries. Scrolling through just the first page gives two distinctive opinions. The first is muscle memory is important to reduce decision-making time; the second is it takes at least 3-6 months to break muscle memory.

What does this have to do with changing the way we approach cybersecurity? Muscle memory exists in every organization and has the potential to help or greatly hurt a team’s effectiveness. Let’s start by defining muscle memory: **“the ability to repeat a specific muscular movement with improved efficiency and accuracy that is acquired through practice and repetition”**[1]

Living with a soccer coach means I’m exposed to training and athletic speak every day.

My role with any client, sponsor, or organization generally lasts less than 3 years. I am not a CISO, rather I am a chief architect and software engineering leader keenly aware of the foundational role of cyber security and resiliency. With each new client, my first step is situational awareness and

1 Merriam-Webster. Muscle memory. merriam-webster.com/dictionary/muscle%20memory

"You've gotta build up an automatic and unconscious response"
Autopilot
Tap into that "forgotten" skill
"You react as you train"
"Practice doesn't make perfect, it makes things permanent"
Instantly, naturally

understanding the capabilities, instincts, and relationships of the group or situation. It is similar to starting a new high school soccer season and meeting this year's players. The time starts with intros, a few stories, and banter to get a feel for personalities and understand if anyone has played together before, then break out into a few small-sided games. The coach is looking for muscle memory and if it needs to be retrained.

Trust as a Foundation

I consistently find the biggest obstacle to addressing cyber resilience is not technical, but rather a lack of trust. While there are many technical challenges, organizations where a pervasive sense of trust exists fare much better.

My career as a software architect and engineer requires that I am keenly aware of the security needs and requirements of my clients. I have spent years focused on solving complex government challenges with software that involves private and sensitive information about taxpayers, the health and wellbeing of children enrolled in enabling government programs, and data that help our warfighters protect us as a nation.

While there are many policies and stakeholder-related security requirements, the pattern I repeatedly encounter is that cyber security is carved off as a separate team. As the proverbial protectors of the castle, there exists a natural elitism.

The First Day of Practice

A few years ago, I was asked to take on the role of enterprise architect for one of the largest state-level agencies in the US. The architecture leadership team was already in place, including 10 architects with varying specialties: information architecture, database architecture, enterprise services architecture, and application architecture. They were all steeped deeply in domain knowledge for their “product line”: Child Welfare, Integrated Eligibility, Child Support, and so on.

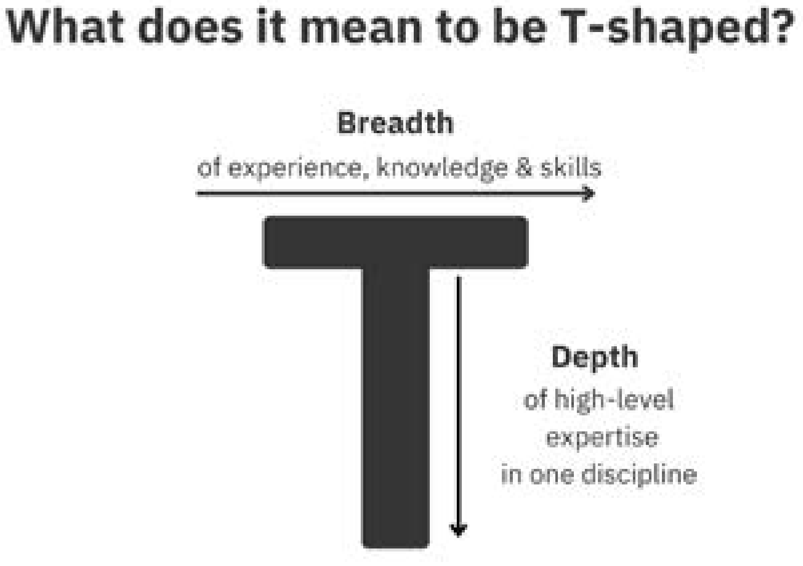
On my first day, I took a page from my husband’s coaching approach and started with intros, stories, and small-side games. I brought together the architects and the senior engineers to get to know them, their superpowers and to understand their products and lines of business.

From a title and specialty perspective, it was quite a diverse group gathered around the conference room: data and database architect, infrastructure architect, network engineer, middleware engineering lead, application architect, performance testing leader, enterprise services operations, and even document management subsystem engineer.

Missing from the table was anyone focused on security. Maybe some of these guys are “T-shaped” I wondered? The gaggle of tech pros discussed their roles and how their teams operated. As the conversation progressed, it became obvious that the organization was comprised of rugged individualists with their own approaches and language. I was stepping in to lead a colony of cowboys.

“T-shaped” means having a mix of breadth and depth allowing for more horizontal collaboration and sharing than traditional “depth only” specialists.[2]

2 Howard, C., & Chansler, G. (2015, February 9). Fitting to the T. IT Briefcase. itbriefcase.net/fitting-to-the-t


What does it mean to be T-shaped?
Breadth
of experience, knowledge & skills
Depth
of high-level expertise in one discipline

I felt like I was watching an international soccer team made of individual star players. These stars could occasionally pass to one another, but usually, they were concerned about moving the ball to goal themselves.

Observations: Skills, Alliances, and Elitism

The credentials of my team were impressive and included academic degrees, certifications from industry, corporate awards, and time served. There is some merit to understanding the journey of the individuals, though ultimately a team is defined as an interdependent group of individuals working together to achieve a goal.[3] In getting to know the individuals, there were definitely excellent individual skills.

I could spot the alliances right away, guys who had shared war stories from over their many, many years together. Their familiarity with one another could masquerade as trust. This group did not operate as a unified team; they operated as a loose federation of individuals lacking a common goal and shared mission.

3 Thompson, L. L. (2007). Making the Team: A Guide for Managers (3rd ed.). Pearson..

Missing from the room was cybersecurity. Over the years, the cybersecurity group became more and more insulated and isolated. Their funding and their corporate sponsorship had taken a different approach by funding constant training and certifications. They met as a group, wrote whitepapers, and published mandatory edicts and standards. In essence, they had their own private-club jerseys.

Preseason Practice Plan

Leading means mentoring and coaching. I sat down for dinner with my husband (who is an Ops guy by day) to draft out an initial practice plan. Any coach will tell you that you hope for and envision an undefeated season, and you start with the foundational building blocks. My team did not have all the skills needed. The security organization needed to be at the table; security needed to **voluntarily join the team**. Our organization needed to have a **shared mission**. That mission needed to be defined using a **common language** to help set shared expectations. These rugged individuals needed to have a sense that we were part of the same team. We needed to build trust with one another and with me.

Extending an Invitation

The distancing and regimented separation of the cybersecurity pros has always rubbed me the wrong way. Why? Am I anti-specialization? No; I am totally in favor of individuals choosing to dive deep and be the resident expert. Do I want exclusive ownership? No; I want reasonable shared responsibility. You see, I'm not a typical architect and I don't fit the profile of a typical software engineer.

I have always sought shared knowledge, shared responsibility, and having complete visibility and understanding of all aspects of any product, solution, or digital platform that we design, deliver and operate. Add in that I thrive on building a sense of community, and you have a sense of why my approach to cyber has raised eyebrows and caused turf wars.

I spent time with the leaders and mid-level pros from the security team and was able to make the case for them to "help the architects and the delivery teams". While it was true that help was needed from the security team, the time was not right to make formal changes to add security directly to the architecture group... at least not yet. Security would be in the room as we recalibrated.

The Shared Mission

It's ironic that we all make fun of mottos and mission statements that became trendy in the late 1980s. However, the techniques for crafting mission statements sow the seeds for buy-in, inspiration, and a common language.

Taking into consideration this government agency's imperative and legal charter, I rallied the team together to define a plausible and attainable mission statement aligned to the agency's mission. This was not without eye-rolls and murmurs of "more touchy-feely crap." Keeping focus on the task, we spend a morning together defining our purpose, our values, and our goals. Many of the individuals I've met during my career have been drawn to work with the government because of their values and the core purpose of the agency. The hours can be long and the pay does not always keep pace with our cousins in the commercial space.

As we started to openly discuss why we were drawn to that agency and that team, some core elements and emotions surfaced. This agency has the responsibility to find and protect abused children in need of effective child protective services to prevent any further suffering. It was also immersed in providing rehabilitative services to children and parents to restore the integrity of family life.

That is what we were called to do! Each of us had a shared responsibility to align business agility with technical agility to protect children and restore families. Can you think of another domain with such a need to secure data and have cyber-resilient solutions supporting the agency?

A Common Language

After a few more days of quiet and inquisitive observation, it was time to pull the team together and establish our definitions and language. I often start by collaboratively laying out a common set of definitions, the rationale, and the impacts. The first step is checking if the organization has existing definitions or if there are corporate policy or statutory regulations that define some terms and procedures. Often, policy or corporate definitions are focused on siloed assignment of responsibility. This reflects the slow evolution from waterfall to agile, or worse, reflects the organizational structure.

This phenomenon of codifying an organization's structure in the processes and the resulting software is often called Conway's law, taken from Melvin Conway's description written in 1968: "Any organization that designs a system (defined broadly) will produce a design whose structure is a copy of the organization's communication structure."[4]

With markers and post-it notes in hand, we began to map out the existing structure and called out the historic context for how it had evolved. Coupled with our wonderful shared mission, the initial posturing began to break down. We laid out clear and succinct definitions of agile laying flat that there was no one on the team that defined it the same way. We did not prescribe the brand of agile or any named methodology. The focus was on core values of the agile manifesto.[5] It was not without heated debate that continued to highlight the depths of the lack of trust. It did, however, provide honest dialog and considerations for valuing the left more, but not ignoring the items on the right.

More Value		**Less Value**
Individuals and interactions	over	Processes and tools
Working software	over	Comprehensive documentation
Customer collaboration	over	Contract negotiation
Responding to change	over	Following a plan

DevSecOps Fireworks

The next term to address was DevSecOps; this is where the fireworks began and justifiably so. DevOps is not simply "Development + Operations", nor is it simply "Development + Security + Operations". These terms carry across people, processes, technology, and culture. They are foundational

4 Conway, M. E. (1968, April). How Do Committees Invent? Datamation. melconway.com/Home/Committees_Paper.html

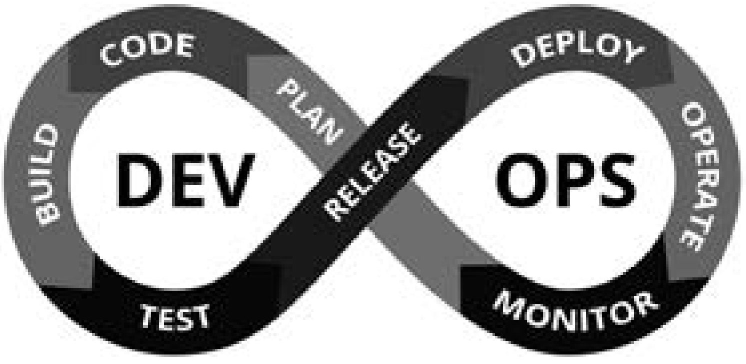
5 Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Martin, R. C., Mellor, S., Thomas, D., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Schwaber, K., & Sutherland, J. (2021, August 10). Manifesto for Agile Software Development. Agile Alliance. agilealliance.org/agile101/the-agile-manifesto/

to emerging digital transformation for software and software-intensive offerings.

Most of the team had wildly different definitions and very limited experience with DevSecOps. Generally, their experience was on applying some time-saving automation to their slice of the software development lifecycle. As an organization they had struggled to implement what you could call a pipeline, falling back to the muscle memory of deployment checklists, scans, and putting the quality assurance (aka testing team) in the cross-hairs for any defects.

I put on my architect's hat to drive and started by exposing the current state (the baseline), then working on a reasonable target vision. I also asked everyone to take 30 minutes before lunch to sketch, draw, or copy a DevSecOps image that spoke to them personally. We would share these over a working lunch as inputs to crafting a succinct shared definition.

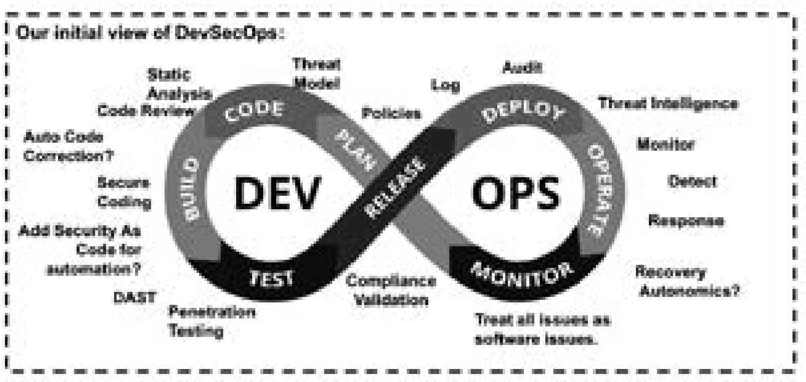
Driving ownership of the definition means we did not adopt any one suggestion. We started as simply as possible with the "ole infinity loop" drawn on the whiteboard. Together we discussed how all the humans in the process should think about and react to each of the chevrons. We also needed to create our own shared understanding of where and how security fits into modern software engineering.



Basic DevOps infinity loop as a starter template[6] [7]

6 DevOps. Office of the Chief Software Officer, U.S Air Force. software.af.mil/training/devops/.

7 Nichols, B. (2021, March 29). The Current State of DevSecOps Metrics. Carnegie Mellon University SEI. insights.sei.cmu.edu/blog/the-current-state-of-devsecops-metrics/



A sketch of our initial ideas for adding security to DevOps concepts

Who Is Responsible for Security?

The tension and tug-of-war opinions during our discussions on security highlighted that there was universal concern about security. This was a +1 situation, everyone is concerned about security. I breathed a sigh of relief. The discussion also highlighted there are differing opinions on who has responsibility for a holistic cyber security and resiliency culture.

Who is responsible for security? My immediate gut instinct is to shout out “EVERYONE!”, though I don’t actually say anything to start. If I had immediately asserted this, it would have been to the dismay of some on the security team, the developers, the product owners, and those living in “the cyber castle”. (The cyber castle is the ivory tower metaphor that is heavy in theory and light in practical application and action.)

The assertion that cyber is shared responsibility is becoming a more familiar response across the industry with the rise of DevOps concepts and its doppelgänger, DevSecOps. These approaches bring with them excellent patterns and practices that need substantial rigor and learning. At the heart of DevSecOps is constant feedback and the ability to respond quickly to discoveries, situations, requested or necessary changes. This type of rapid response requires a team that shares the same mission, can see risk and take bold action, speaks the same language, and inherently trusts one another. (Sounds like an emergency response team or perhaps a soccer team, doesn't it?)

Ultimately I believe security is a child of quality and it is a quality attribute. You may be familiar with the calling quality attributes the "-ilities". The origins of this definition are my training with Carnegie Mellon's Software Engineering Institute (SEI).[8] From that perspective, I've found the definition of quality attributes and ruthlessly prioritizing them another mechanism to drive cyber wholeness, regardless of the type of software or methodology being used. Quality includes both the features that delight and engage the end-users, as well as keeping the users, the enterprise, and all data safe.

As part of our ongoing discussions and ideation, it's important to gain agreement on this concept as well. Can you claim to have a quality service that meets mission objectives and user expectations without a holistic cyber mindset? Clearly not, and we needed to build those collective relationships and opinions so we could move from ideas to actions.

Why Are You Spending Precious Hours Talking?

This was a real question from the overall executive leadership. The expectation was that as a new leader I would *impose* my wisdom in the form of practices and processes, based on successful efforts with other similar organizations. The realization was that when we had met and discussed my joining their organization, they had only heard the sentences that they wanted to hear, especially the CIO.

So it was time for a frank, transparent discussion. This conversation could not be siloed with only the CIO, it required bringing the executive leadership team together, along with their senior staff and trusted advisors.

8 Reasoning About Software Quality Attributes.
Carnegie Mellon University SEI. resources.sei.cmu.edu/library/asset-view.cfm?assetid=513803.

Walking a fine line to keep the trust and integrity of the team's debates, and sharing my own observations, I laid out the four transformational elements necessary to any sustainable success:

1. Shared vision
2. Blended team
3. Common language
4. Trust

About 10 of us met as I answered questions, gave insights, and laid out specific actions and activities to address those four transformational elements. How would success be measured? It is a combination of qualitative and quantitative data. Qualitatively, meet with the individual team members, ask questions, and listen. What are their pain points, opinions or worries? What is exciting about the growing consensus and team building?

Using anonymous surveys can be helpful if the intended audience is not too small, implying a lack of privacy and ability to speak openly. It is a relatively fine line. Quantitatively, we can also look at indicators such as hours worked, defects by type by environment, scan results, and number of hand-offs in a particular value stream.

A new technique that I intend to begin leveraging is called measuring HumanDebt. Duena Blodstrom, CEO of People Not Tech, has a series of measurable and detectable indicators on psychological safety, a cornerstone to building a culture of courage based on honesty and trust.[9]

Another Day, Another Definition: Cyber Resiliency & Cyber Security

Our team's shared mission and objectives required a few more definitions and rally points for continued growth. Shifting from traditional security-team scans of a final project before go-live, to having a cyber-centric culture, required some definition finessing.

Expanding our lexicon to include cyber resilience was a cornerstone definition. Cyber resilience is not an alternative to cyber security, but rather

9 Blomstrom, D. (2022, January 27). The HUMANDEBT™ . People Not Tech. peoplenottech.com/articles/the-humandebt/

an expanding universe of principles and capabilities tied into and aligned with modern software engineering and DevSecOps.[10] Cyber resiliency is the concept of ensuring operational and business/mission continuity. For my team, and with the support of the executives, a bit more whiteboard time was warranted to make an investment in discussions that drove us to action.

Using inputs from industry experts as a launching spot, we agreed that **Cyber Security** cannot mean impenetrable defenses. The reality is that breaches and attacks will happen. The enemy is ever-evolving using new techniques and exposing more and more attack vectors and surfaces than ever. From script kiddies to organized combatant nations, we need to expand the definitions. **Cyber Resilience** is our ability to prevent, respond, and recover.

The team rallied and agreed we needed to prepare, respond and recover from cyberattacks and data breaches while continuing to operate effectively. Like every topic, every technology, and every new idea, it was going to be too easy to get overwhelmed with the universe of guidance from CISA's Cyber Resilience Review (CRR) assessment (developed by the Department of Homeland Security) to CERT services (from Carnegie Mellon's Software Engineering Institute).[11] We opted that we would get smart together using NIST Cybersecurity Framework quick start.[12] This discussion proved to be powerful and haunting, exposing a rawness: wildly mixed opinions on who actually had "security smarts."?

Security Smarts

A small surface crack started to show in our newly formed team. Folks who hailed from the legacy security team lacked street credentials as developers and engineers, yet pushed out secure coding standards. In the past, they

10 Alkove, J. (2021, November 3). Cyber security is no longer enough: businesses need cyber resilience. World Economic Forum. weforum.org/agenda/2021/11/why-move-cyber-security-to-cyber-resilience/

11 Assessments: Cyber Resilience Review (CRR). CISA. www.cisa.gov/uscert/resources/assessments

12 NIST. (2016, May 24). Getting Started with the NIST Cybersecurity Framework: A Quick Start Guide. CSRC. csrc.nist.gov/Projects/cybersecurity-framework/nist-cybersecurity-framework-a-quick-start-guide

had provided lengthy word documents on a SharePoint site. Occasionally, a proposed-code review checklist template was suggested.

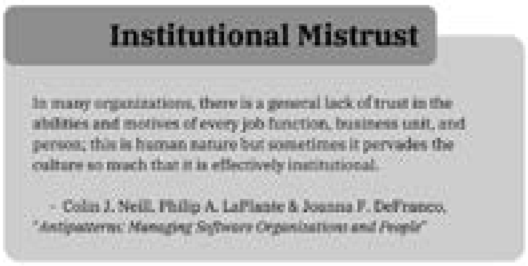
This came directly from the land before Agile where silos dominated the landscape. The developers and engineers were the recipients of heavy-handed guide books that read like legal policy without reference implementations (that is, production quality samples) or automated tests.

The architects and engineers who grew up as hands-on developers focused on the requisite software engineering needed to meet the end-user needs (provided via requirements and focus groups). They also addressed some of the cross-cutting concerns such as performance and maintainability. (Long ago, these were called "non-functional" requirements but I prefer the definition used earlier, that is, software quality attributes or the"ilities".

This lack of respect and understanding of the shared nature of security manifested in a battle that I needed to take on and shed some blood for; provide security tool access for developers and delivery teams. To get to that point, I'd have to double down on building more trust.

Institutional Mistrust: An AntiPatterns

I often turn to a book called, "Antipatterns: Managing Software Organizations and People".[13] The structure and easy consumption appeal to my inner software architect and my love of design patterns. My book is dog-eared, with myriad highlights. One of the first I landed on in brainstorming how to address this systemic situation: **Institutional Mistrust**.



Institutional Mistrust

In many organizations, there is a general lack of trust in the abilities and motives of every job function, business unit, and person; this is human nature but sometimes it pervades the culture so much that it is effectively institutional.

- Colin J. Neill, Philip A. LaPlante & Joanna F. DeFranco, "*Antipatterns: Managing Software Organizations and People*"

13 Laplante, P. A., Neill, C. J., & DeFranco, J. F. (2011). Antipatterns: Managing Software Organizations and People. Taylor and Francis Group.

Institutional Mistrust is classified as an Environmental antipattern that has its roots in communication, culture, and honestly. Some of the dysfunction that is symptomatic of this anti-pattern include:

1. Tasks that are duplicated because the results from one group are not trusted by another group
2. Energy is lost with the pointing of fingers
3. Good staff level

Addressing **Institutional Mistrust** on my team needed a crawl, walk, run approach. First, I needed to take responsibility for my work and assume the best in people. Second, I demonstrate model behavior by being trustworthy myself and setting and holding others accountable, helping others to do the same. Finally, the one that would bring the most backlash AND the most reward, refactoring to distribute responsibilities.

Building Human Trust

When we speak about trust in the context of cybersecurity and cyber resiliency, it is usually referencing machine-to-machine trust, or a belief of a machine or sensor to act dependably, reliably, and secure in a given situation.

I am talking about good old-fashioned, elbow-to-elbow trust. In the age of zero-trust architecture, a generation of script kiddies, and surge of cyber espionage, there is an absolute role for trust among the team, the executive leadership, and the entire organization.

There are multiple psychological studies that provide frameworks or equations for how to build trust. In 2000, Charles H. Green wrote a book titled "The Trusted Advisor" where he outlined an equation for trustworthiness.[14] His equation defines the components that need to exist in order for someone to be perceived as trustworthy and for others to extend trust.

The 4 components are:

1. Credibility has to do with words and ideas and how someone expresses and shares.

14 Green, C. H., Maister, D., & Galford, R. M. (2021). The Trusted Advisor: 20th Anniversary Edition. Free Press.

2. Reliability are deeds and actions and while it is action oriented, it does include an emotional element. Does the person understand my unique context?
3. Intimacy has to do with a sense of personal and psychological safety; Do you believe that confidences are kept?
4. Self-orientation deals with how much someone is self-absorbed OR are externally altruistic and empathetic.

The components of the trust equation are as follows:[15]

The Trust Equation

$$\underset{\textit{Trustworthiness}}{\mathbf{T}} = \frac{\underset{\textit{Credibility}}{\mathbf{C}} + \underset{\textit{Reliability}}{\mathbf{R}} + \underset{\textit{Intimacy}}{\mathbf{I}}}{\underset{\textit{Self-Orientation}}{\mathbf{S}}}$$

Establishing trust requires both someone to present trustworthiness and for someone else to extend an offer of trust. To diagnose and understand the steps to take for my team and the organization, we would need to display trustworthiness. Clearly this team of senior architects and engineers had credibility, were well educated, and credentialed.

Reliability was, at times, suspect for many as this was often not within their control. Their ability to "make good" on their word was often at the whimsy of redirection by others. To address this, we would need a shared and transparent cross-organizational backlog. This was a tremendous sticking point! The existing fiefdoms needed reassurances that the intent was not to seize total control, but to protect the team and give them the necessary buffer to deliver on time and as promised.

15 Understanding the Trust Equation. Trusted Advisor . (2020, April 27). trustedadvisor.com/why-trust-matters/understanding-trust/understanding-the-trust-equation.

Intimacy includes understanding the context of others, both specific to the unique domain (child welfare, health care, military, fintech), and the contextual surroundings for the project or program and its impact on individuals. To build intimacy, we must understand one another's role, responsibilities, and cognitive load. As the adage about empathy goes, "You can't understand someone until you've walked a mile in their shoes".

This was not limited to our team and also included the senior executives. They needed to openly share and champion for the actions and vision of the team. Doing this empowers us to propagate trust across the entire department vertically and horizontally. My approach is to build momentum and trust from the top down. Model the positive behaviors you want to ultimately see manifested in a new culture built together. As our team continued with open dialogue, we also needed to problem solve together. With the right guidance, scaffolding, and buffering in place, the team begins to display trustworthiness and extend trust to one another.

I must admit that an undertone existed that attempted to paint me as a "diva demanding full control". I responded with transparency, empathy, and modeling the behaviors I wanted to see in others. I'll concede this was difficult at times; more than once I left for the day with a heavy heart. Bootstrapping myself each morning became my own personal mantra.

Challenge #1: Access to scanning tools for delivery teams

Fundamentally we all agreed that defects needed to be surfaced as quickly as possible to give developers context to quickly address the issue. In my experience, this needs to include any tool that developers can use to quickly uncover defects and flaws. Making this a self-service event removes dependencies on others for handoffs and makes the process as effective and as timely as possible.

This presented an obstacle for us to solve. All security tools and scanners were reserved for the security team and not available to the development organizations. With the dual purpose of building trust through joint problem solving and to enable self-service code quality improvements, I proposed we solve this together.

Step 1: Show how rapid feedback speeds delivery of secure, quality code

The development teams had been toying with test-driven development (TDD) and educated on automated testing. We selected a product team to use as our shared experiment. “Win or Learn Fast,” is the phrase used by Cyber pro, Nicole Dove. Replacing the term “fail fast” became our battle cry.

For the pilot development team, I asked that unit test code be subject to code reviews until developers and team leads cut their teeth on quality and appropriate unit tests. This was viewed as my being too controlling and a few choice sexist slang terms were heard in the hallway chatter. To address the chatter I was consistently open and honest, demonstrating repeatedly that my “self-orientation” component was negligible, with complete focus on the success of the humans in the mix, the teams, and the organization.

I mentored the architects and engineering leads to jump in to take on a user story and code unit tests as well. My point was that you’ll have ineffective code reviews if you’ve never done it. We use the opportunity to execute some pair programming as a type of “trading places” game. This allowed our security pros to demonstrate they could do more than push out security coding manuals. It was a bit stressful; I hate it when I make silly logic errors or when my code is not textbook elegant. I had to do the very thing I was asking of others. At times it was embarrassing, and on a few occasions, I knew my face was a bit flushed, but together we had fun. The shared lesson demonstrated the faster the delivery team had feedback on quality and defects, the more effectively and quickly defects and issues could be addressed.

Step 2: Demonstrate accelerated feedback with self-service tooling

After demonstrating value on rapid code reviews, the next step was to rally for self-service scanning. This proved to be quite difficult! While we were well on our way to tearing down transitional swim lanes across the team, the overall organization and funding were organized so that only the security team could be allowed licenses to the security scanning tools.

Truth be told, there were some lightweight territorial attitudes and fears of no longer being necessary. Much like operations, teams may fear they will be less necessary when replaced by the automated deployments, the security team feared that they would be less necessary. The reality is the opposite! When operations and security are freed up by automation or shared execution by others, they are now free to dive into the more difficult challenges. Both teams could also start working more predictable hours.

It took a few months and negotiation on funding to eventually provide access to the paid version of Burp, Fortify, and other scanning tools. In the interim, we held OWASP lunch-and-learn sessions and encouraged using free versions to fine-tune their skills.

Challenge #2 - Developer Access to Production

Embracing shared responsibility for the health of production was another necessary step in our evolution as a Cyber-centric and DevSecOps-savvy organization. In the past, developers were completely banned from any access to production. They had reacted by creating what I call “Easter egg” health reports. These were screens that were obfuscated from normal navigation, but that provided functional health metrics, such as the number of cases currently opened or time in queue for addressing an issue. Overall, development teams had clandestine and well-intended workarounds to address their lack of access to production monitoring.

This surfaced a fundamental mistake in our team building. While we asked operations to participate on the team, I was so excited by the growing relationship with security that we didn’t passionately pursue operations in all discussions. The doors were open, but we didn’t send the proverbial follow-up note and calls to court them.

Operations had a laundry list of reasons to exclude dev production access, including fear of developers having access to PII data. Debunking this particular opinion was relatively simple. We had a shared cyber faux production access and production data was routinely ported to non-production environments for performance testing, as well as debugging of production defects.

This surfaced a historic lack of trust based on the distance between the historic teams and the strength of the bricks used to route the swim lanes. Residual opinions by the security team held a view of the developers and delivery team that they were more worried about pushing fast features, even if the quality was low, and had "no clue" what security really entails. There was an undercurrent that only operations could be trusted to protect production.

The muscle memory for Ops included NO access for anyone. A multi-stepped process was needed to move forward. First, operations were often left as uninvolved, except for being provided roadmap dates for go-lives along with hardware and software specifications. The enterprise often treated Ops as doers and not as thought leaders and considered them reactive and not proactive. While it may have been true that operations were often **forced to be reactive**, the culpability was horizontal.

Improving quality (feature and cyber) improved the lives of the Operations team and we negotiated for access to operations dashboards for a select set of senior engineers. There was also an agreement that all "Easter Egg" monitoring would not be hidden but openly discussed and secured.

Challenge #3 - Protecting Lower Environments

While we tout using DevOps principles to commit code to the main branch and have it automatically progress through the pipeline to production, there is a need for a mirror production-like environment, especially when about to go live with a new solution or ecosystem.

The number of features to be included in a minimum viable product for a new constituent-facing capability can be fairly extensive, even when taking an agile mindset. While pitching blue/green (A/B) environments was considered a bridge too far to start, I was labeled as wearing rose-colored glasses and endured discussions explaining the costs associated with hosting production and all the accouterments.

Muscle memory was too great with historic environment names, ownership, and usage. The muscle memory of how they went live in the past with new systems meant they did not buy into the need or cost to secure "pre-prod" (pre-production) in the same way.

As the new pipeline started to mature, features were being released into pre-prod and almost instantly pushed into the new production location as well. Pre-prod would be used for performance and load testing and possibly some traditional user acceptance testing. Keep in mind that in any organization, there are generally different identity, credential, and access management solutions for development and testing than for production. I needed to expose the need to have a complete parity environment that was managed with identical rigor and security concerns as production.

As luck would have it, a formal request was made to expose pre-prod to internet traffic for senior stakeholders, funding suppliers, and quite frankly, political leaders to "try out". Think of this like a traditional user acceptance testing environment but open to the internet. I knew we needed to expose the security risk and targeting of pre-prod. It required two steps to expose the danger.

Step 1: Familiarize Stakeholders with Real-time security dashboards

First, I educated the stakeholders about security and analytics dashboards by simply adding basic Google Analytic tagging to an existing LIVE health and human services related website. Using a simple java script added to the inherit base page, the stakeholders would be able to see real-time dashboards. Here is a sample custom HTML tag to capture a visitor's IP address.[16]

```
<script type="application/javascript">
 function getIP(json) {
 dataLayer.push({"event":"ipEvent","ipAddress" : json.ip});
 }
</script>

<script type="application/javascript" src="https://api.ipify.org?format=jsonp&callback=getIP">
    </script>
```

With the stakeholders' permissions, I had my team configure Google Analytics dashboard application on the mobile devices of the attendees. We then set about getting the stakeholders comfortable with navigating the dashboards. They were shocked to see foreign IPs hitting the login page for this solution. Using the geolocation feature, we watched the map highlight hits from Israel, Germany, and Romania. The identity expert and the network firewall expert both explained how the current capabilities were secured.

16 Kravitz, A. (2021, February 25). Get Visitor IP Address with Google Tag Manager. Mixed Analytics. mixedanalytics.com/blog/visitor-ip-address-google-tag-manager/

Step 2: Shock and educate

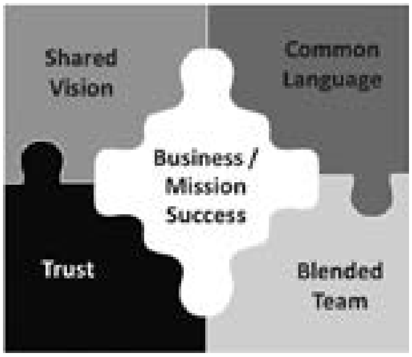
As everyone in the room proudly discussed our seeming ability to thwart production attacks, I had the team help the stakeholders toggle to a different environment: pre-prod. Surely this would be uneventful. What we saw not only mirrored production but was busier. More foreign IPs were targeting the pre-prod environment. Given the rather poor practice of using production data for pre-production performance testing, the exposure risk was evident.

This showmanship was a dangerous step to take given I was still new to the overall enterprise and battling ill-founded perceptions. Using the trust equation, I doubled down on keeping my TQ (trust quotient) high by focusing on self-centeredness (that is, not being self serving) and credibility components. This exercise was needed to shock stakeholders and even some on the team into unifying. It worked.

Summary

Although the bulk of this chapter took place with one agency, I have repeated these techniques and steps with other teams, other organizations, and other clients.

- First, establish a shared vision. If one has been written down already, revisit as a team since everyone in the organization and the effort must support and believe the vision.
- Second, define your language together. Recalibrate and redefine when necessary, breaking down barriers and forging new ground together. Look at the makeup of the team and think like a coach. Do we have both breadth and depth with cross over? If not, recruit!
- Finally, build trust using the trust equation. Be bold in modeling your behaviors and even more so, mentor others by taking them aside and helping them grow.



Cultural building must balance a shared vision, common language, blended teams, and trust to enable success.

One Final Story: Anatomy of a Failure

I would not be an honest or real individual if I didn't share failures as well as successes. Perhaps failure is too strong. We were eventually successful in reaching production, providing new capability at speed, and we learned much along the way. As Nicole Dove says, "Win or Learn!"

In 2018, I had the opportunity to work with one of the US Military Services. The scope of work was to conduct an experiment with the outcomes being a working prototype and the associated RMF package that would be necessary to operationalize the prototype if warranted.

> NIST Risk Management Framework, or RMF, helps ensure organizations are able to address rampant cybersecurity threats by providing "a disciplined, structured, and flexible process for managing security and privacy risk." [...] a framework is just that: a frame of reference from which to adapt according to your needs and situation.[17]

I saw a clear path to a WIN! The architecture was straightforward and the high-level mission objectives and subsequent quality attributes (scalability, security, maintainability, performance) were known and prioritized.

The vision was clear and shared, we had a well-focused and shared lexicon, the team was blended and we had initial trustworthiness displayed by all on the team. With my coach's hat on, I realized we needed to extend the team to have the RMF package creation team, as well as experienced RMF evaluators associated with the team, so we could begin building out our RMF package incrementally. What I did not take into consideration was the difficulty in fighting muscle memory associated with RMF.

RMF is based on a waterfall mentality. When a *project* is completely designed and ready for production release, a full "package" is created and

17 Tracy, R., & Price, G. (2021, April 13). The Risk Management Framework is Dead. Long Live the RMF. www.nextgov.com/ideas/2019/06/risk-management-framework-dead-long-live-rmf/157717/

submitted for evaluation and approval. I enthusiastically invited the RMF group to participate in early design and build as partners.

We underestimated the impact of being matrixed by another organization. Their home organizational structure had too much power and lacked the interest to change. While the mid-level and boots on the ground were interested, they were not given the top cover they needed.

As their coach, I underestimated the amount of pre-season training to help them. The pace of change was too much for this small and part-time group to consume. With limited time to complete the experiment and even more budget limits for upskilling, the path to closure became clear. We resigned to complete the technical build of the experiment and conduct all the requisite tests and such, then enlist the rag-tag group of RMF folks to create the RMF package and shepherd it through the evaluation.

In the end, the overall experiment was considered a success. It met mission objectives and laid the foundation for moving to a cloud-based ability to spin up new tenant-centric, vessel-specific environments for training purposes. It was not without some loss of credibility and a lower overall trust quotient, however. My zeal to help others work in a nimble and agile way had actually increased my self-centeredness. I was hell-bent on rapid new ways of working and had failed to extend intimacy in the form of contextual understanding to the RMF team.

You can learn from both good examples and bad examples. My goal in sharing this final “failure story” is for you to see the importance of the truth quotient, the need to think like a coach, and the tremendous strength of muscle memory!

ABOUT TRACY BANNON

Passionate Architect!!! Tracy (Trac) Bannon is a Senior Principal in MITRE Corporation's Advanced Software Innovation Center. She is an accomplished software architect, engineer, and DevSecOps advisor having worked across commercial and government clients.

Tracy's architecture specialty emphasizes cloud native, decoupled architectures, DataOps/xOps, and DevSecOps. Understanding complex problems and working to deliver mission/business value at the speed of relevance is job one!

Trac walks the walk and talks the talk. She's just as passionate about mentoring and training as she is about architecting sustainable ecosystems! She enjoys community and knowledge building with her teams, her clients, and the next generation of technologists by guest lecturing at universities, leading working groups, and sharing experience stories.

Trac is a featured speaker and panel moderator at industry events holding certifications from Microsoft, AWS, DevOps Institute, PMI, Scrum Alliances, and the Software Engineering Institute (SEI).