Blameless Culture: The Path to Agility in IT

You've inevitably heard the word blameless used in the context of technology and tech culture. The foundation of DevOps and agile and many other methodologies is around blameless post-mortems. Retrospective meetings and recaps that allow for trust and honesty when discussing what went wrong and what went well.

Here's the issue: being blameless is really difficult. Like, really, really difficult.

Battling Human Nature

Blameless meetings in many organizations are more like "don't blame me" meetings which erodes the very foundation of what blameless culture is all about. A societal challenge is getting past the "Not it!!" mentality and embracing the issues without seeking a person to pin it on who isn't ourselves. People often mistake not taking responsibility with being blameless. Just because you don't assign blame, constantly un-assigning yourself from it is not a blameless approach.

It's not easy to admit we made mistakes. Trust me…

A Personal Story: "Wait, that was production?!"

It was 9:30 AM on a Tuesday. Two Active Directory administration windows open, one for production, and one for the test domain with the identical OU structure (to look just like production). As I clicked the option to upgrade the Active Directory version to Windows 2000 native mode and proudly clicked the Change Mode button (clearly stating "this operation cannot be reversed") and the following acknowledgement that it was irreversible, I sat back in the chair and smiled…for a moment.

Then I checked the other window to what I thought was production…and saw the domain version was mixed mode (legacy). But then I also noticed the domain name was the test domain. I had updated production in the middle of business hours during the potentially busiest login period of the day.

I stood up and took a lap around the cubicle. Once I stepped back in and confirmed it had happened, I called my systems architect who worked with me. "James, I made a mistake. I just updated production instead of test."

James calmly said "Ok. Let's get some folks looking at it and figure out if anything happened and what our options are if something broke". James understood blameless culture. We survived the change (without issue, luckily) and I learned that rallying the team around an issue was better than seeking blame. We would work together on other issues that did have far reaching effect at other times, and the blameless culture and teamwork got us through those events.

Hire the Person that Made a Big Mistake – They Won't Want to Make Another One

Making mistakes and learning to change process and recovery procedures meant building better IT. As a systems architect and operations team member, our culture grew stronger under blameless acceptance of issues. I also witnessed the very negative results of not using this tactic. A manager at one time told me that even though we had a large, preventable situation occur from human error,

we don't remove the person. We teach them to not have it occur again. If you make the same mistakes again…and again, well, there isn't a blameless team around who will not take notice and perhaps have to make some changes in your role.

Test-driven development and test-driven infrastructure are wrapped around the foundations of finding the error first and then working back from there. Seek a problem and then find resolution. The same goes for teamwork and production deployments. To move faster, you have to be confident that you are testing and also able to work together as a team without fear of retribution when issues occur. Some issues are avoidable, and some are not. Be blameless before you find the solution and then when you find the cause, accept that it happens…but should happen less. The real leaders then ask "how can we do it better next time?"

Leadership Defined: We Succeed; I Fail

Acknowledging failure or just challenges is healthy, Celebrating success as a team is also extremely positive. My way of encapsulating what it means to be a leader of people in a team goal is "We succeed; I fail". In other words, celebrate successes as a team, because we all did this together. When looking for where we may not have succeeded, look into yourself for what you feel could have been done better. There may be others involved in the tactical things that went wrong. The best outcome is a learning experience for everyone throughout the team.

Even when things go well, we should always ask how it could have gone better, or what would you do differently? This is the beginning of embracing a culture of experimentation. You have to know that things can fail and we can recover, without blame.

All about:virtualization

Thanks to the great community that has been build around virtualization, cloud, and everything that I've been writing about for many years, I have been given some great opportunities to create content to share in those communities.

In my role as Technology Evangelist for Turbonomic, I get to contribute to a very cool community blog which is called about:virtualization that features content related to virtualization, cloud, development, networking, and much more.

Just in case you didn't already find that content, here is the link to be able to read articles there including some from me, and many other great content contributors from the industry.

http://discopos.se/about-virt

Want to contribute to about:virtualization?

We are always looking for community content that can be hosted at the about:virtualization blog. If

you are interested in creating content and being able to get your voice out to the community, please email me to eric.wright {at} vmturbo.com with your contact information and I'll get you on the road to being a published blogger!

Who has a conference? Everyone DOES! – DevOps Enterprise Summit 2014

It seems sometimes that we have a lot of conferences happening. This is a good sign for the strength of the technology sector, and the size of the audience who is prepared to consume this content. As a massive fan of DevOps and the great community wrapped around it, I was very happy to watch some of the recent DevOps Enterprise Summit 2014 which was

The great thing about this conference was that it can be watched virtually, which is exactly how I did it. Luckily the team has been kind enough to post the sessions on YouTube (https://www.youtube.com/user/DOES2014) for our enjoyment.

If I could suggest something to begin with, it's my usual thing which is BUY THE PHOENIX PROJECT

Back to the conference though! The reason that I'm bringing up the book is that the Phoenix Project is that one of the co-authors, Gene Kim led sessions and was part of the opening session also. Gene is a phenomenal resource and a great speaker, so if you have to start anywhere, you should absolutely spend some time watching the opening of the Tuesday session here:

From there, I would say that any session is a great place to start. There are a lot of great user stories about how they had success and challenges with adoption of DevOps practices. This is all great stuff and even if you aren't already on the road towards implementing a DevOps methodology in your organization, this is a good opportunity to spend a little time finding out how it may benefit you.

DevSecOps – Why Security is Coming to DevOps

With so many organizations making the move to embrace DevOps practices, we are quickly highlighting what many see as a missing piece to the puzzle: Security. As NV (Network Virtualization) and NFV (Network Function Virtualization) are rapidly growing in adoption, the ability to create programmable, repeatable security management into the development and deployment workflow has become a reality.

Dynamic, abstracted networking features such as those provided by OpenDaylight participants, Cisco ACI, VMware NSX, Nuage Networks and many others, are opening the doors to a new way to enable security to be a part of the application lifecycle management (ALM) pipeline. When we see the phrase Infrastructure-as-Code, this is precisely what is needed. Infrastructure configuration needs to extend beyond the application environment and out to the edge.

NFV: The Gateway to DevSecOps

Network virtualization isn't the end-goal for DevSecOps. It's actually only a minor portion. Enabling traffic for L2/L3 networks has been a major step in more agile practices across the data center. Both on-premises and cloud environments are already benefitting from the new ways of managing networks programmatically. Again, we have to remember that data flow is really only a small part of what NV has enabled for us.

Moving further up the stack to layers 4-7 is where NFV comes into play. From a purely operational perspective, NFV has given us the same programmatic, predictable deployment and management that we crave. Using common configuration management tools like Chef, Puppet, and Ansible for our regular data center management is now extensible to the network. This also seems like it is the raison d'être for NFV, but there is much more to the story.

NFV can be a confusing subject because it gets clouded as being L2/L3 management when it is really about managing application gateways, L4-7 firewalls, load balancers, and other such features. NFV enables the virtualization of these features and moving them closer to the workload. Since we know that

NV and NFV are Security Tools, not Networking Tools

When we take a look at NV and NFV, we have to broaden our view to the whole picture. All of the wins that are gained by creating the programmatic deployment and management seem to be mostly targeting the DevOps style of delivery. DevOps is often talked about as a way to speed application development, but when we move to the network and what we often call the DevSecOps methodology, speed and agility are only a part of the picture.

The reality is that NV and NFV are really security tools, not networking tools. Yes, that sounds odd, but let's think about what it is that NV and NFV are really creating for us.

When we enable the programmatic management of network layers, we also enable some other powerful features which include auditing for both setup and operation of our L2-L7 configurations. Knowing when and how our entire L2-L7 environments have changed is bringing great smiles to the faces of InfoSec folks all over, and with good reason.

East-West is the new Information Superhighway

Well, East-West traffic in the data center or cloud may not be a superhighway, but it will become the most traffic-heavy pathway over the next few years and beyond. As scale-out applications become the more common design pattern, more and more data will be traveling between virtualized components on behind the firewalls on nested, virtual networks.

There are stats and quotes on the amount of actual traffic that will pass in this way, but needless to say it is significant regardless of what prediction you choose to read. This is also an ability that has been accelerated by the use of NV/NFV.

Whatever the reasons we attach to how DevSecOps will become a part of the new data center and cloud practice, it is absolutely coming. The only question is how quickly we can make it part of the standard operating procedures.

Just when you thought you were behind the 8-ball with DevOps, we added a new one for you. Don’t worry, this is all good stuff and it will make sense very soon. Believe me, because I’ll be helping you out along the journey.