Intro to Patching

Thomas Cameron, RHCA, RHCSS, RHCDS, RHCVA, RHCX
Chief Architect, Western US, Red Hat
thomas@redhat.com
twitter: thomasdcameron
IRC: choirboy on Freenode

Agenda

- Who am I?
- Who are you?
- What is patching?
- Types of patches
- How important are patches?
- From where do we get patches?
- How do I find out when I need patches?
- Think about patching when you design!
- What should we patch? What should we not patch?

Agenda

- Buy vs. build patching systems
- Patch promotion
- Patching cycles
- Collaboration (Charlie Yankee Alpha)
- The future of patching


Who am I?

Who am I?

- Thomas Cameron – Chief Architect @ Red Hat for Western US
- In IT since 1993
- Written curriculum on Linux & Windows
- Been an IT admin in numerous Fortune 500 shops
- Held certs from Novell, Microsoft, TurboLinux, and Red Hat
- Made some stupid, stupid mistakes in my career. LFMF!


Who are you?

Who are you?

- New-ish sysadmin, or someone who is interested in being one.
- My assumption is that you are a beginner.
- There are about a zillion ways to do this stuff, that's one of the cool things (and curses) of this job


What is patching?

What is patching?

- Per Wikipedia:
  - A patch is a piece of software designed to fix problems with, or update a computer program or its supporting data. This includes fixing security vulnerabilities and other bugs, and improving the usability or performance. Though meant to fix problems, poorly designed patches can sometimes introduce new problems (see software regressions). In some special cases updates may knowingly break the functionality, for instance, by removing components for which the update provider is no longer licensed or disabling a device.

What is patching?

- Per Wikipedia:
  - Patch management is the process of using a strategy and plan of what patches should be applied to which systems at a specified time.


Types of patches

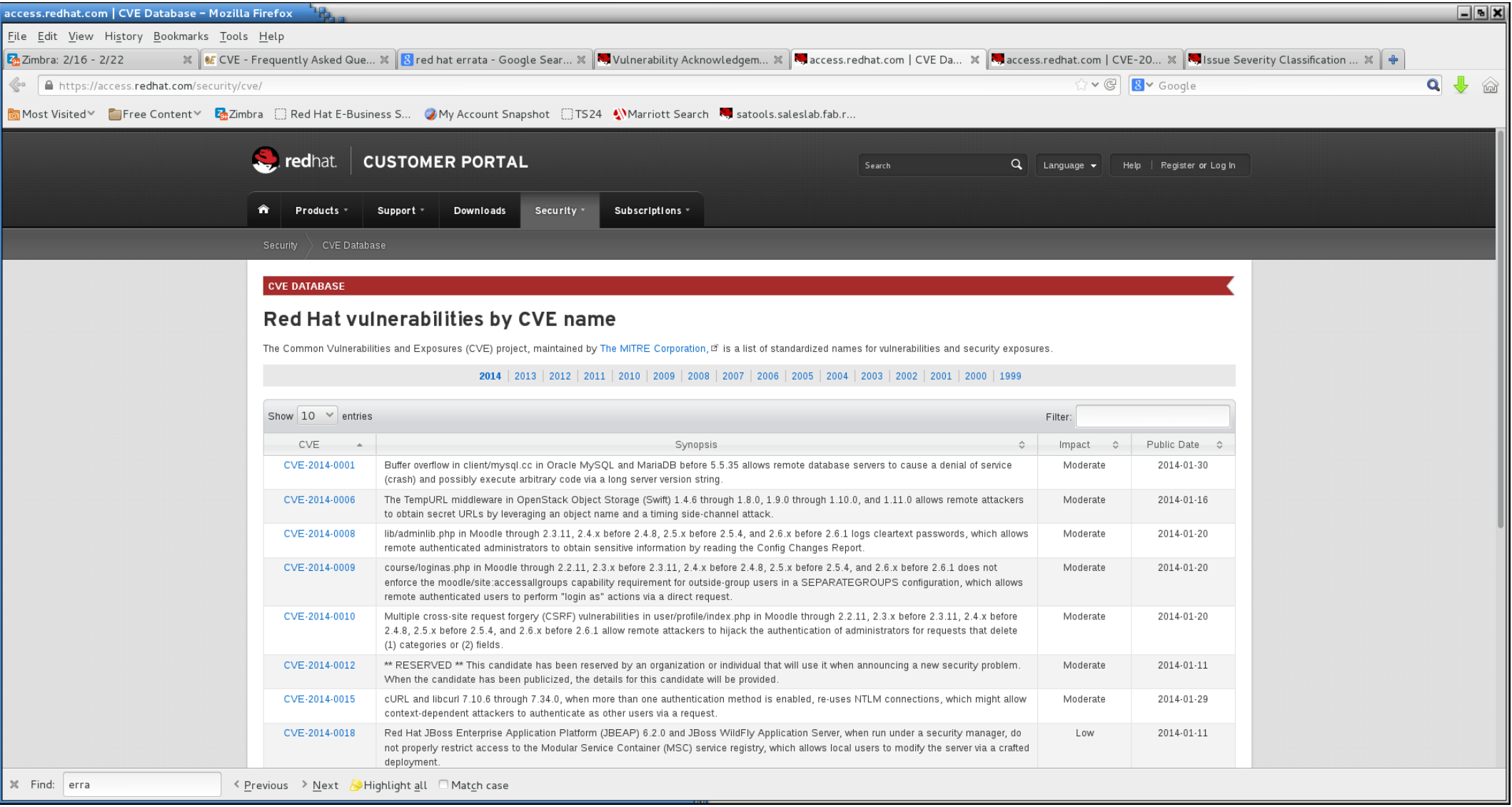
Types of patches

- In very general terms, we will generally need to apply updates for three reasons:
  - There's a bug in software functionality
  - There's an enhancement to a software package
  - There's a security flaw in a package


How important are patches?

How important are patches?

- This varies greatly. I work at Red Hat, and we generally use MITRE Corporation's Common Vulnerabilities and Exposures (CVE) information to determine the severity of patches.


access.redhat.com | CVE Database – Mozilla Firefox
File Edit View History Bookmarks Tools Help
Zimbra: 2/16 - 2/22
CVE - Frequently Asked Que...
red hat errata - Google Sear...
Vulnerability Acknowledgem...
access.redhat.com | CVE Da...
access.redhat.com | CVE-20...
Issue Severity Classification ...
https://access.redhat.com/security/cve/
Most Visited
Free Content
Zimbra
Red Hat E-Business S...
My Account Snapshot
TS24
Marriott Search
satools.saleslab.fab.r...
redhat. CUSTOMER PORTAL
Search
Language
Help | Register or Log In
Products
Support
Downloads
Security
Subscriptions
Security
CVE Database
CVE DATABASE
Red Hat vulnerabilities by CVE name
The Common Vulnerabilities and Exposures (CVE) project, maintained by The MITRE Corporation, is a list of standardized names for vulnerabilities and security exposures.
2014 | 2013 | 2012 | 2011 | 2010 | 2009 | 2008 | 2007 | 2006 | 2005 | 2004 | 2003 | 2002 | 2001 | 2000 | 1999
Show 10 entries
Filter:
CVE
Synopsis
Impact
Public Date
CVE-2014-0001
Buffer overflow in client/mysql.cc in Oracle MySQL and MariaDB before 5.5.35 allows remote database servers to cause a denial of service (crash) and possibly execute arbitrary code via a long server version string.
Moderate
2014-01-30
CVE-2014-0006
The TempURL middleware in OpenStack Object Storage (Swift) 1.4.6 through 1.8.0, 1.9.0 through 1.10.0, and 1.11.0 allows remote attackers to obtain secret URLs by leveraging an object name and a timing side-channel attack.
Moderate
2014-01-16
CVE-2014-0008
lib/adminlib.php in Moodle through 2.3.11, 2.4.x before 2.4.8, 2.5.x before 2.5.4, and 2.6.x before 2.6.1 logs cleartext passwords, which allows remote authenticated administrators to obtain sensitive information by reading the Config Changes Report.
Moderate
2014-01-20
CVE-2014-0009
course/loginas.php in Moodle through 2.2.11, 2.3.x before 2.3.11, 2.4.x before 2.4.8, 2.5.x before 2.5.4, and 2.6.x before 2.6.1 does not enforce the moodle/site:accessallgroups capability requirement for outside-group users in a SEPARATEGROUPS configuration, which allows remote authenticated users to perform "login as" actions via a direct request.
Moderate
2014-01-20
CVE-2014-0010
Multiple cross-site request forgery (CSRF) vulnerabilities in user/profile/index.php in Moodle through 2.2.11, 2.3.x before 2.3.11, 2.4.x before 2.4.8, 2.5.x before 2.5.4, and 2.6.x before 2.6.1 allow remote attackers to hijack the authentication of administrators for requests that delete (1) categories or (2) fields.
Moderate
2014-01-20
CVE-2014-0012
** RESERVED ** This candidate has been reserved by an organization or individual that will use it when announcing a new security problem. When the candidate has been publicized, the details for this candidate will be provided.
Moderate
2014-01-11
CVE-2014-0015
cURL and libcurl 7.10.6 through 7.34.0, when more than one authentication method is enabled, re-uses NTLM connections, which might allow context-dependent attackers to authenticate as other users via a request.
Moderate
2014-01-29
CVE-2014-0018
Red Hat JBoss Enterprise Application Platform (JBEAP) 6.2.0 and JBoss WildFly Application Server, when run under a security manager, do not properly restrict access to the Modular Service Container (MSC) service registry, which allows local users to modify the server via a crafted deployment.
Low
2014-01-11
Find: erra
Previous
Next
Highlight all
Match case

How important are patches?

- Critical impact: This rating is given to flaws that could be easily exploited by a remote unauthenticated attacker and lead to system compromise (arbitrary code execution) without requiring user interaction. These are the types of vulnerabilities that can be exploited by worms. Flaws that require an authenticated remote user, a local user, or an unlikely configuration are not classed as Critical impact.

How important are patches?

- Important impact: This rating is given to flaws that can easily compromise the confidentiality, integrity, or availability of resources. These are the types of vulnerabilities that allow local users to gain privileges, allow unauthenticated remote users to view resources that should otherwise be protected by authentication, allow authenticated remote users to execute arbitrary code, or allow local or remote users to cause a denial of service.

How important are patches?

- Moderate impact: This rating is given to flaws that may be more difficult to exploit but could still lead to some compromise of the confidentiality, integrity, or availability of resources, under certain circumstances. These are the types of vulnerabilities that could have had a Critical impact or Important impact but are less easily exploited based on a technical evaluation of the flaw, or affect unlikely configurations.

How important are patches?

- Low impact: This rating is given to all other issues that have a security impact. These are the types of vulnerabilities that are believed to require unlikely circumstances to be able to be exploited, or where a successful exploit would give minimal consequences.


From where do I get patches?

From where do we get patches?

- This is highly dependent upon what you're running in your environment:
  - Are you using 100% vendor-supplied bits?
  - Are you using a community-supported distro?
  - Did you download and compile yourself?
  - How about third-party repositories?

How do I find out when I need patches?

How do I find out when I need patches?

- If you use one vendor's bits, contact the vendor. Most have notification services.

How do I find out when I need patches?

- If you use a community distro, find and subscribe to their security mailing list.
  - http://www.debian.org/security
  - http://lists.centos.org/mailman/listinfo/centos-announce
  - http://www.ubuntu.com/usn
  - https://access.redhat.com/site/security/updates/advisory
  - And so on

How do I find out when I need patches?

- If you roll your own, it becomes a bit more complex. Most F/OSS projects will have mailing lists. It's up to you to follow them.

How do I find out when I need patches?

- If you use third party repos, they **should** have a mailing list as well.

How do I find out when I need patches?

- You can also follow MITRE's CVE web site:
  - http://cve.mitre.org/
- I also like the United States Computer Emergency Readiness Team (US-CERT) web site
  - http://www.us-cert.gov/ncas/current-activity/
- Security Focus Bugtraq
  - http://www.securityfocus.com/

Think about patching when you design!

Think about patching when you design!

- Remember, if you choose an @everything installation, you're going to have to update a TON of packages – even if you're not using them
  - I start my design with @core or @base (the extra functionality of @core is, IMHO, worth the extra packages to maintain)
  - Then I figure out what the minimum package set necessary for the workload. Web server? httpd. Not necessarily @web-server since that might bring in e.g. php, which you might not need and may even bring in additional vulnerabilities!

What should we patch?
What should we not patch?

What should we patch? What should we not patch?

What should we patch? What should we not patch?

- Patching should not be an automatic thing!
- Analyze the patches available. Do they apply generally (every kernel), or to a specific configuration (only with ipv6 and a specific configuration)?
- Does the criticality of the CVE or update **really** match the criticality of your environment?

What should we patch? What should we not patch?

- I have been in organizations which have turned on automatic updates.

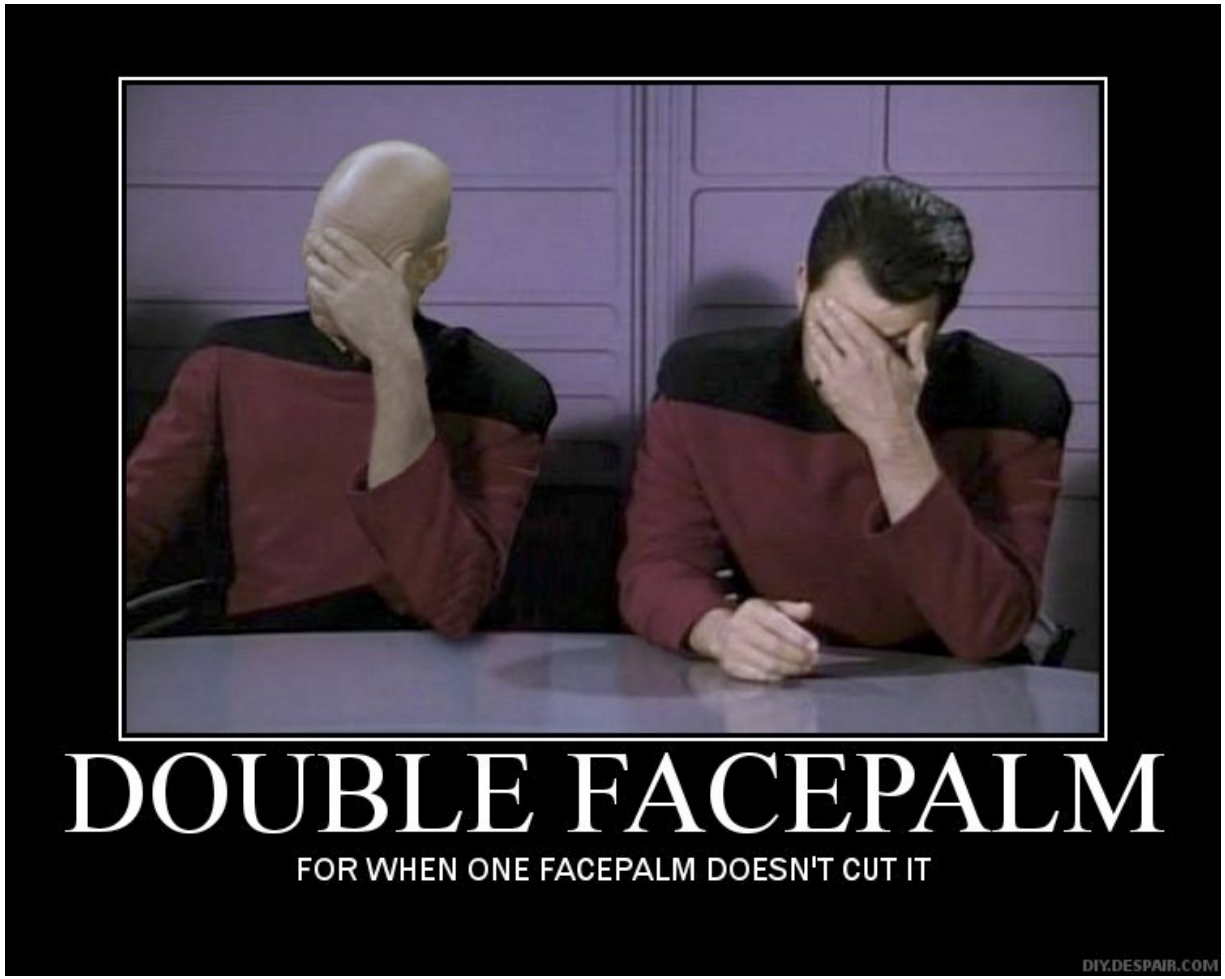
What should we patch? What should we not patch?

- I have been in organizations which have turned on automatic updates.
  - In production.

What should we patch? What should we not patch?

- Wanna guess what the Wednesday was like after the first Patch Tuesday?

What should we patch? What should we not patch?



What should we patch? What should we not patch?

- I did an analysis of Red Hat patches for a customer back in November because they complained that they were rebooting their 15,000 servers “all the time” because of security updates.
  - For the previous year, I found that there was exactly **one** kernel update that actually applied to them based on their production environment configuration.
  - And the risk was effectively mitigated by border security.

What should we patch? What should we not patch?

- I did an analysis of Red Hat patches for a customer back in November because they complained that they were rebooting their 15,000 servers “all the time” because of security updates.
  - However, I also found a handful of performance updates which could have increased their productivity.

What should we patch? What should we not patch?

- Consider the workload running on the servers. Rank your servers.
  - High criticality - outage affects all or a significant portions of the business, 24x7 operation, etc.
  - Medium criticality - departmental or divisional, sporadic workloads, etc.
  - Low criticality – edge servers, servers with distributed workloads, etc.

What should we patch? What should we not patch?

- Consider the workload running on the servers. Rank your your applications in complexity and sensitivity to change.
  - Static web site vs. clustered enterprise DB

What should we patch? What should we not patch?

- Consider the political risk of taking an outage
  - It sucks to have to think about this, but this is a significant reality.
  - Who is the business unit owner? Are they IT friendly or not?
  - How about the application owners? Do the DBAs think they should have root, and will they throw you under the bus to prove their point?

What should we patch? What should we not patch?

- Remember that patching does **not** necessarily mean rebooting!
  - When to reboot
    - kernel, glibc updates
    - Complex updates of multiple services
    - When needs-restarting(1) says you need to
  - When to restart services
    - Pretty much every other time

Buy vs. build patching systems

Buy vs. build patching systems

- There are a number of really good F/OSS patching systems out there.
  - Spacewalk
  - Katello
  - mrepo
  - reposync + createrepo
  - ... and the list goes on and on

Buy vs. build patching systems

- Pros of F/OSS patching systems:
  - They're free!
  - They're flexible!
  - You'll learn a LOT!

Buy vs. build patching systems

- Cons of F/OSS patching systems:
  - They're free (like a puppy)!
  - No SLA for bugfixes
  - No SLA for trouble reports
  - Licensing/redistribution concerns
  - They're usually snowflakes

Buy vs. build patching systems

- Pros of commercially supported patching systems
  - Commercial support
  - SLAs
  - Often times do more than just package updates
    - Config management
    - Monitoring
    - Workflow/life cycle management

Buy vs. build patching systems

- Cons of commercially supported patching systems
  - Commercial support
  - They cost money - sometimes a LOT of money
  - Many times they're opaque
  - One size fits none
    - “We do all platforms - badly!”
  - Third party repository support is often sketchy
  - Licensing/redistribution concerns

Patch promotion

Patch promotion

- I strongly recommend that you develop a patching plan which involves multi-step patch promotion
  - From vendor to development
    - Subscribe dev systems to the development channel and test
  - From development to QA
    - Subscribe QA systems to the QA channel and test
  - From QA to production
    - Subscribe prod systems to the prod channel and deploy

Patch promotion

- Have a backout plan!
  - For Red Hat based systems, it can be as simple as
    - yum downgrade [foo]
  - For Debian based systems
    - apt-get -t=<target release> install <package-name>
  - It might be as extreme as reinstalling the system to the previous installation version


Patching cycles

Patching cycles

- Determine what makes sense in your environment
  - It will not be the same from company to company!
    - Retail environments may have a freeze from the week before Thanksgiving through December 26th or January 1st.
    - Financial services may freeze around tax time
    - Third party app releases may interfere with OS patching, and vice versa


Collaboration
Charlie Yankee Alpha

Collaboration (Charlie Yankee Alpha)

- When you are developing your patch promotion process and your patching cycles, adopt and Open Source methodology!
  - Be inclusive
    - Bring in all the stakeholders early and often
  - Be collaborative
    - Solicit feedback - and take it to heart
  - Get buy-in
    - Involve the stakeholders in testing - get their sign-off

Collaboration (Charlie Yankee Alpha)

- There's also a “Charlie Yankee Alpha (CYA)” factor
  - When you include the stakeholders... especially when they sign off on changes, you're not as likely to be hammered **when** - not **if** - things go wrong


The future of patching

The future of patching

- As we move towards DevOps and other Buzzword 2.0 compliant technologies, patching will change

The future of patching

- I've given up on the crystal ball, but here are some things that seem to be trending:
  - DevOps means more, smaller changes, with more collaboration, more rapidly. This is generally a Good Thing™
  - Huge, disruptive changes are becoming less desirable, and being proven to be unnecessary.
    - This does NOT obviate the necessity of dev --> QA --> prod cycles, it just makes it easier.
  - If you get a chance, read The Phoenix Project. It's a great novel which is an intro to DevOps and how it can be adopted.


Questions?


Thank you!