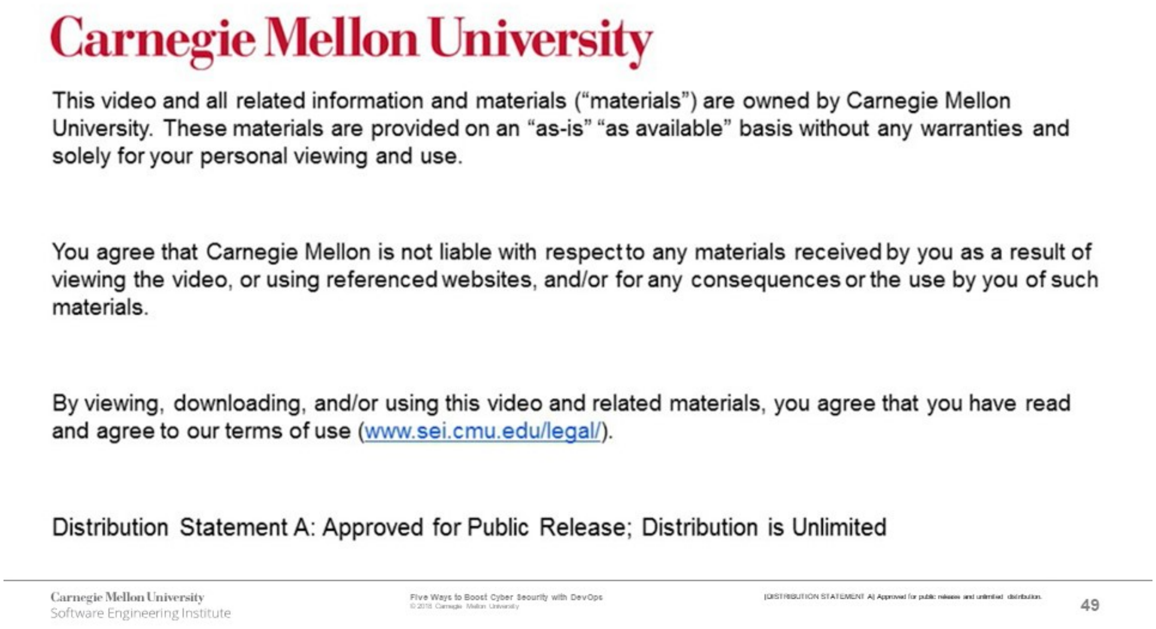
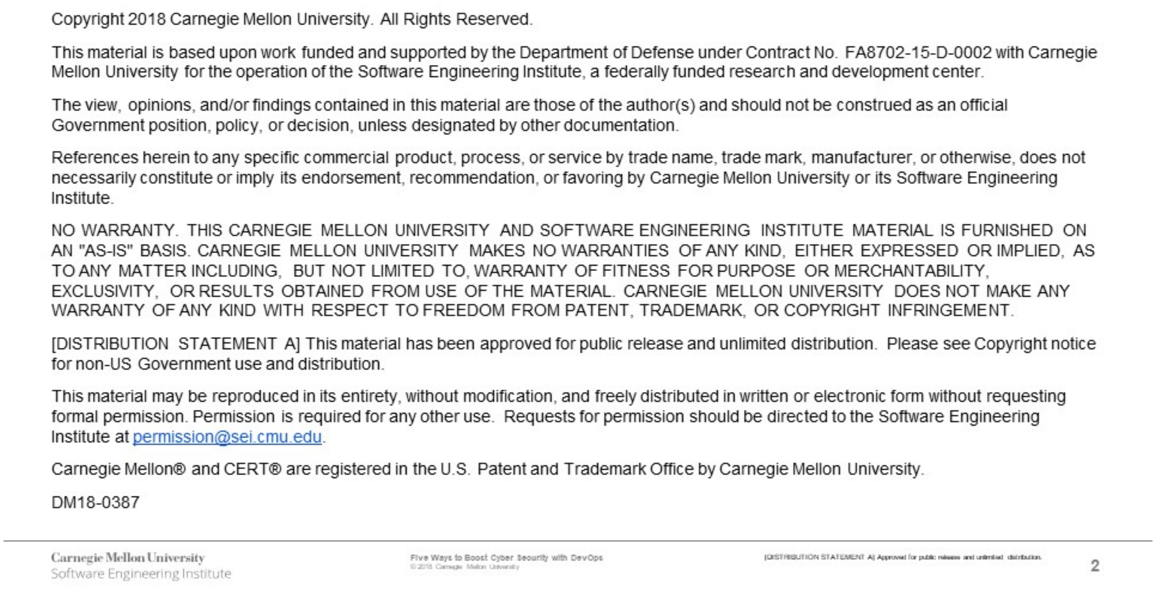
Five ways to boost Cyber Security with DevOps

Table of Contents

Copyright 2018. Carnegie Mellon University



Copyright 2018.



Five Ways to Boost Cyber Security with DevOps

Collaboration

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

**004 Speaker 1: And hello from the campus of Carnegie Mellon University in Pittsburgh, Pennsylvania. We welcome you to virtual SEI. Our presentation today is "Five Ways to Boost Cybersecurity with DevOps." We thank you for participating in today's presentation. We will take as many questions as we can throughout the talk. So, you can feel free to type them into our chat tab or the Q and A tab, depending on whatever platform you're watching this on. And as I stated, we will take as many questions as we can during the talk and as many at the end.

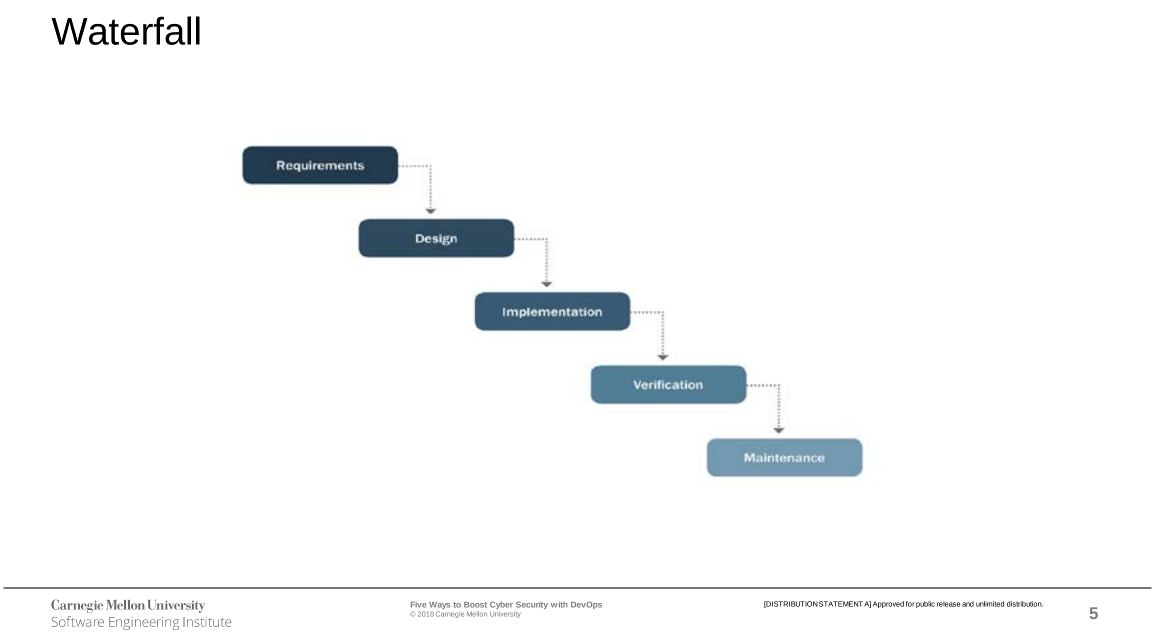
Now, I'd like to introduce-- I should introduce myself first. My name is Shane McGraw, and I'll be your moderator for today's presentation. And now, I'm going to introduce our two panelists for today's talk. First is

Doug Reynolds. Doug is a software engineer within our CERT Division. Doug focuses on creating and improving DevOps and software solutions for government entities. Next, we have Aaron Volkmann. And Aaron is a software engineer and DevOps researcher within our CERT Division where he specializes in product development research. Aaron advises and assists Department of Defense acquisition programs in their adoption of DevOps principles and automation technologies.

Before I turn it over to Aaron, I'll just remind everybody we'd like you to fill out our survey upon exiting today's presentation as your feedback is always greatly appreciated. And you'll see that link to the survey in the chat tab now. So, Aaron, welcome, all yours.

Speaker 2: Thanks, Shane. So, we can't-- let's just roll right into it. Our number one way to boost cybersecurity with DevOps is collaboration. Often at times, whenever people talk about DevOps, the number one quality attribute of DevOps is collaboration. It's the people aspect of it. We'll have organizations though, because it's something that can be seen on an issue tracker, it's something that can be built with a definite deliverable, a lot of organizations focus on automation. But our number one biggie is the collaboration between people. So, that's our first topic. Just to get us rolling into this--

Waterfall



Waterfall

Carnegie Mellon University
Software Engineering Institute

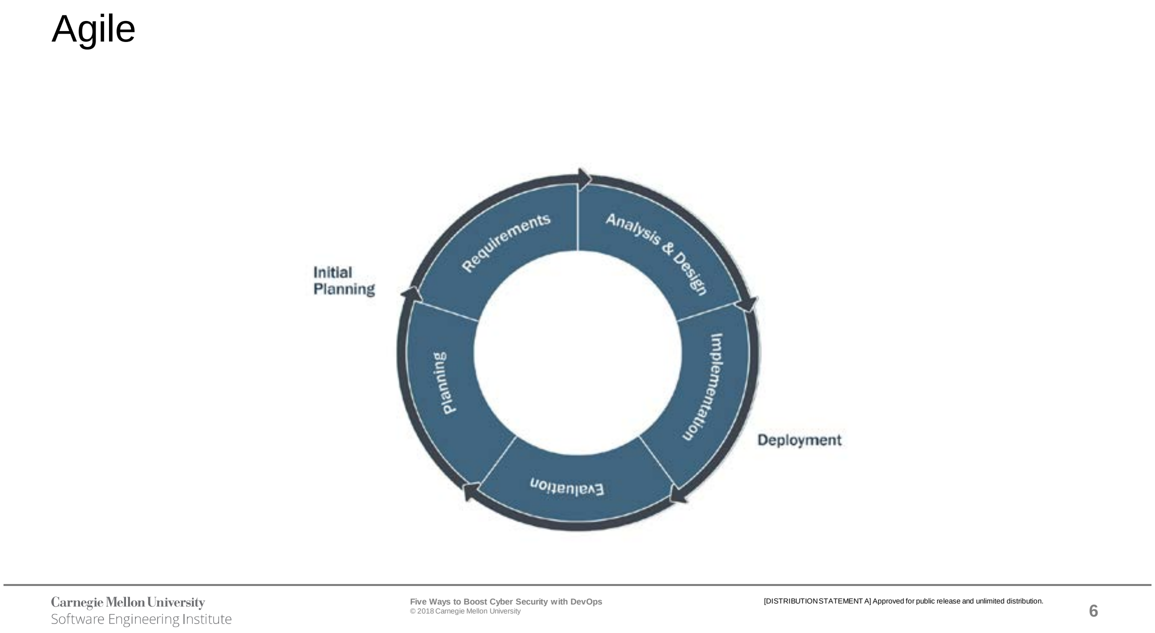
Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

5

**005 We'll talk about what is DevOps, super quick. This is a waterfall methodology.

Agile



Agile

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

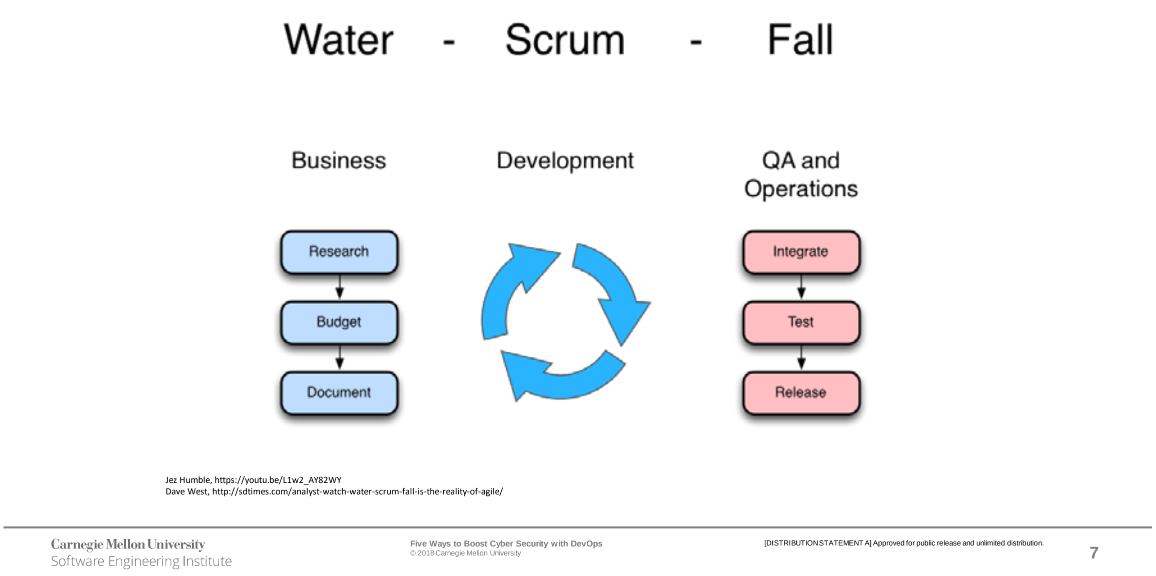
[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

6

**006 This is an Agile methodology.

Lots of businesses, whenever they look to modernize their software development, implement Agile. And whenever they start getting their development teams doing standup meetings every day, what they end up with is this right here.

Water-Scrum-Fall



Water - Scrum - Fall

Jez Humble, https://youtu.be/L1w2_AY82WY
Dave West, http://sdtimes.com/analyst-watch-water-scrum-fall-is-the-reality-of-agile/

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

7

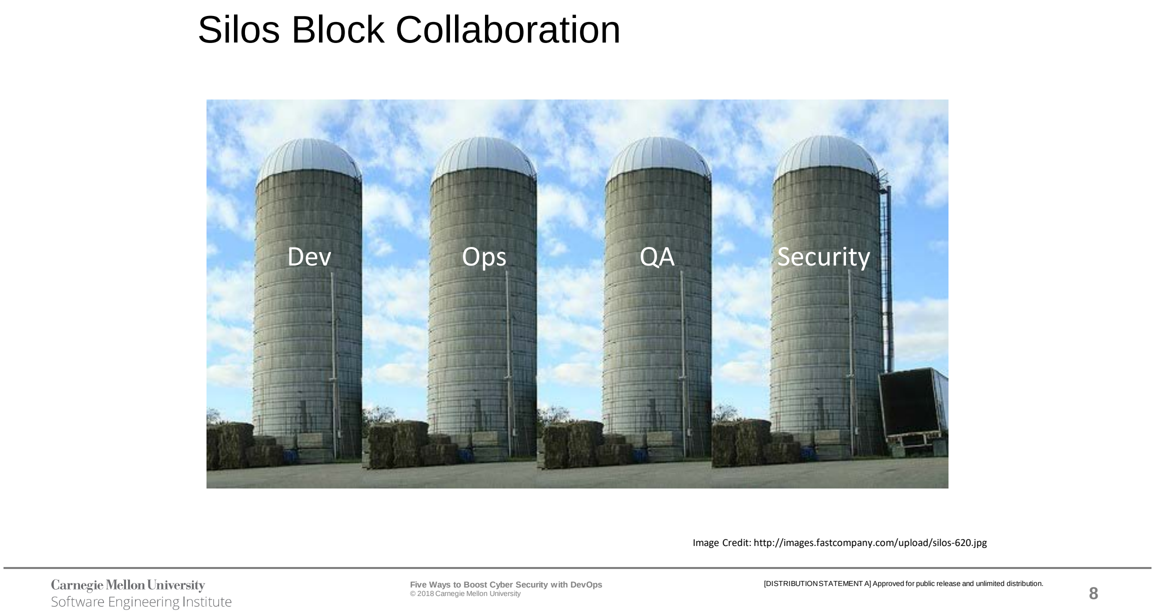
**007 Water, Scrum, Fall, where we have monolithic business processes on the left.

Our development team's going to release software very, very quickly in the middle, and then our QA and operations teams doing very monolithic, slow processes on the right. And as our developers release code very, very, quickly, you find that QA and operations teams are unable to keep up.

Speaker 3: Yeah, and the good thing about Agile is is Agile really helps the development process because waterfall, the way it's implemented, generally slows down the development process. And a lot of companies see a lot of better software and increases in patches and whatnot when you're developing things. But the problem is is Agile mainly just focuses on creating great software, not so much on running great software or getting great user feedback for their software.

Speaker 2: Right, I think it's--

Silos Block Collaboration



Silos Block Collaboration

Dev
Ops
QA
Security

Image Credit: http://images.fastcompany.com/upload/silos-620.jpg

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

8

**008 Embedding the customer and making sure we're building the right thing but not necessarily operating it in the real world. A big tenet of DevOps is breaking down business silos. As you see here, here's our picture of the silos. These silos block collaboration mainly because people within a silo, day to day, they only work within a silo. And they report to leaders that are within those silos. And naturally speaking, these different groups can have competing incentives and have competing different goals.

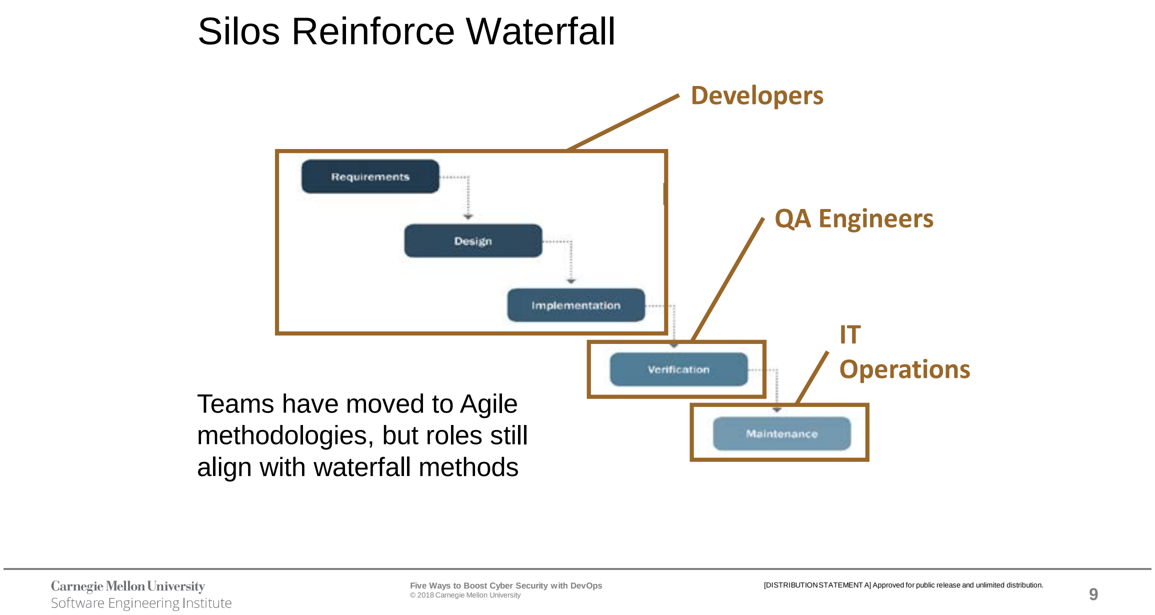
Speaker 3: Yeah, and a lot of times, some of these teams might not even talk to each other. Other times, the teams hate each other. You have security folks doing-- checking on the application saying, "This is insecure. You can't have this in your app." And then it's just not a happy process normally.

Speaker 2: Yeah, whereas devs, they just want to get their stuff out the door. And the QA and operations are raising the red flag and blocking their progress.

Speaker 3: Exactly.

Speaker 2: So, what DevOps aims to do is get these different silos working together and all on the same page.

Silos Reinforce Waterfall



Silos Reinforce Waterfall

Teams have moved to Agile methodologies, but roles still align with waterfall methods

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

9

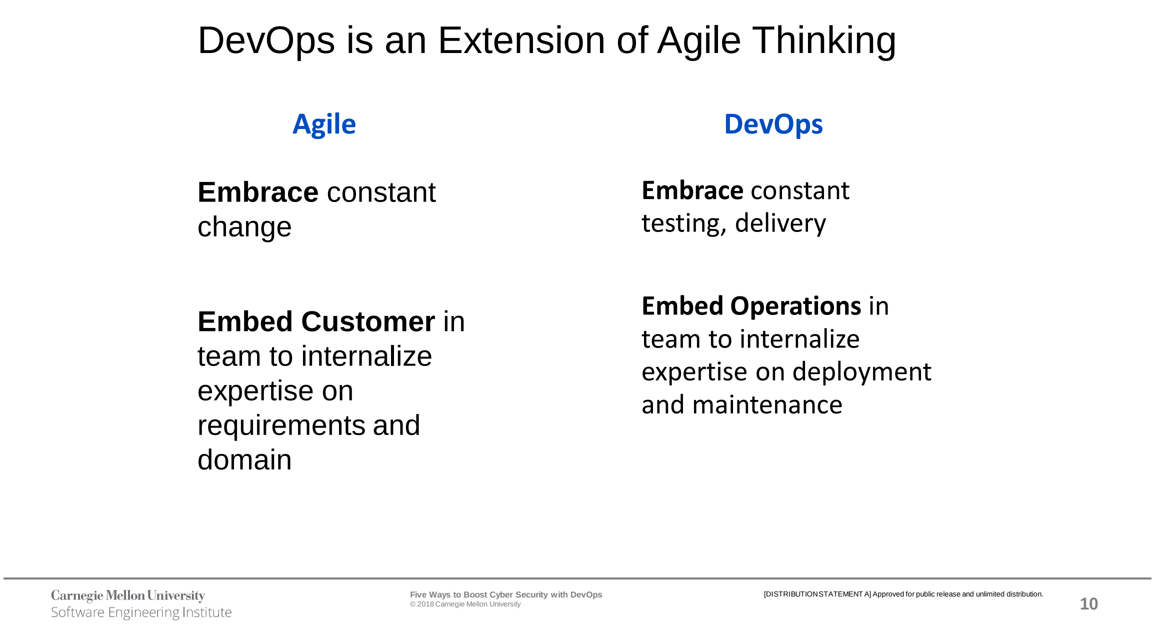
**009 I just want to say having your organization organized in business silos where there is no collaboration, these reinforce the waterfall software development methodology. It makes sense logically. We have our developers concerned with the requirements, the design, the implementation. Then once their completely done, hand off their work to QA. Once that activity is done, look at-- hand it off to IT Operations. We're talking about strictly DevOps, but security's mixed in here as well. Perhaps, at the end in order-- especially on DoD programs where the information assurance aspects are only taken care of at the end.

Speaker 3: Sure and at least during this process, you hope that somebody's doing the security process because a lot of times, developers are more focused on the new shiny thing, getting the code out, and not taking a security stance, since security is kind of paramount that everybody in the process has security in mind.

Speaker 2: Right because security is everybody's responsibility.

Speaker 3: Exactly.

DevOps is an Extension of Agile Thinking



DevOps is an Extension of Agile Thinking

Agile	DevOps
Embrace constant change	**Embrace** constant testing, delivery
Embed Customer in team to internalize expertise on requirements and domain	**Embed Operations** in team to internalize expertise on deployment and maintenance

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTIONSTATEMENT A] Approved for public release and unlimited distribution.

10

**010 Speaker 2: So, DevOps is an extension of Agile thinking. In Agile, we want to embrace constant change to our software. We want to embed the customer in the process of making the software to make sure we're building the right thing that the customer actually wants. DevOps is an extension of that where, in addition to embracing changing the software quickly, we want to embrace delivering the software constantly and embedding operations teams so that we get their-- so we make sure that the software works once it gets deployed.

Now, the term is DevOps, but we can also see that the same principle can be applied to-- extending to security, to legal. So, we can say LegDevOps, D-- DevSecOps, keep roping in the-- we want to keep roping in the relevant stakeholders, into the acronym. But for the sake of conversation, we just call it DevOps.

Speaker 3: Yeah, DevOps is normally where most organizations focus because that's normally where the biggest silo--

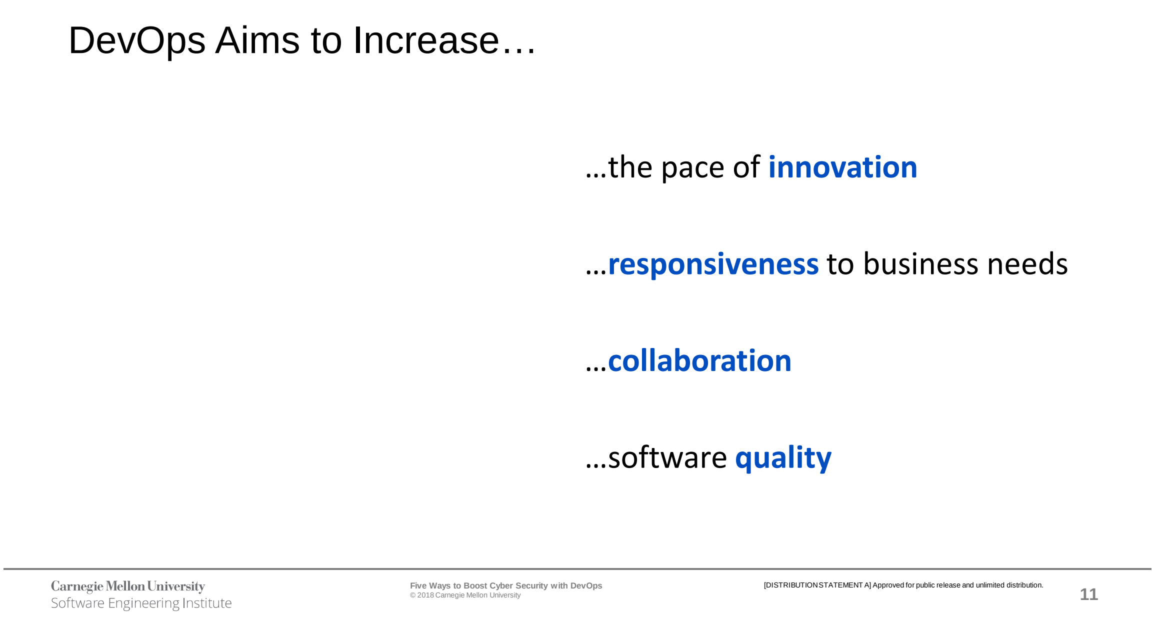
Speaker 2: Right, they're at the core of-- They're at the core of things.

Speaker 3: Yes.

Speaker 2: But our focus today is how to bring security into the DevOps process.

Speaker 3: Exactly.

DevOps Aims to Increase…



DevOps Aims to Increase…

…the pace of **innovation**

…**responsiveness** to business needs

…**collaboration**

…software **quality**

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

11

**011 Speaker 1: So, a quick comment just from an audience member is are you saying-- is the waterfall process bad for DevOps and security?

Speaker 2: I'd say--

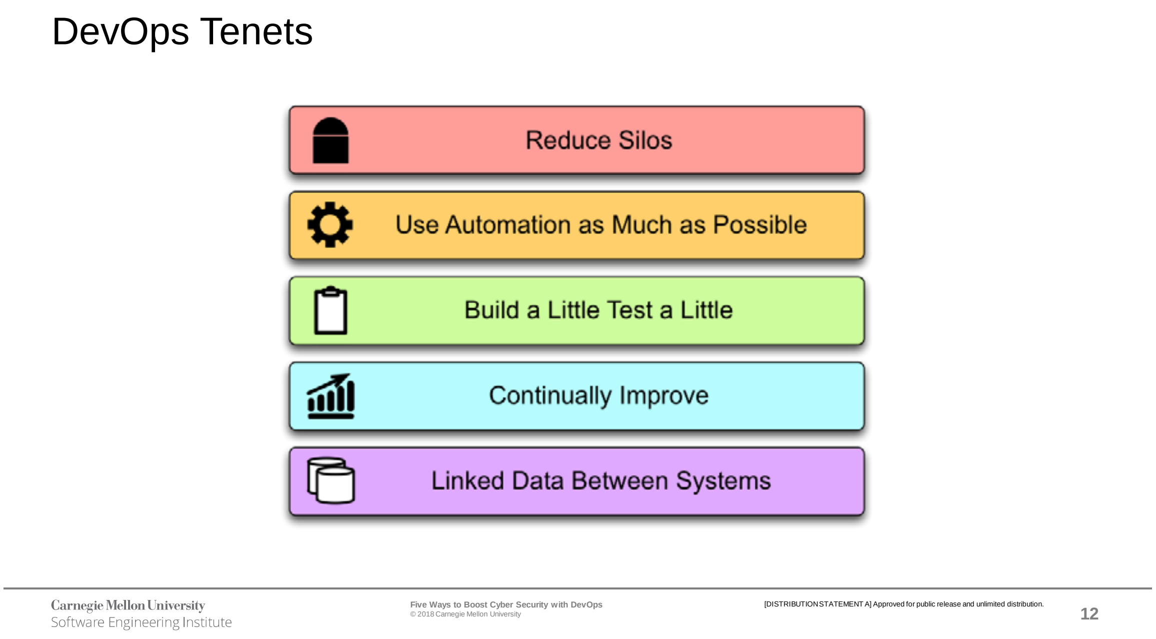
Speaker 3: It's not necessarily bad, but the-- with the waterfall process is it's kind of-- everything's kind of designed to be in a compartment. So, when everything's in a compartment, then security is just a compartment kind of stuck on the waterfall. So, perhaps it's at the beginning of the waterfall if you're doing a successful waterfall project, or maybe it's the last jump where the salmon jump down to get along the river. And then you're just a second thought because you're just trying to get the thing out the door because your deadlines are up. You have people saying, "Where's my product?" And you're like, "Well, it's-- the security isn't any good. So, we have to fix it."

Speaker 2: Right, or else it's best effort security. So, I'd say that-- I say it a conference talks a lot of times. If McDonalds' McGriddles taught us with syrup is securities best baked in from the beginning.

Speaker 3: Yes, that is very true.

Speaker 2: So, just to wrap things up, DevOps aims to increase the pace of innovation and responsiveness to business needs, increase collaboration, and increase software quality.

DevOps Tenets



DevOps Tenets

- Reduce Silos
- Use Automation as Much as Possible
- Build a Little Test a Little
- Continually Improve
- Linked Data Between Systems

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

12

**012 Just to go over some DevOps tenets, DevOps aims to reduce business silos and increase the collaboration and communication amongst teams to make sure that-- to increase the likelihood that we're going to get it right the first time. We want to use automation as much as possible. We want to build a little, deploy a little, test a little. We want to have a culture of continually improving our daily work habits to get better and better, and in addition to that, make sure that we have our systems, our automation systems, linked so that we can fully harness them and take advantage of the data that these automation systems provide, so that we have transparency and invisibility to all stakeholders on what's going on.

Speaker 3: Yeah, visibility's important especially with other teams trying to find out when you're deploying. Upper management wants to know where the status is. So, visibility is key. And that's one of the whole parts of the collaboration.

Speaker 2: Yeah, I remember being in a shop where the only the only way we had visibility is to have status meetings where people would talk together, talk to each other. Where, if we have automation-- these automation capabilities and dashboards, it democratizes the information so that everybody can be on the same page without getting together and speaking about it verbally.

Speaker 3: Sure, and then what always happens with status meetings is you have higher managers, midlevel management, and they have other concerns to deal with, so they can't make the status meetings. So, they're getting second hand information from the person who was there and is their liaison. So, they may not have a full grasp on what the goals were and what management was expecting.

Speaker 2: Yeah, I've been in different shops advising them on DevOps where I attend all these status meetings and sort of follow the thread. And the midlevel managers, I see the same crowd all day long in the meetings that I attend. And I think to myself you know important decisions are being made. But at what point does this information get- if these people are in meetings all day, at what point does this information get disseminated down to the individual contributors?

Speaker 3: Yeah, and that is a good concern because unless you're in that particular meeting that that-- when that decision's being made, you're kind of out of the loop, especially if you're-- if it's involving a software piece that has a lot of functionality built into it. And it's an important feature.

Five Ways to Boost Cyber Security with DevOps

Five Ways to Boost Cyber Security with DevOps

Unified Data

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

**013 And you're kind of on the last-- the last person to know. You're like, "Well, I could have been working to make this better."

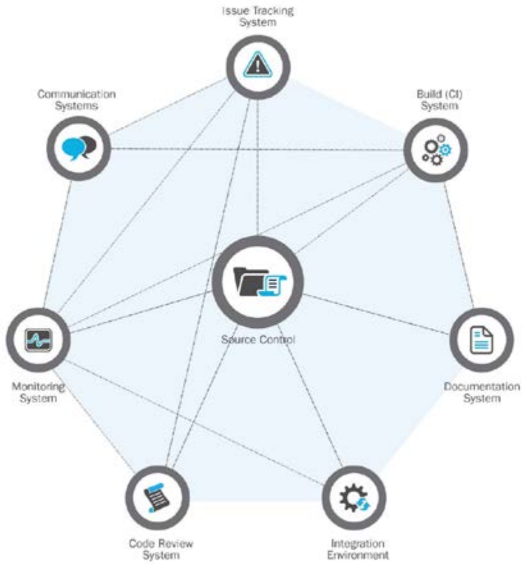
Speaker 2: Yeah, the old game of telephone.

Speaker 3: Yes.

Speaker 2: All right, our next way to boost cybersecurity is unified data.

Speaker 3: Unified data.

Integration and Communication-1



Integration and communication, even among tools, is key to assuring security!

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

14

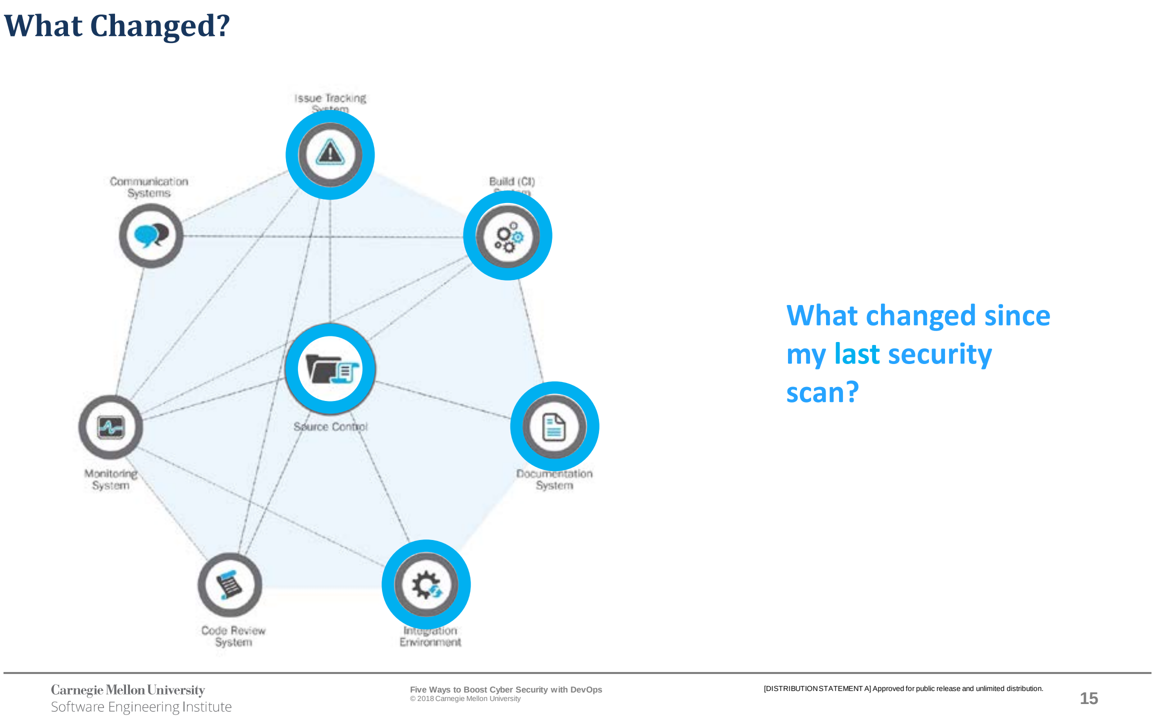
**014 Speaker 2: We've been practicing that a lot all night.

So, up on the screen, we have-- this is a representative diagram with a bunch of different types of tools that you might have in your DevOps automation environment. And the lines between them, they may vary depending on your environment. But the point here is that we want all these tools to be able to share their data and talk to each other where it makes sense. We talked about how important it is for the human aspect to be integrated and collaborating and communicating. The tools doing the same thing is just as important if not more because that's how we can give everybody visibility of what's actually going on.

Speaker 3: Yeah, tools-- we try to not emphasize tools in DevOps, but they do serve a great purpose as long as you don't just consider having tools-- that's how you get your DevOps. You just have a bunch of tools doing all these things that nobody pays attention to.

Speaker 2: They say tools don't get you DevOps sort of like the same way money doesn't buy you happiness, but it sure helps.

Speaker 3: It sure can help, yes.



What Changed?

What changed since my last security scan?

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

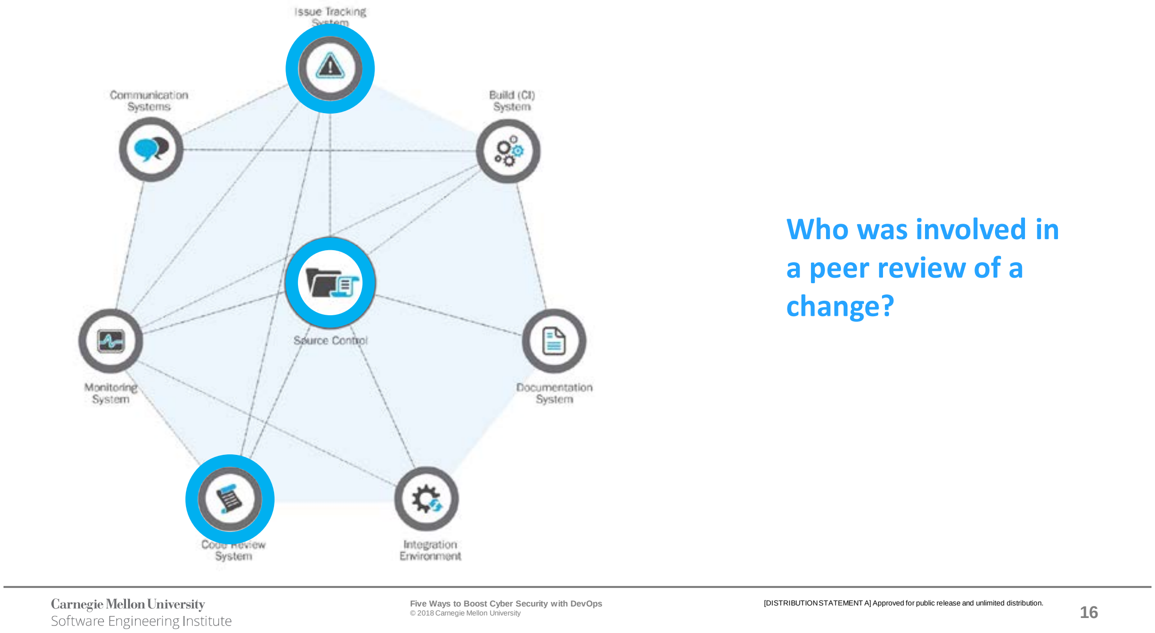
15

**015 Speaker 2: So, just to go through some use cases of questions you might want to ask and what tools need to be integrated in order to answer these questions. So, there's some sort of security incident. We're saying, "What changed? Why is our security posture different?" We want to look at the data from our issue tracker. We want to look at source control and know what changes to the source code were linked to what issue. We want to be linked to our documentation system so that we know what changes to the documentation-- what's been written by tech writers about the latest changes. We want to be connected to our build system, so we can know when and what was deployed. We also might want to look at our integration environment to inspect the actual implementation.

Speaker 3: This also-- this workflow here, it goes for security. It also goes for production support for people that have to deal with outages due to development deployments.

Speaker 2: Yeah, the focus of this is on security but this-- security's really a QA problem. If you have a security vulnerability, that's sort of the same thing as the QA defect in terms of how we can identify and how we can roll out a fix for that.

Who was Involved?



Who was involved in a peer review of a change?

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

16

**016 If we want to know who was involved in a peer review of a change, say a bad change went in and we need to beef up our peer review process. Having an integrated code review system lets us have a hard record of who was involved in looking at looking at a change before it went in.

Speaker 3: Yeah, everybody likes to get blame command until you realize it points back to you're the person who did the commit.

Speaker 2: Yeah, and I think every single IT shop under the sun has a policy that every change must be peer reviewed. But in organizations that I've been in, it's very inconsistent. It can vary from team to team how the peer review process is instituted. Sometimes, they follow a checklist, and they do their due diligence. Sometimes, after a standup meeting, it might be looking over somebody's shoulder and saying, "Oh, that looks good enough. We need to get this out right away." So, by integrating a peer review system with the changes, we can better have more tight control and more visibility over who's involved with the peer review process and get a more standardized control over how that happens.

Speaker 3: Sure, because part of the review process-- sometimes, the other thing that can happen is you might have a review process, but you have to push a change out. Maybe it's a quick fix. Maybe it's a security issue. And I say, "Oh hey, Aaron. I'm doing XYZ. What do you think?" And you're like, "Oh yeah, it's okay." But if you don't have a gatekeeper to say, "No, you have to do a full review. Review the disk, check the code." And it's important just to have that as a step.

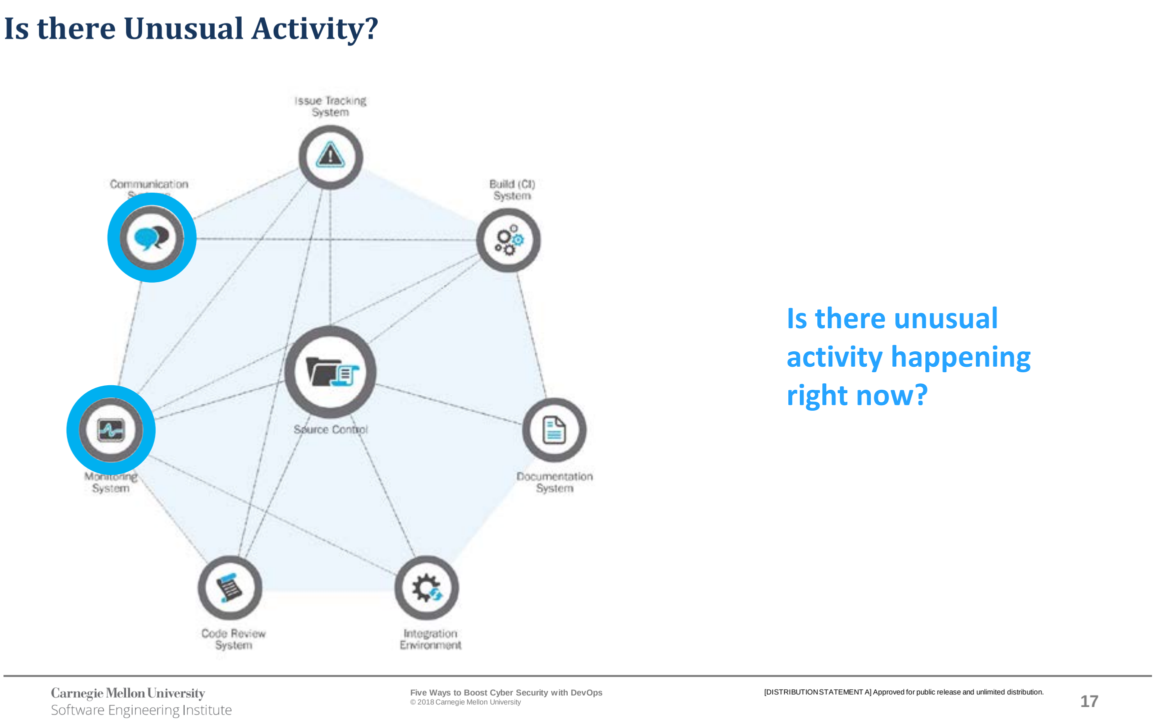
And additionally, it provides a lot of documentation because say you go on GitHub, and you look at, for example, you go to Django, and you look at some of the pull requests, or even some of the discussions you see for the Linux kernel, the pull request has a lot of documentation where there's a lot of good back and forth about what the change is doing, what the change might affect. And it's a good history lesson for a developer who wasn't around for that change. And it's good for the person who was there that can't remember exactly why they put that change in or the details about it.

Speaker 2: Yeah, I can see how that's crucial for solving the bus problem whenever there's staff turnover. By having something written in ones and zeroes, so to speak, that can be preserved. That's a great history to learn why certain things were made and to pass on knowledge about the system to future generations.

Speaker 3: Sure, it beats handing the new person a four-hundred-pound manual.

Speaker 2: Yeah, that might be out of date, or just read-- typically, what I've found is you just have to read the source code to figure out how it works.

Speaker 3: Exactly.



Is there Unusual Activity?

Is there unusual activity happening right now?

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

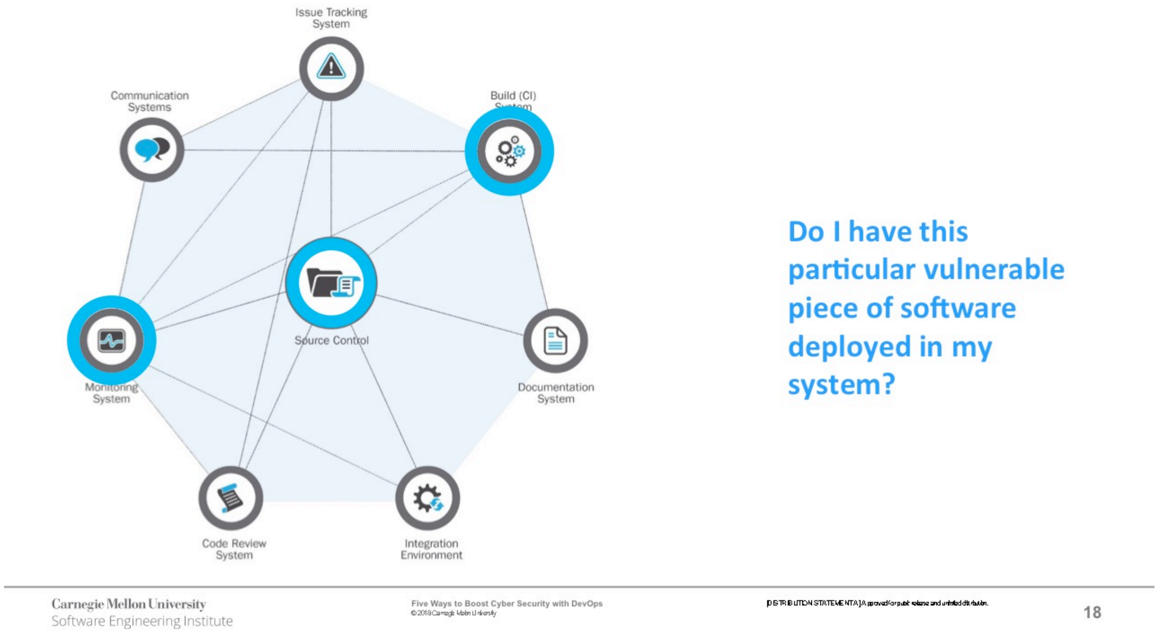
[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

17

**017 Speaker 2: Another question, "Is there anything unusual happening right now in my system?" Say we're deployed, and I just want to know is there something up with my system. We can look at our monitoring system in addition to our communication system. If those things are integrated together, we're talking about maybe a monitoring system firing alerts to a chat window or sending emails to whoever should respond to those things.

Speaker 3: Yeah exactly, chat ops, a lot of places integrate that specifically with monitoring, and it makes it easier for somebody to say, "Hey, I'm jumping on this problem." And also, some folks are putting bots in there now too, so they can say, "Hey, do XYZ." Then they can look at the problem when they're taking a node out of a load balancer to fix whatever the issue is.

Do I have this particular vulnerable piece of software?



Do I have this particular vulnerable piece of software deployed in my system?

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

18

**018 Speaker 2: Next question we might want to answer, especially if there's a security incident, "Do I have this particular piece of software deployed anywhere?" We have visibility of that through our monitoring system, through our source code repository, and through our build CI system so we can see when things are deployed.

Speaker 3: Sure, not every vulnerability is easy as looking for ShellShocked, and hopefully you have UNIX boxes where you just know you have to patch everything. You might just have maybe OpenLDAP has a security problem, and you only have two OpenLDAP servers. But maybe you have a smattering of an application across hundreds of nodes. And if you can just do a quick query and get a list of nodes, it makes it so much easier.

Speaker 2: Yeah, I think it also makes it easier to get a handle on how many different versions of a particular library or piece of software we have so that we can make an attempt to try to standardize within an organization on the number of versions we have in order to make it easier to mitigate incidents like that.

Speaker 3: Sure, and that is a very valid thing because some libraries, you would think that the semantic versioning would be version one is all compatible. Version two is all compatible. But personally, that experience where I think it's libpng or libjpeg, one point four to one point five is exactly nothing alike in binary compatibility. And one OS version has one version or another. So, if you're trying to move your code from one side to another version of the OS, you're just like, "What's happening here?"

Speaker 2: I know. I thought semantic versioning was the major version stays the same, we're supposed to be all good. But reality--

Speaker 3: It's not always the case.

Speaker 2: Not always the case.

Five Ways to Boost Cyber Security with DevOps

Security Hardening

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

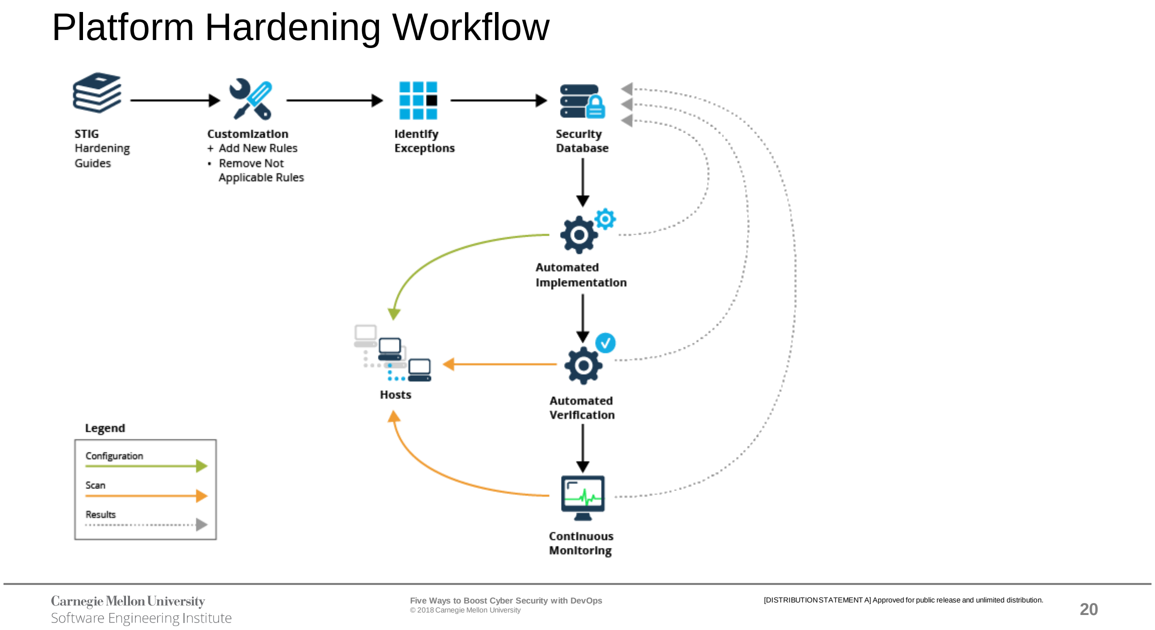
[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

**019 Our next way to boost cybersecurity is security hardening.

Speaker 3: Security Hardening.

Speaker 2: So, this is all about hardening our platform, platform security, the servers, the networks that our software runs on.

Platform Hardening Workflow



Platform Hardening Workflow

STIG Hardening Guides

Customization
+ Add New Rules
• Remove Not Applicable Rules

Identify Exceptions

Security Database

Automated Implementation

Automated Verification

Continuous Monitoring

Hosts

Legend

Configuration

Scan

Results

Carnegie Mellon University
Software Engineering Institute

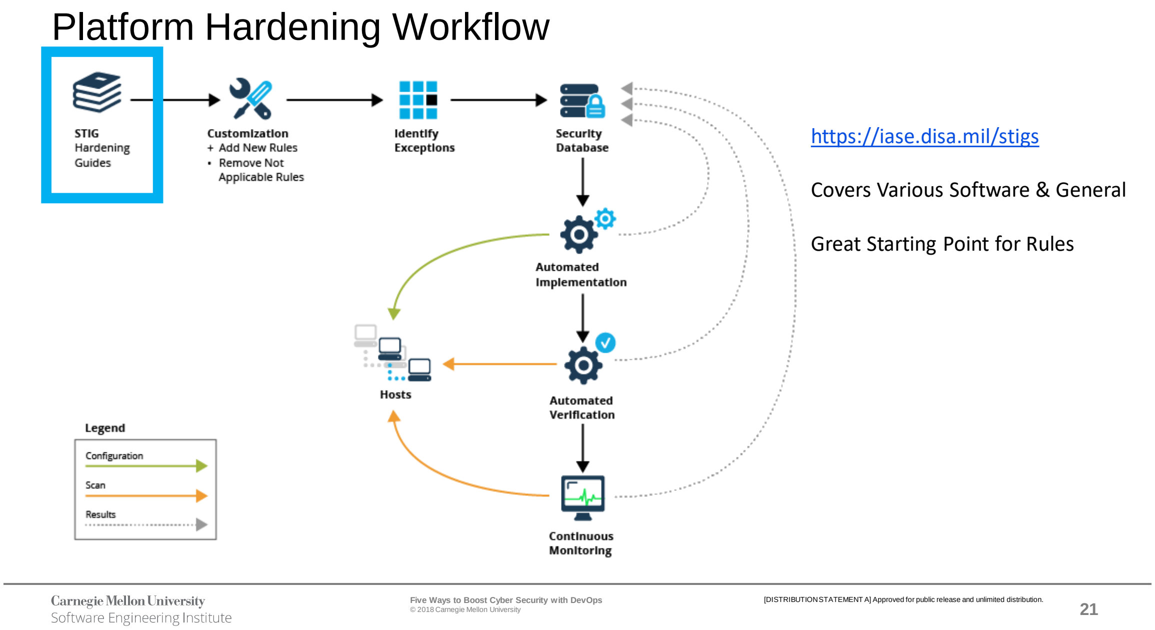
Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

20

**020 On the screen here, I have an overall diagram of a suggested workflow that you might take. First, we're going to go in detail through this, but basically, developing a list of security hardening rules that we wish to apply to our commercial off the shelf software or operating systems or open source software. We want to get a list of security hardening rules. We want to customize them and tweak them so that it fits in our environment. We want to develop the automation to apply those rules as well as un-apply those rules in the case where we need to lower-- we want to take a hardened box and lower the drawbridge in order to go into a maintenance mode or a fixit mode so that we can allow engineers into a system in order to fix it. We want to verify that that stuff worked. And after we're hardened, we want to continuously monitor the system to assure that nothing has diverged.

Platform Hardening Workflow



Platform Hardening Workflow

https://iase.disa.mil/stigs

Covers Various Software & General

Great Starting Point for Rules

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

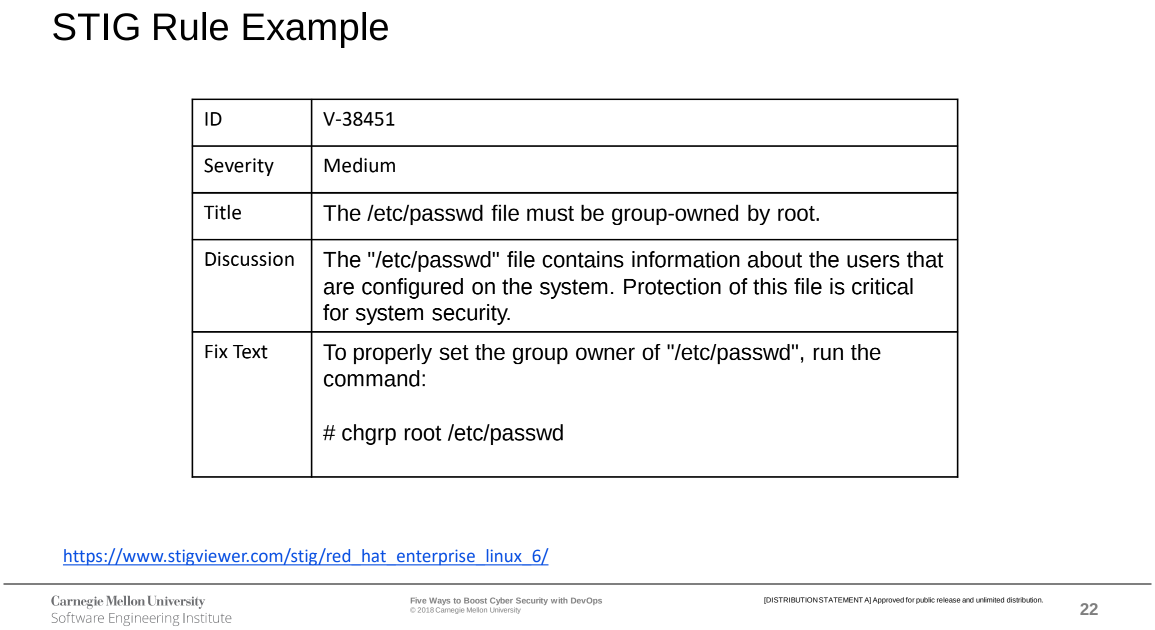
[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

21

**021 So, a great place to start is with STIG Hardening Guides from DISA. Up on the screen, I have the URL. These STIGs, they're security technical implementation guides. They cover various specific software that's mainstream enough for DISA to develop one of these and also general guidance if your piece of software doesn't fall within one of these hardening guides. It's a great starting point for rules.

And I'll show you what an example rule is.

STIG Rule Example



STIG Rule Example

ID	V-38451
Severity	Medium
Title	The /etc/passwd file must be group-owned by root.
Discussion	The "/etc/passwd" file contains information about the users that are configured on the system. Protection of this file is critical for system security.
Fix Text	To properly set the group owner of "/etc/passwd", run the command: # chgrp root /etc/passwd

https://www.stigviewer.com/stig/red_hat_enterprise_linux_6/

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

22

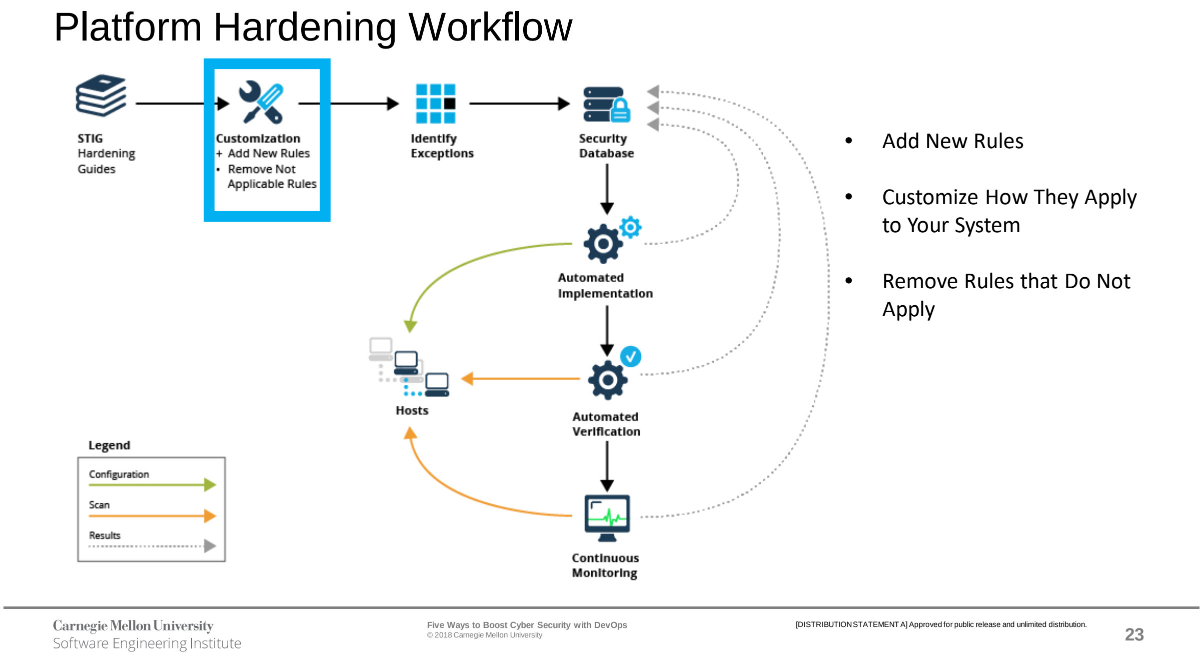
**022 Here's one for RHEL Linux 6 I think is where I got this. And it's just saying that the etc password file must be owned by root meaning that it can only be modified by the system administrator. And we see here that the fixed text, it gives the UNIX command line to implement this rule. I think what's important here is that, depending on your flavor of operating system, the fixed text might need to be tweaked, or we might be able to go above and beyond what the STIG is suggesting here.

Speaker 3: Yeah, with most of the Linux variants, your group and your root user are root and root. But if you go to the BSDs, like FreeBSD, root and the group for the root operating is wheel.

Speaker 2: Yeah so, the implementation's slightly different.

Speaker 3: It could be different.

Platform Hardening Workflow



Platform Hardening Workflow

STIG Hardening Guides

Customization
+ Add New Rules
• Remove Not Applicable Rules

Identify Exceptions

Security Database

Automated Implementation

Automated Verification

Hosts

Continuous Monitoring

Legend
Configuration
Scan
Results

- Add New Rules
- Customize How They Apply to Your System
- Remove Rules that Do Not Apply

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

23

**023 Speaker 1: And a quick comment just asking who's performing this platform hardening task. Is it a dev team, or ops, or security team?

Speaker 2: Now, I think that varies from org to org. I think what's really important is that those three groups are all involved. If you have expertise in implementing them on your dev team, I think it would make sense for them to actually do the work but to also have operations and security maybe perhaps peer reviewing network and being very intimately involved in how this process is orchestrated. So, depending on the skill sets within the organization, I think who does what can vary a little bit. But the important thing is that it's all three of those groups' responsibility. And it's somewhat blended.

Speaker 3: Yeah, the security-- with the security folks, they might be more apt to understand exactly what the focus is of the STIG hardening rule. And I've been in some organizations where the developers, they're pushing down the infrastructure as code to do the changes. I've also known other organizations that their ops team is working on those changes. And they're doing a lot more of the monitoring to verify that they're staying in place. So, it just varies on, like you said, skill set and who has the availability to do that and who's called out and saying that hey, I'm going to do that.

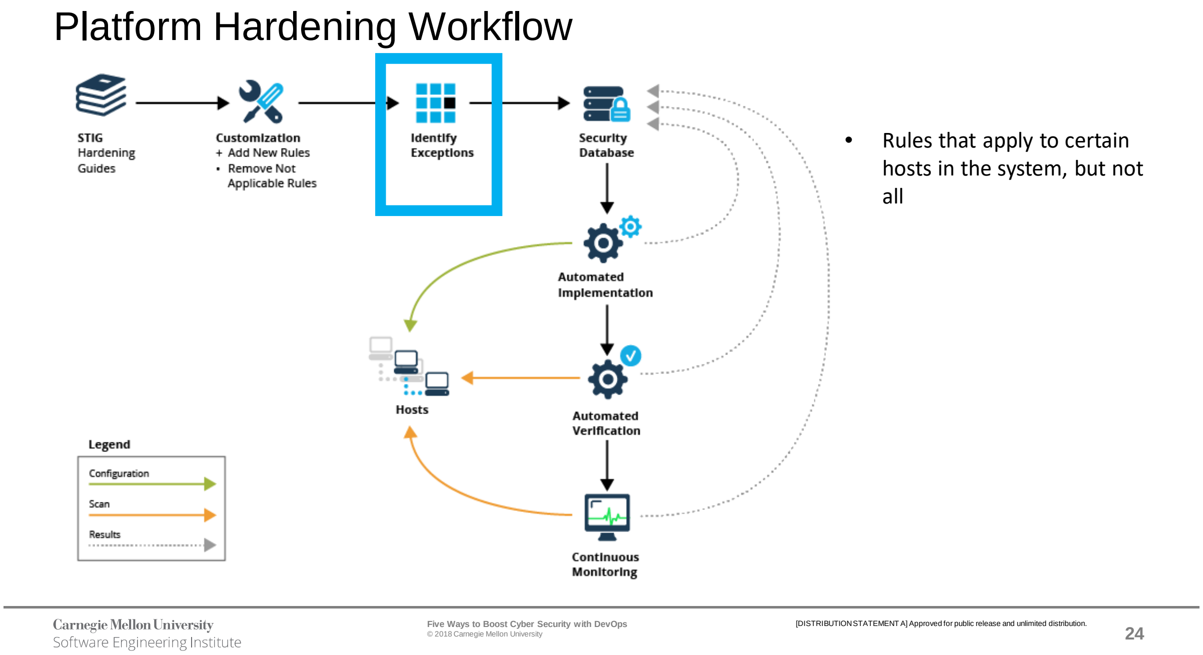
Speaker 2: Yeah, but just to rei-- but at the end of the day, it's a blended responsibility. And all three of those groups should be a stakeholder in that.

Speaker 3: Exactly, it should be something that everybody's saying, "Hey, this particular thing, I saw this. This is going to make security a little bit better."

Speaker 2: Right so, there's one task of coming up with the rules and customizing them like we have on the screen right now where we want to maybe add new rules that we dream up ourselves or tweak them on how they're implemented or maybe remove ones that aren't applicable to our system at all.

Speaker 3: Yeah, for example, looking through the rules for Red Hat 6, there's a rule for make sure you don't have TFTP installed.

Platform Hardening Workflow



Platform Hardening Workflow

- Rules that apply to certain hosts in the system, but not all

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

24

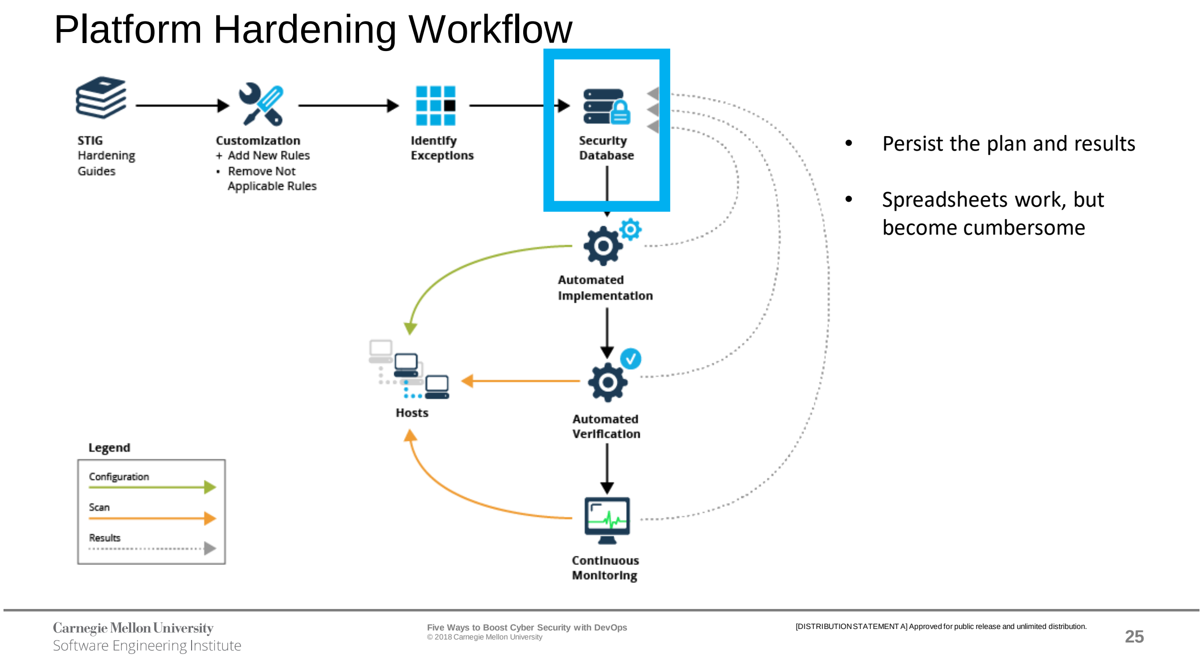
**024 Or if you do, make sure it's set so it's only in one directory. Now, if you have an organization that doesn't PXE boot any network devices, then you wouldn't have TFTP installed. So, it's just kind of common sense to say, "Hey, if you do do this, then you should probably implement this change." But not everybody's going to be able to do that or need to do that.

Speaker 2: Yeah, I think what you just mentioned sort of outlines how important it is for security, dev, and ops to be working in concert because if we have a security team who isn't very familiar with how the system's implemented, and they decree from on high that TFTP should be disabled, the only way we might-- and we're in a waterfall model, we might break things right before things go into reproduction unless these things are tested early when the collaboration is happening across those groups.

Speaker 3: Or maybe it is a general rule, and then now, you have system administrators. Now, they're forced to go around and manually-- wiring CDs into virtual machines to boot them up and get them configured. In any case you, could have unintended consequences.

Speaker 2: Yes, so, just to move on in talking about our platform hardening workflow, we want to identify exceptions. This step's sort of picking and choosing which hosts in our system deserve which rules where our TFTP server, for example, might need that service enabled. Or a webserver might need port 80 open, where other app servers might need that closed. So, this exceptions process is where we design which rules get applied to which host in the system. And if there's a rule that should not be applied, that should be understood and documented.

Platform Hardening Workflow



Platform Hardening Workflow

- Persist the plan and results
- Spreadsheets work, but become cumbersome

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

25

**025 Speaker 3: Exactly, you don't want to shut off all the ports in your web server because it's going to be difficult to serve content.

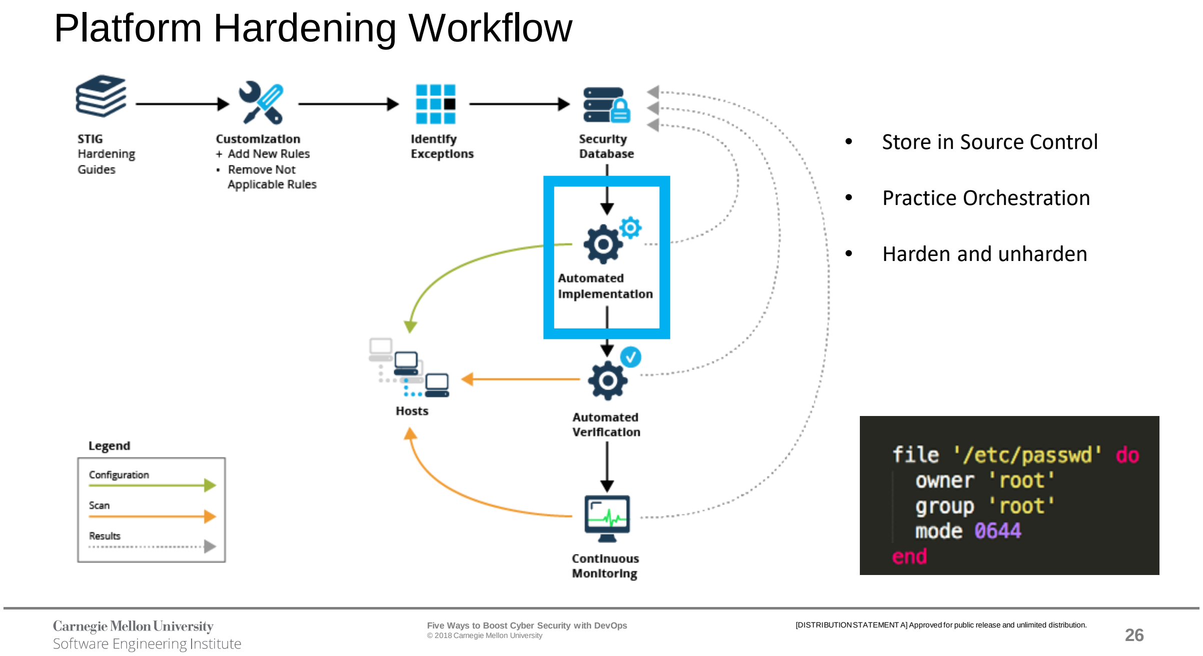
Speaker 2: But it would sure be secure though.

Speaker 3: Sure, yeah.

Speaker 2: Next up, we're saying persist everything in some sort of data store. It says security database. There's COTS and FOSS tools that do this. A lot of orgs that I've worked in use spreadsheets to manage this. I'm suggesting a secure-- some sort of relational database or something a little more beefy than a spreadsheet just because it becomes cumbersome. And reporting out of spreadsheets becomes a manual piece of work.

Speaker 3: The important thing about having your database is you can get really caught up in how you're laying these out in a COTS or off the-- FOSS tool. But the important thing is you don't take this process and turn it into this big giant organization where you have to hire twenty people to maintain everything because you can really get carried away about dependencies on this and this. And it just-- you need to keep what you have so it's useable but not unmaintainable.

Platform Hardening Workflow



Platform Hardening Workflow

- Store in Source Control
- Practice Orchestration
- Harden and unharden

```
file '/etc/passwd' do
  owner 'root'
  group 'root'
  mode 0644
end
```

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

26

**026 Speaker 2: For sure. The next step in our workflow is automated implementation. This is the infrastructure as code and where the DevOps-y tooling comes in. Some just ground rules, you want to store this in source control along with our application code for our system that we're hardening. We want to practice the orchestration of applying these hardening rules early and often because some of these hardening rules, they do some pretty hardcore things to a system. And me, I know engineering can do its best effort in designing which rules can be turned on. But we want to make sure that we're testing this often because these rules can break something. And we also want to keep in mind whenever we're implementing that we might need to apply rules and undo rules for different modes of the system. For example, if we need to upgrade our system, we might need interactive login to be applied.

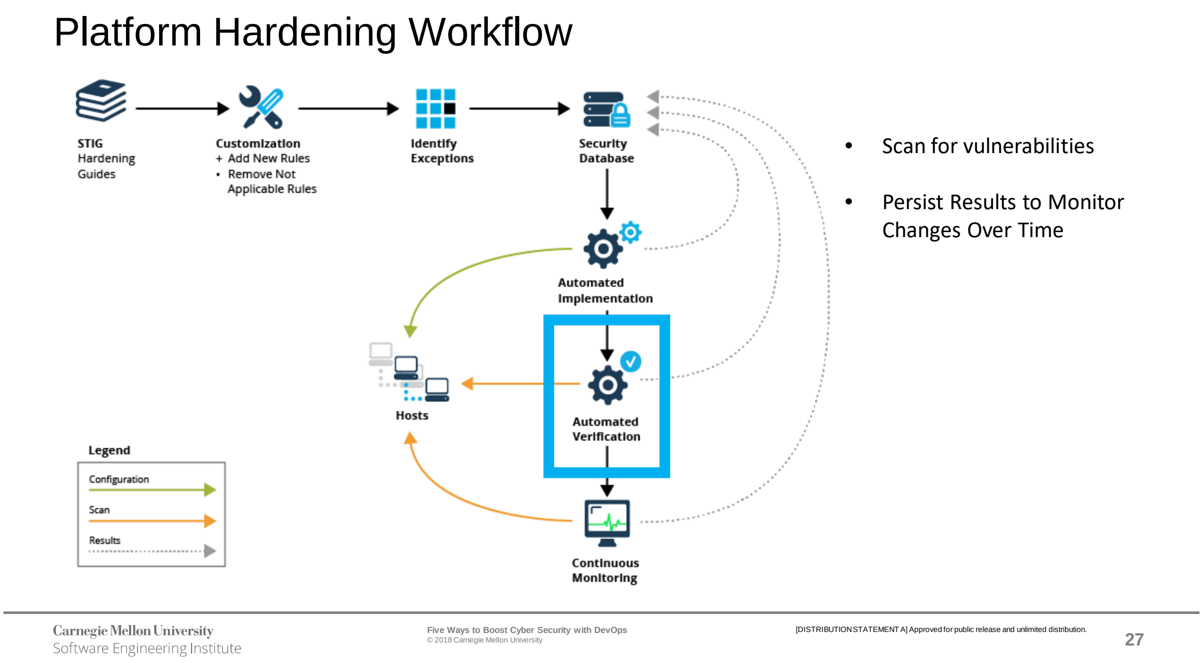
Speaker 3: Exactly, and if you're using Yum or Aptitude to do system upgrades, sometimes those tools are expecting a specific permission or file in a certain location. So, they could break your update. The other possibility is some UNIX systems allow you to set file immutability so even root can't change a file. So, if you go to apply all these rules, and you have a lot of files set to immutable, then all your updates are going to fail, and you can't change anything.

Speaker 2: Which is a bad, bad way to be.

Speaker 3: Yes.

Speaker 2: And just there the lower right-hand corner, there's just a little snippet of some chef code, which is a common, very popular infrastructure as code tool on how we would implement the STIG rule that we show where we're setting the etc password file to be owned by root.

Platform Hardening Workflow



Platform Hardening Workflow

- Scan for vulnerabilities
- Persist Results to Monitor Changes Over Time

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

27

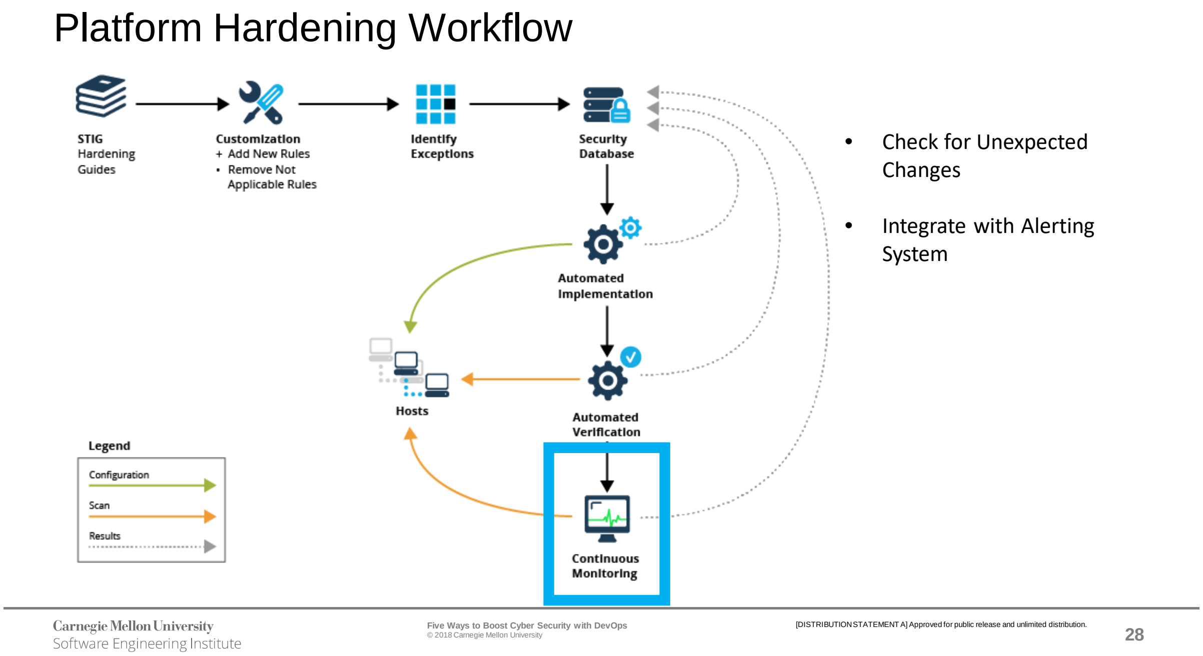
**027 The next step in our workflow is scanning for vulnerabilities. This is sort of our check. And there's many required tools that a lot of organizations require to use to scan their boxes to assure that there's no known vulnerabilities. And the DevOps-y note here is that we want to persist these results, the output from these tools, so that we can look at how they change over time and how they evolve.

Speaker 3: Yeah, it's-- a lot of these tools, the first thing that comes to my mind is directory webserver, directory transversal, where you try to go outside of the webserver route. And that's a common thing that you want to make sure that you can't do that and download--

Speaker 2: You have a bad day if you can do that.

Speaker 3: Yeah, download the shadow file or the password file.

Platform Hardening Workflow



Platform Hardening Workflow

- Check for Unexpected Changes
- Integrate with Alerting System

Carnegie Mellon University
Software Engineering Institute

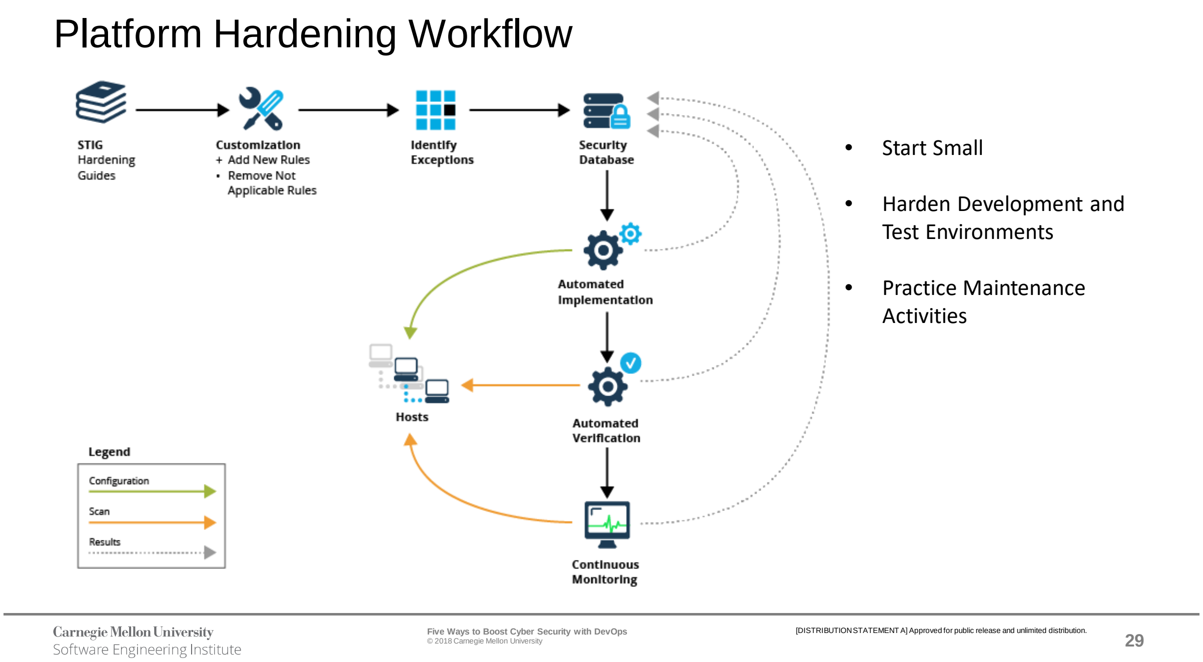
Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

28

**028 Speaker 2: The next step in the workflow is to have some sort of system for continuous monitoring of the system from a security perspective. This is something like a Tripwire to-- so, we take a snapshot of a subset of our system, mainly the file system or main processes running, if there's any deviation from that behavior, then we are able to trigger an alert so that we are aware that something unexpected might have happened.

Platform Hardening Workflow



Platform Hardening Workflow

- Start Small
- Harden Development and Test Environments
- Practice Maintenance Activities

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

29

**029 And for this process, my sage advice on implementing one of these is to start small. I've been in organizations where they send a team of twenty developers off to the races creating automation to apply hundreds and hundreds and hundreds of rules only to find that the way that they were implemented didn't work quite right in their environment. So, this led to tons of rework. So, my advice is to start off with just a few rules and run them the whole way through this lifecycle to get learnings in the shift left learnings that you might have on maybe-- you might find that the way we have these implemented, we're not able to un-harden them, things like that.

Speaker 3: Yeah, sometimes when you're doing this, it's kind of like doing drywall. When you're doing this, less is actually more until you figure out what you're doing.

Speaker 2: Yeah, exactly, just so we don't waste time. Very important to harden as much as possible our development and test environments so that we don't have any surprises in our ops-like environments before we go to production. And again, practicing the maintenance activities because the un-hardening can be quite tricky.

Speaker 3: Yes.

Speaker 2: So, Shane, is there any question from the audience?

Speaker 1: We do have a couple of questions I think--

Five Ways to Boost Cyber Security with DevOps

Five Ways to Boost Cyber Security with DevOps

Application Security

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

**030 Relative to this section. Rick

was asking, "What recommendations would you offer as a skill set for security folks to work with the DevOps team?"

Speaker 3: I think the security folks having a good overall grasp of security folks. I know some of the folks I've worked with before, they know a lot of really good security principles. And maybe it's something you sit down, and you review STIG rules because if you're in the government sector, you're probably going to have to implement some sort of STIG rule. If you're in the commercial sector, maybe you're not required to do that. But maybe it's just a good practice to look through and see. A lot of the security folks go, and they collaborate with different companies. And they get together, and they talk about what's their primary issue. So, sometimes, it's just a lot of collaboration, understanding the attack vector you have today.

Speaker 2: Yeah, I guess my comment on that is the more general people are, the better. If you have security folks with a little bit of system knowledge of the system that they're securing, the more we have that, the better. If they're able to configure their home network, their home router or--

Speaker 3: Sure, yeah.

Speaker 2: The better to have some hands-on knowledge of these things. That will just increase their ability to collaborate with how these rules are implemented and be able to contribute in that way.

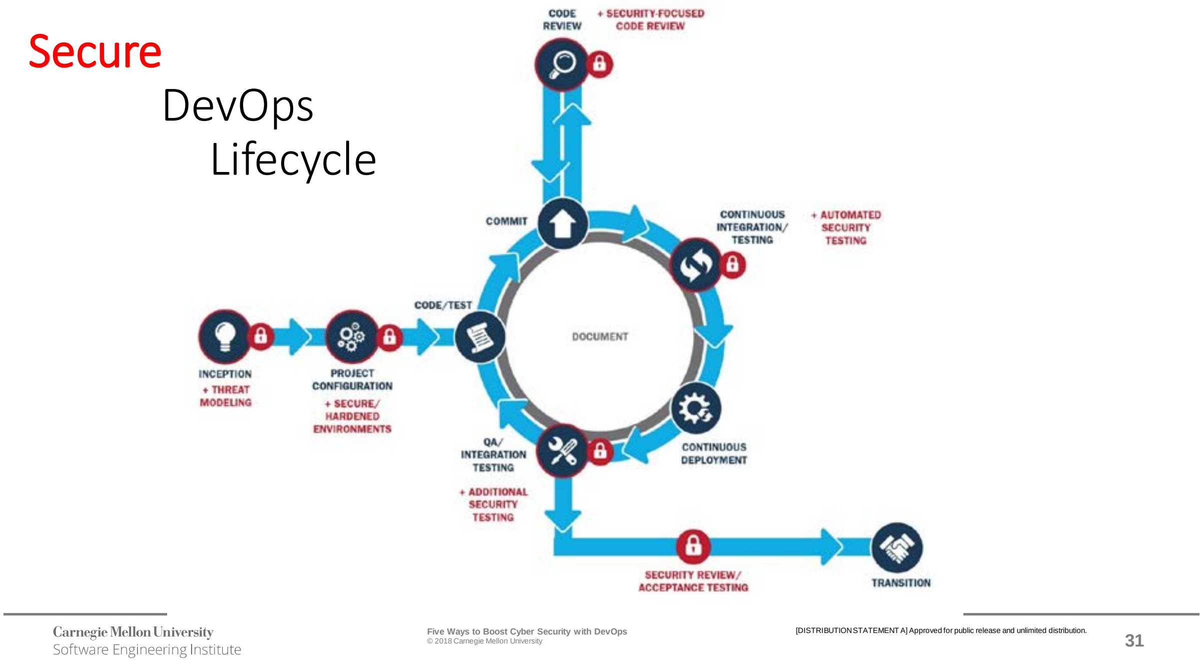
Speaker 3: Yeah, going on that, I formerly worked with a sysadmin turned to security, and it really made things great because you could talk system internals, and you get to talk security. And it was a lot easier to put two and two together.

Speaker 2: Right, and even if you don't have those skill sets, getting those people's work closely aligned and in the same room so that they can cross-moginate and share those skill sets. The more we can maximize that, the better of course.

All right, well, moving right along to our next way to boost cybersecurity with DevOps is AppSec.

Speaker 3: AppSec.

Secure DevOps LifeCycle



Secure DevOps Lifecycle

CODE REVIEW
+ SECURITY-FOCUSED CODE REVIEW

COMMIT

CONTINUOUS INTEGRATION/ TESTING
+ AUTOMATED SECURITY TESTING

CODE/TEST

DOCUMENT

INCEPTION
+ THREAT MODELING

PROJECT CONFIGURATION
+ SECURE/ HARDENED ENVIRONMENTS

CONTINUOUS DEPLOYMENT

QA/ INTEGRATION TESTING
+ ADDITIONAL SECURITY TESTING

SECURITY REVIEW/ ACCEPTANCE TESTING

TRANSITION

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

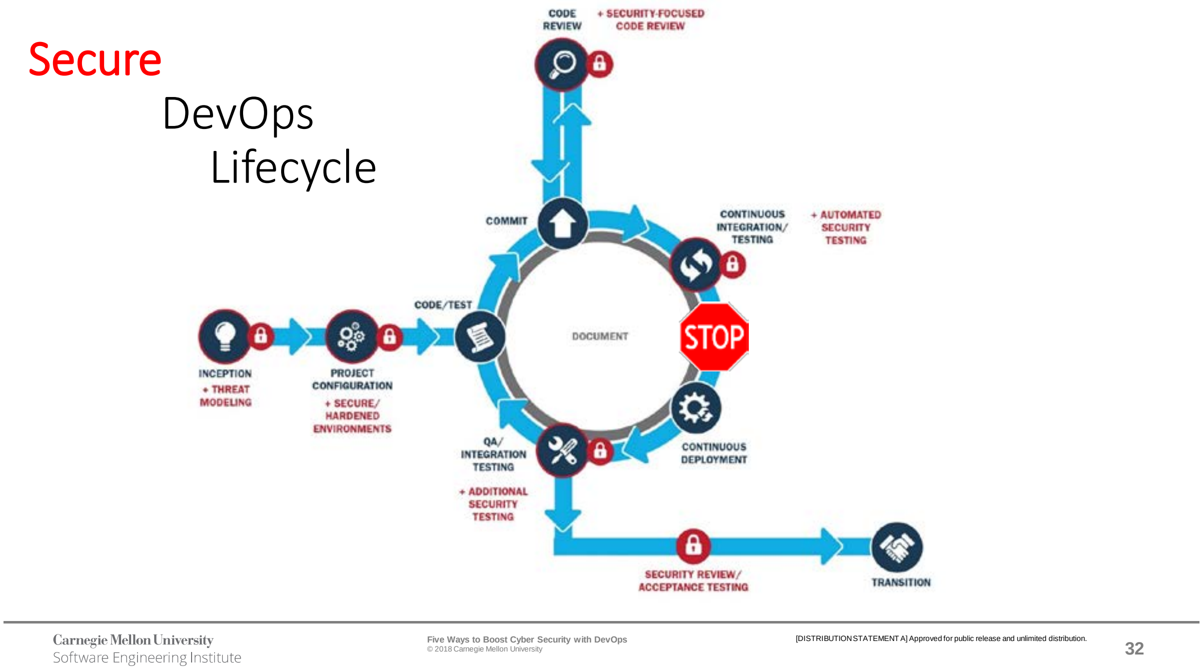
31

**031 Speaker 2: Here on the screen, we have the SEI Secure DevOps lifecycle. And just pointing out the different places in the FDLC, where we can use-- apply automation and modernization to increase security. At project inception, very important to focus on threat modeling. During our project configuration stand up, secure hardened environment like what we mentioned in the last section, whenever we're going though coding, having a security focused code review process, perhaps with some automation sprinkled in there so that whenever a developer wants to change the developer's definition of done, to make sure that there's an automated system in place, to make sure that there's some sort of code review, preferably security focused code review if the change warrants that. And may perhaps systems in place to make sure that they're going through a checklist and the results of that review are persisted.

Speaker 3: Sure, and along those lines is that this is also great where you're running both your unit and functional test to ensure that the actual story your working on is completed through the functional test, and you know it provides the functionality that you're intending. And that goes along with security. And it's all a great place to scan and check to make sure that you're not in a state where you don't want to be.

Speaker 2: Yes, indeed. Yeah, and what you're mentioning leads us into continuous integration testing, functional testing, perhaps building our software out, running unit tests, running static code analysis, and deploying the software to an ops-like environment and running automated security testing on the system stood up.

Secure DevOps LifeCycle



Secure

DevOps

Lifecycle

CODE REVIEW
+ SECURITY-FOCUSED CODE REVIEW
COMMIT
CONTINUOUS INTEGRATION/ TESTING
+ AUTOMATED SECURITY TESTING
CODE/TEST
DOCUMENT
STOP
INCEPTION
+ THREAT MODELING
PROJECT CONFIGURATION
+ SECURE/ HARDENED ENVIRONMENTS
QA/ INTEGRATION TESTING
+ ADDITIONAL SECURITY TESTING
CONTINUOUS DEPLOYMENT
SECURITY REVIEW/ ACCEPTANCE TESTING
TRANSITION

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

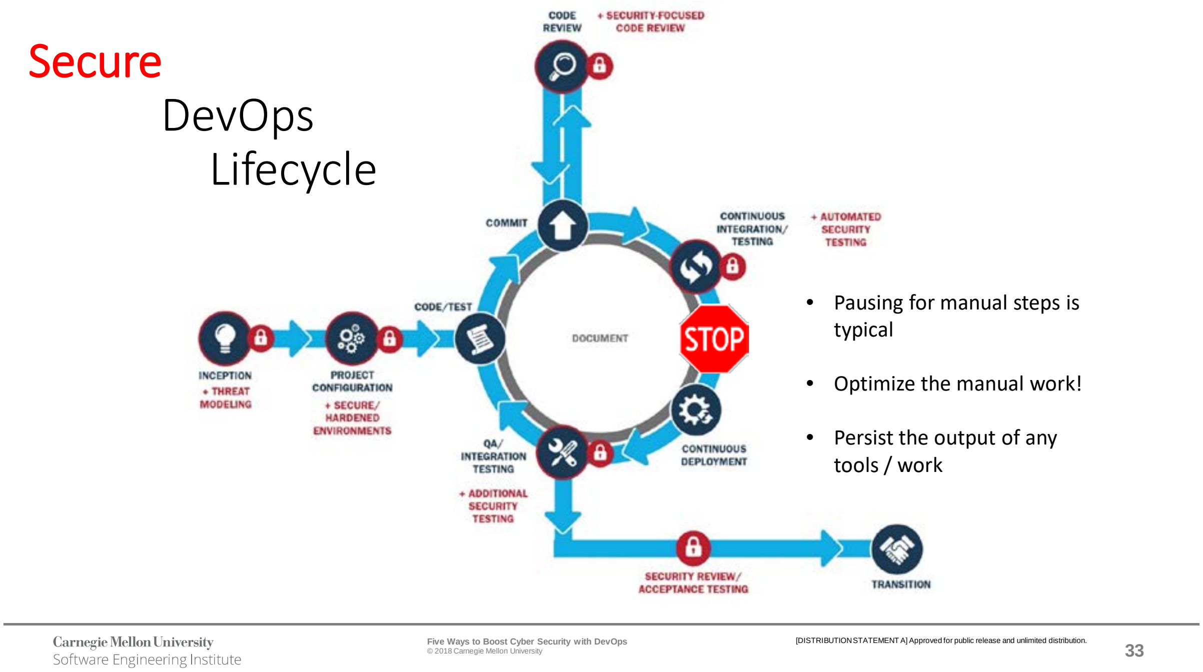
32

**032 I think it's very important to note that we have all this automation. If we have security testing in our pipeline, do we still need human intervention? Absolutely.

Speaker 3: Sure.

Speaker 2: We do. And it's very common for there to be a stop sign in our automated build environment, software factory, where somebody has to go in and take a look-see-loo.

Secure DevOps LifeCycle



Secure DevOps Lifecycle

CODE REVIEW
+ SECURITY-FOCUSED CODE REVIEW
COMMIT
CONTINUOUS INTEGRATION/ TESTING
+ AUTOMATED SECURITY TESTING
CODE/TEST
DOCUMENT
STOP
INCEPTION
+ THREAT MODELING
PROJECT CONFIGURATION
+ SECURE/ HARDENED ENVIRONMENTS
QA/ INTEGRATION TESTING
+ ADDITIONAL SECURITY TESTING
CONTINUOUS DEPLOYMENT
SECURITY REVIEW/ ACCEPTANCE TESTING
TRANSITION

- Pausing for manual steps is typical
- Optimize the manual work!
- Persist the output of any tools / work

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

33

**033 I think what's important here is to recognize that pausing for manual steps is typical, and what our goal is to optimize this manual work. So, before we release our software, if we have to do an application-- and appsec security evaluation by a security team, perhaps automate the process of getting those folks all the information they need in front of them like credentials, server locations.

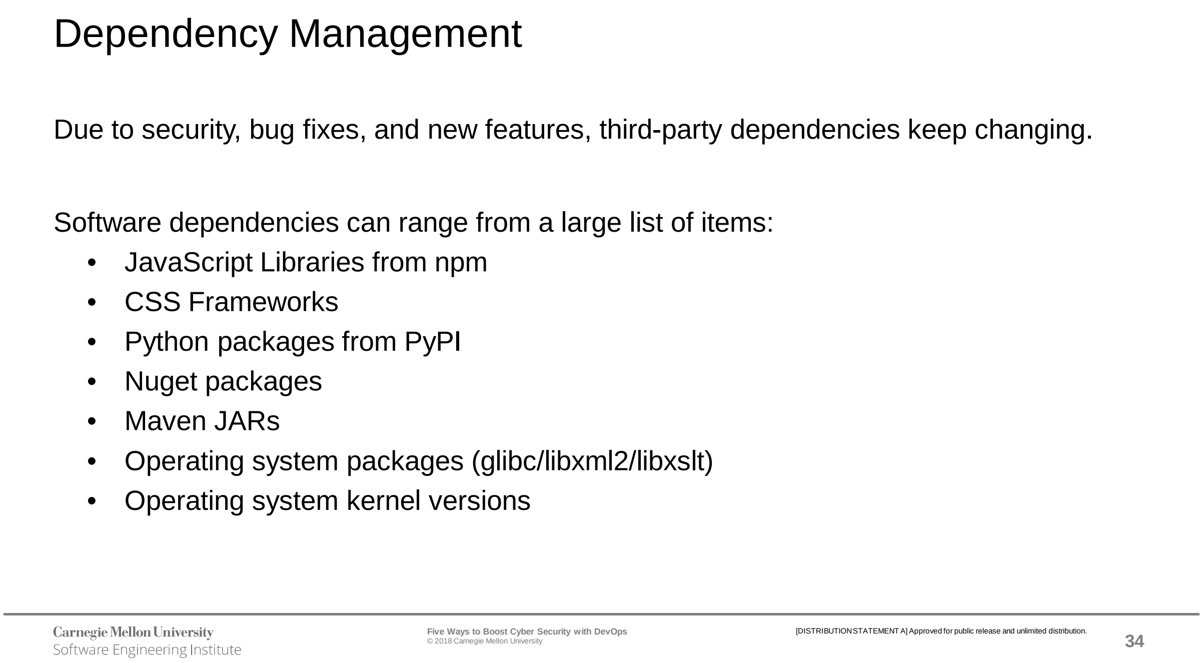
Speaker 3: Server logs, logging.

Speaker 2: Automate the part so we can provide to them a package of data that they need to do their manual step. Perhaps they run tools on it. We want to persist the output of any tools that they run so that we have a history there and maximize the speed that these manual processes can take place.

Speaker 3: And this is natural in the development process. The goal is to catch it before it goes into production. And so, this might seem like a big-- a blocker, but in the grand scheme of everything, it's actually going to benefit everybody in the long run.

Speaker 2: Absolutely.

Dependency Management



Dependency Management

Due to security, bug fixes, and new features, third-party dependencies keep changing.

Software dependencies can range from a large list of items:

- JavaScript Libraries from npm
- CSS Frameworks
- Python packages from PyPI
- Nuget packages
- Maven JARs
- Operating system packages (glibc/libxml2/libxslt)
- Operating system kernel versions

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

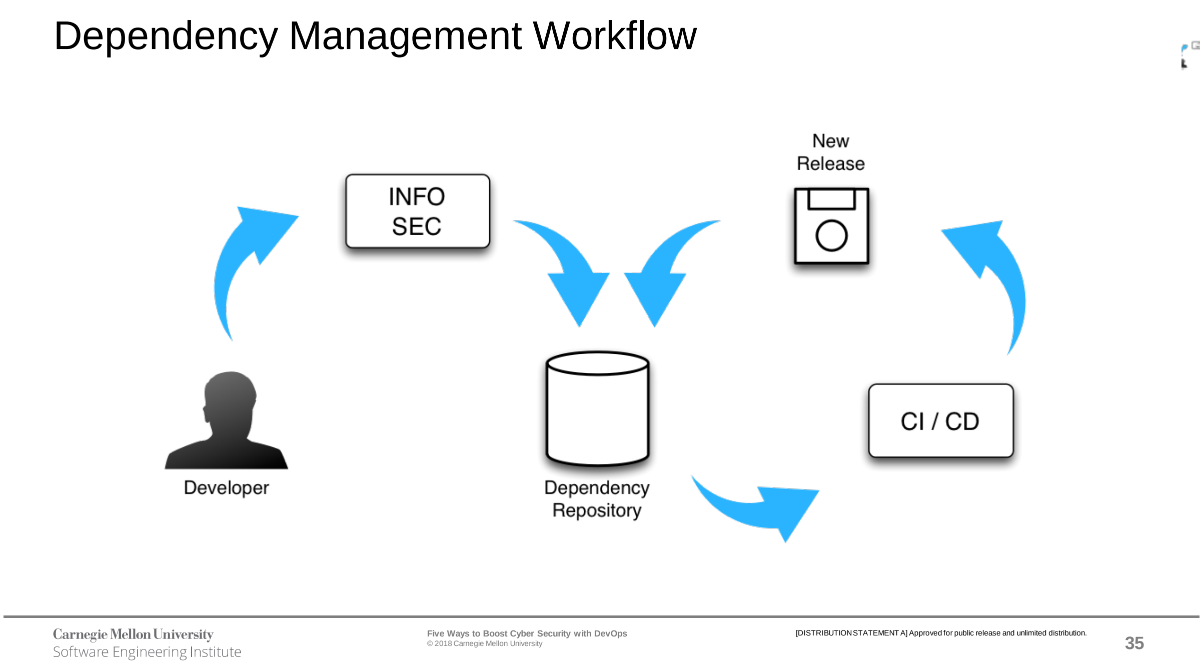
34

**034 Next were going to talk about some dependency management.

Speaker 3: Sure so, nowadays when you build software, you're not just writing all these lines of code that do everything in your piece of software you need. For example, most people aren't writing their own object relational mapper. So, most software nowadays, is you're getting probably seventy to eighty percent open source software. And you're writing glue code, you're writing your front end, you're writing your business logic. So, you don't want developers just to be pip installing any package they come into contact with or downloading every JavaScript framework that you can download on the Internet and putting it into your application because there's like four hundred million JavaScript frameworks. So, that's just bad.

So, the idea of dependency management is it gives you-- you need to have some sort of sheriff to say, "Hey, let's verify that these dependencies or something that we want or something that's not bad or have security issues." So, part of doing the management is--

Dependency Management Workflow



Dependency Management Workflow

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

35

**035 We have InfoSec involved in

the process. We have developers, and then we have a dependency repository. So, the goal with the repository is you take-- the developer says, "Hey, I want to use SQLAlchemy in my project because it's going to make a great object relational mapper. So, InfoSec will take a look at it. Maybe they have tools to scan Python code. Maybe they have different ways that they're vetting their software. And if they say yes, we can use this package, then they can put it into a dependency repository that will be able to be accessible by your build server and your CI server. So, when you're building a release, it will pull own that dependency from your repository and not allow the build server to pull it from an outside location. And then once it's in that repository, you keep it there for as long as possibly it's needed. That way that you always have this available. And these-- dependency repository is kind of a generic term. There's different ways you can do that.

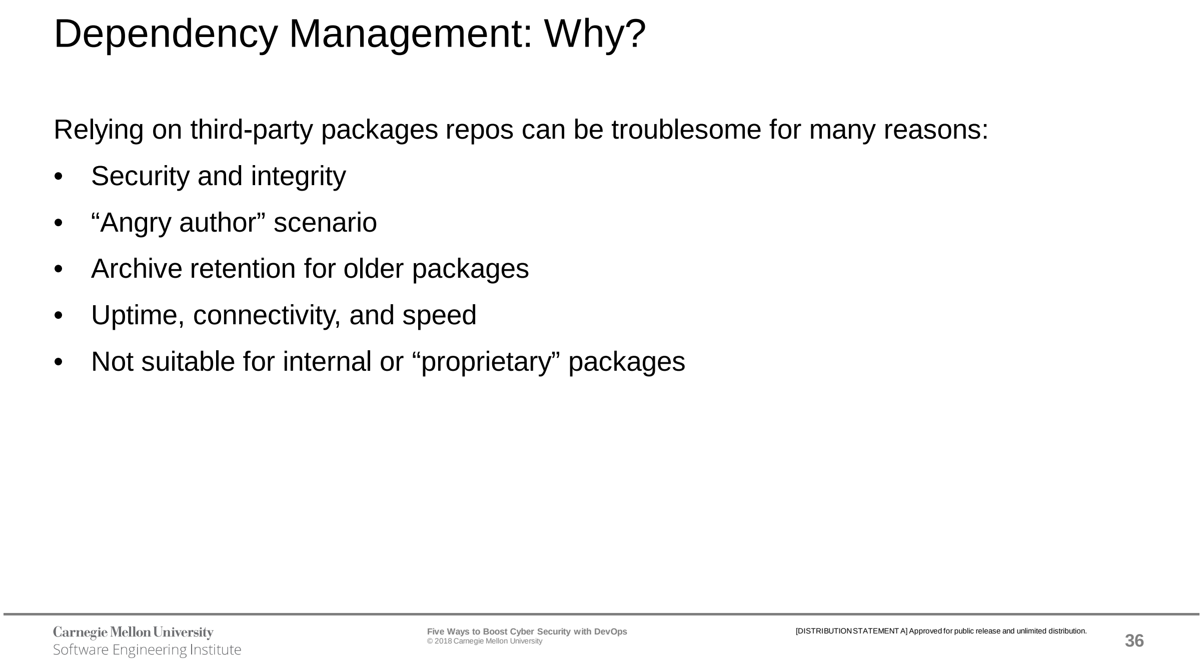
Speaker 2: Yeah, it can be something as simple as a file-- some directory full of files. Or it could be something like there's different open source tools that provide this that-- or you can have your own internal PyPI repository or your own Maven Central in order so that all the build tools can sort of talk seamlessly. All you have to do is point them at your internal repository full of quote unquote blessed artifacts that have been screened by security.

Speaker 3: Yeah, some of them it all depends on what these purposes are serving. You can go as fancy as having a full repository that has all-- a nice interface and automated. Or you can just have-- it's easy enough to set up a patch to your nginx server and point at it direct with Python files to serve as your PyPI repository.

Speaker 2: Yeah, the important thing is to avoid having developers, your build server, reach out to the Internet and download tons of dependencies during your build process because never know what you're going to get.

Speaker 3: Sure.

Dependency Management: Why?



Dependency Management: Why?

Relying on third-party packages repos can be troublesome for many reasons:

- Security and integrity
- “Angry author” scenario
- Archive retention for older packages
- Uptime, connectivity, and speed
- Not suitable for internal or “proprietary” packages

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

36

**036 So, part of why we're saying you don't know what you're going to get is you have different things that can happen. First, you always have the security issue. You have what I like to call the angry author issue where you have somebody who's trying to do-- who's just mad about something. Then you always have packages that you want to hold for eternity. And then, people don't think about this, you always have up time connectivity. If you're trying to force out a build, and your Internet's down, that's going to limit your connectivity to pull down a package. So, if it's internal--

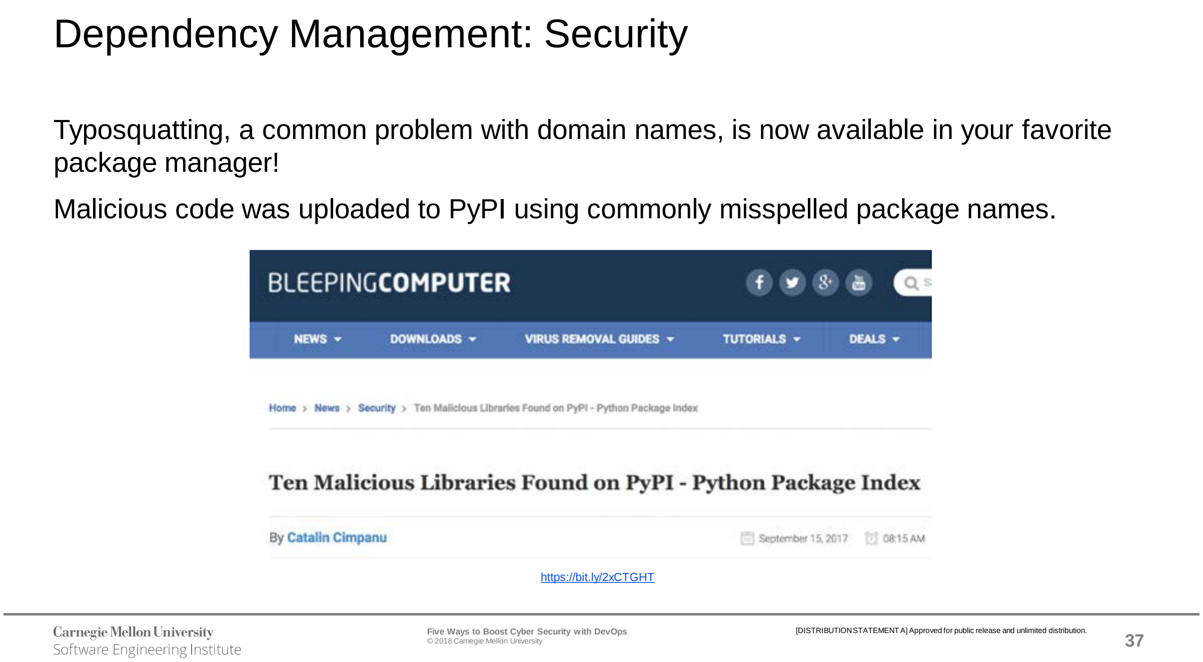
Speaker 2: Right.

Speaker 3: That's going to save you time and probably money.

Speaker 2: Right so, lots of benefits here. And I think we're going to go into detail in some of these because they're very interesting stories that we wanted to share with the audience on different things that can happen if you don't manage your dependencies and assure supply chain hygiene.

Speaker 3: Sure.

Dependency Management: Security



Dependency Management: Security

Typosquatting, a common problem with domain names, is now available in your favorite package manager!

Malicious code was uploaded to PyPI using commonly misspelled package names.

Home > News > Security > Ten Malicious Libraries Found on PyPI - Python Package Index

Ten Malicious Libraries Found on PyPI - Python Package Index

By Catalin Cimpanu September 15, 2017 08:15 AM

https://bit.ly/2xCTGHT

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

37

**037 So, security is always a good topic to focus on when you're looking at dependencies. Just as a simple example, everybody's run into typo squatting where you go to a domain name, and you spell it wrong, and it goes to some--

Speaker 2: I never-- I'm such a good typer, that's never happened to me.

Speaker 3: You end up at a link farm or some random place. But this can happen with your dependencies. They've found this out on PyPI. People had uploaded libraries with slightly different names. For example, I can never spell SQLAlchemy unless I look it up. So, somebody had flipped some of the words around.

Speaker 2: So, they switched two letters around?

Speaker 3: Yeah, exactly. You would download the bad package. And some of these packages were phoning information home to foreign actors. And that probably is not a good thing, I would say.

Speaker 2: No, very, very bad. So, just an example of why it's very important to have security screen all these little bits that comprise our software to make sure that we're dealing with-- not relying on a developer not doing-- maybe having a typo.

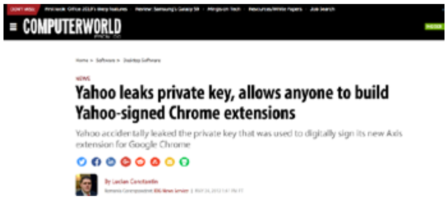
Speaker 3: Sure.

Dependency Management: More Security

Dependency Management: More Security

External packaging is signed, so it is okay, right?

In a perfect world, yes. In today's world, maybe not!

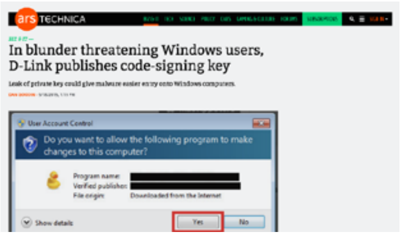


Yahoo leaks private key, allows anyone to build Yahoo-signed Chrome extensions

Yahoo accidentally leaked the private key that was used to digitally sign its new Axis extension for Google Chrome

By Lucian Constantin

https://bit.ly/2DMpIzZ



In blunder threatening Windows users, D-Link publishes code-signing key

Leak of private key could give malware easier entry onto Windows computers.

User Account Control

Do you want to allow the following program to make changes to this computer?

Program name: [redacted]
Verified publisher: [redacted]
File origin: Downloaded from the Internet

https://bit.ly/2IFHKHH

- D-Link
- Yahoo!
- Linux Mint

Beware of hacked ISOs if you downloaded Linux Mint on February 20th!

I'm sorry I have to come with bad news.

https://bit.ly/2pxsFzY

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

38

**038 And the other thing with security is we get packages now with Windows and probably Mac and most of the operating systems that are cryptographically assigned. So, that should be-- make sure that all the packages are perfectly fine, right?

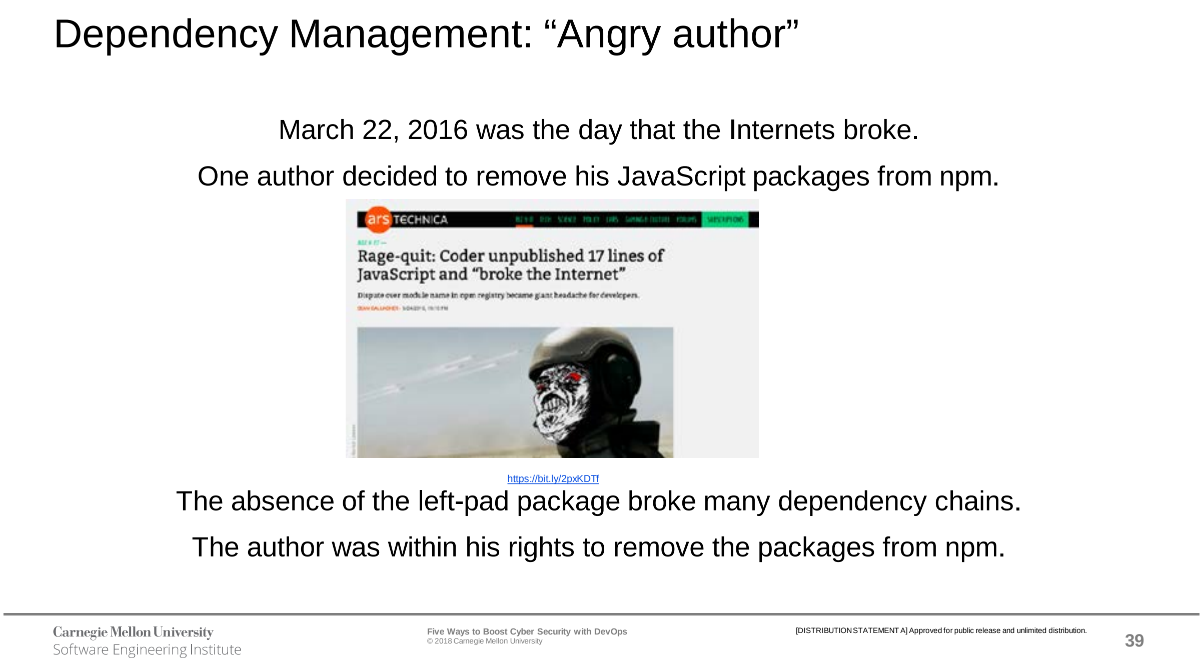
Speaker 2: So, just to clarify for the audience, this means they're signed so that if something is signed by Microsoft, I know that that binary downloading came from Microsoft. And I can trust it.

Speaker 3: So, normally, the signing process, you have a public key that everybody recognizes, and you have a private key that you tuck in a safe. And you only take it out and use it when you're making a signature on an application. Well, in the case of D-Link and Yahoo, somehow their key got out on the Internet. So, that means that I could pick up a package and say, "Hey, I'm Yahoo," or Aaron could say, "This firmware is from D-Link." And nobody would have any better idea because it's all built in and signed and said hey, there's a certificate authority that says yeah, this is D-Link for sure.

And then Linux Mint was kind of like, they were victims of somebody that hacked into their server and replaced their ISO with a malicious one and then changed the signature that you could compare. So, people were downloading this for a short period of time and installing it. And guess what it did? It was phoning home information to somebody.

Speaker 2: And the victims were none the wiser.

Dependency Management: “Angry author”



Dependency Management: “Angry author”

March 22, 2016 was the day that the Internets broke.

One author decided to remove his JavaScript packages from npm.

Rage-quit: Coder unpublished 17 lines of JavaScript and “broke the Internet”

https://bit.ly/2pxKDTf

The absence of the left-pad package broke many dependency chains.

The author was within his rights to remove the packages from npm.

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

39

**039 Speaker 3: So, this is another thing is what I like to call the angry author as I was saying before. So, there was a gentleman who had an npm package that was published and--

Speaker 2: Very widely used by the way.

Speaker 3: Sure, and some of his other packages happened to coincide with a trademark. And he was asked to give up one of his packages to release it to the trademark owner, which it's not like a-- it's kind of one of those, "Hey, this is a nice thing to do. We don't want to sue you." But then he just got-- they got-- he didn't want to do it, so npm said, "We're pulling your package." And then he says, "You know, I'm pulling all my packages off npm" One of his packages happened to be the left pad package, which basically all those four million JavaScript frameworks happen to use. So, everybody's build system broke until they figured out different-- they tried different ways to fix it. And they didn't work because the versioning was wrong. So, finally, they had to republish the package to fix it until they can-- everybody could update their code. So, this is another reason why to have your dependencies in house because those aren't always going to be on the system. Somebody could make them disappear. And so, you don't want to be just victim of an Internet flame war.

Speaker 2: Yeah, it assures the repeatability of our build process because we're-- all the dependencies to build our software, at one point and time, are under our control and under our influence so that we-- things like this--

Speaker 3: Don't happen.

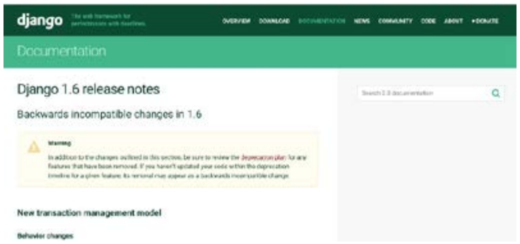
Speaker 2: Shouldn't happen.

Dependency Management: Maintenance Mode

Dependency Management: Maintenance Mode

Eventually, project development is completed, leaving a finished product. Two years in the future, it has to be deployed to another server. Do you know where all of your dependencies are stored, because they aren't available from the original external repo!

- Open-source projects generally move very fast or very slow.
- Slow moving projects will have the same version for years as they are feature complete.
- Fast moving projects will often release major version every year, and the API will not be compatible with the previous major release.



Django 1.6 release notes

Backwards incompatible changes in 1.6

Warning

In addition to the changes outlined in this section, be sure to review the deprecation plan for any features that have been removed. If you haven't updated your code within the deprecation timeline for a given feature, its removal may appear as a backwards incompatible change.

New transaction management model

Behavior changes

https://bit.ly/2pzvobx

**040 Speaker 3: Same thing with maintenance, years down the road, hopefully you don't work on a project that never ends. Eventually, there's hopefully some end in sight. Eventually, that project's going to be running on some lonely server in a dusty closet. And then somebody's going to do a security scan and say hey, we need to update some part of this application because it's insecure. So, you pull out the source code from a dusty closet, and you realize that you don't have a particular package because it's now version twenty-seven. You started off with version two point oh. So, you try to integrate a new package, and it doesn't work because all the APIs changed. And part of this dependency management is to ensure that you have a consistent history of packages, that way you can build a package and make sure at any time in its history that it's ready to go and can be deployed out the door.

Speaker 2: Yeah, I've seen instances where if we were forced to upgrade a single dependency, that dependency depends on other things that need to be upgraded. So, it's sort of a domino effect that we end up biting off a lot more than we can chew and being forced into a DLL hell situation where, by not having an older dependency and upgrading, we end up having to upgrade a whole lot more than we have the time or budget for.

Speaker 3: Yeah, and it could be as much as just dependencies, or you might have to rebuild all your dependency stack if your custom compiling open source tools. So, it's a good idea just to keep those things in your back pocket so you have them.

Five Ways to Boost Cyber Security with DevOps

Monitoring

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

**041 Speaker 2: So, Shane do we have any new questions from the audience?

Speaker 1: One that came in earlier, and we may be able to save it for the end if it's not the appropriate time, I'll just read it off. This one's asking, "How can we establish a platform for collaboration amongst all stakeholders including security?" Like I said, it's from earlier in the talk. So, we may be able to review it at the end. But if there's a quick answer, we can take it now.

Speaker 3: I'll leave you to handle that one.

Speaker 2: Yeah, establish a platform for collaboration, I think-- well, I think you can establish source control repo-- have all the tools available so that everybody's working off of a common workbench. Have a method of communication between these teams, wiki pages, document repositories, all the tools that we listed earlier, ideally would be-- if an organization has all of those tools, I think that's perfect platform for collaboration between security and development.

Speaker 1: And a quick one in regard to security aspects built into the lifecycle. What does that do to cost? Obviously, there's costs involved there. Is this preventative stuff that you're going to pay for later? You're preventing different outbreaks, but what's the cost up front? Is it--

Speaker 2: With all DevOps things, and I think that's one of the big barriers of adopting DevOps is because DevOps sort of forces us to tackle challenges much earlier than traditionally. And in DoD projects that we follow, in programs that are trying to adopt DevOps midstream, that's one of the struggles. For developers in a lot of DoD programs that are set up to be waterfall, development gets credit for doing their work on a best effort basis, meaning as soon as a developer is done with their code, and it works on their machine, they can take credit for that work, and they're done. And then months down the road will it go through formal testing, perhaps, and security things. So, what DevOps forces us to do and what we're talking about is shifting all those left so that they're happening much earlier. So, there may be a greater cost because we're doing more things earlier, but for the benefit of the long-term. All these DevOps things, it's a very long-term focus. And it's one of the tr-- it's a very common challenge to try to quantify the ROI early on because we're just doing so many more things early.

Speaker 3: Yeah, and it's-- especially it's like if you take over an old project that didn't have any testing, automated testing at all. You can't just go and write five million unit tests for code that has a million lines. It's just not feasible. But you can start out small, like if you're making a change, try to put tests on your new changes, try to test things that are around your change and affecting. And that's kind of the way with DevOps is you don't-- you can't always go full tilt into doing DevOps and doing security scans. But the good thing about it is is after you do an investment, it's like money in the bank. Like you don't have to-- once you make a security scan to scan your application or test it, you don't have to do that anymore. It's just-- it's free every time you run it. The only thing you're paying for is CPU cycles. So, you're not going to spend a ton of time after you spent your upfront time.

Speaker 2: Yeah, just to like build on what you're saying, Doug, one of the big challenges to-- for organizations to adopt, convert from an old school waterfall mindset, is the inertia of the organization where this is the way we've always done things. And this is our traditional roles that individual contributors have and the jobs that they have. There's tremendous inertia. And it could be-- it's something that's sort of unquantifiable, but it's definitely there. But on the good side with DevOps, if we start out small, we can gather inertia in the form of skill sets and, for individual contributors, seeing that their jobs are being done better and that they become easier after they automate some of the arduous, tedious work. And I've found that that inertia takes over, and things start to speed up and get easier as we have these core capabilities established on performing security scans. So, if you can automate one pen test, it can be very difficult to get that one out the door. But as each security test is added on to our pipeline, it just becomes faster, easier, and cheaper to implement. You get economies of scale.

Speaker 3: Yeah, it's very similar when you're setting up CI. Everybody can build, go into your development environment and build a software on your laptop, but to have a build server, it takes a little bit of time to have it set up and working quite the way you want it to. But after you have it, it's there. You expect it to be there. If you have to make a tweak to it, you're not redoing the whole entire environment. You're just changing build parameters or how the builds flow into the system.

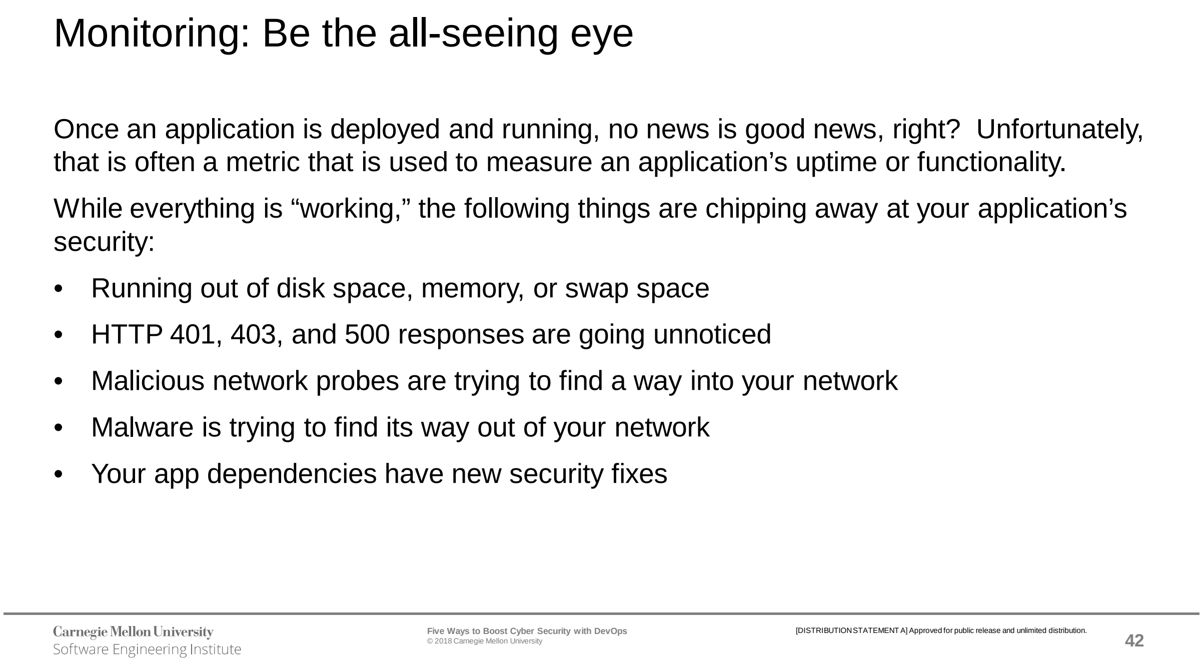
Speaker 2: Making small incremental changes, and if that build server goes away and is down for some reason, then people miss it very, very, very badly.

Speaker 3: Oh, yes.

Speaker 2: All right, well we're headed in the home stretch here. Our last way to boost cybersecurity with DevOps is through monitoring.

Speaker 3: Monitoring.

Monitoring: Be the all-seeing eye



Monitoring: Be the all-seeing eye

Once an application is deployed and running, no news is good news, right? Unfortunately, that is often a metric that is used to measure an application's uptime or functionality.

While everything is "working," the following things are chipping away at your application's security:

- Running out of disk space, memory, or swap space
- HTTP 401, 403, and 500 responses are going unnoticed
- Malicious network probes are trying to find a way into your network
- Malware is trying to find its way out of your network
- Your app dependencies have new security fixes

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

42

**042 So, part of--

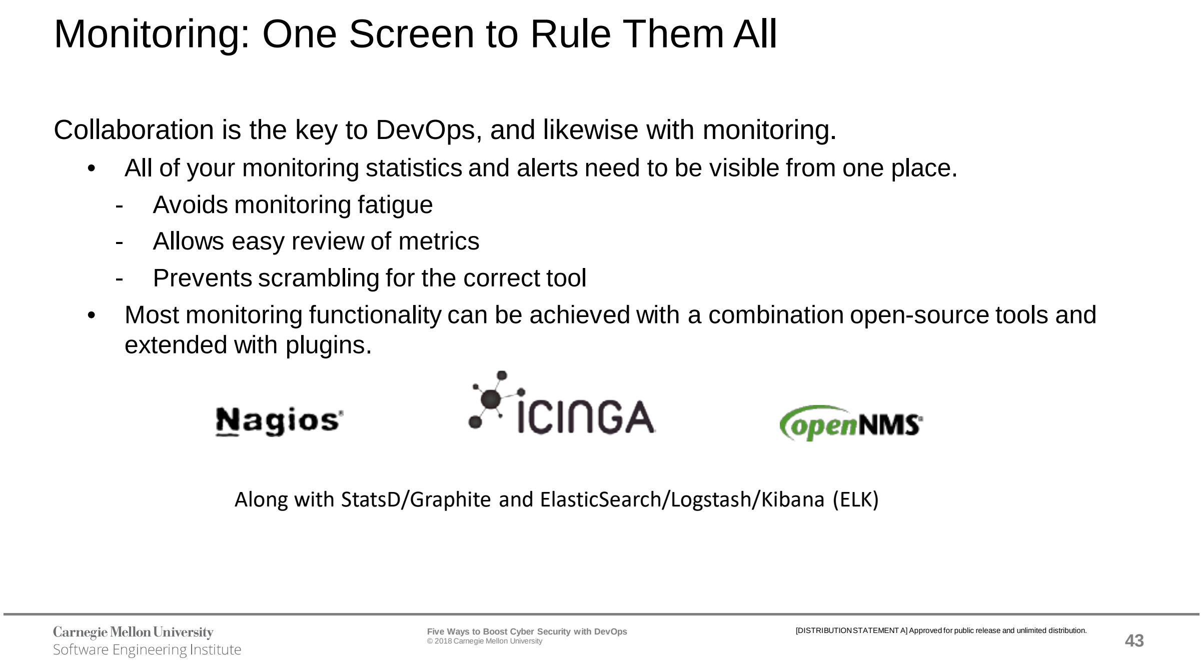
Speaker 2: The Eye of Sauron.

Speaker 3: Yes, or the eye on top of the pyramid on the dollar bill. So, you kind of-- the idea with monitoring is you want to keep-- you can't always see everything. But it's the idea is everything is visual. And everything is available for you to peruse when you're trying to inspect the system or understand what's happening in the system. And there's different things you can monitor. There's different quantities you can monitor. You can monitor-- just like we were speaking with security, you probably want to start out with monitoring small because you can go really-- you can go really-- put a lot of time into monitoring. You can monitor like every aspect of your application. But it's kind of like a gradual-- a little bit-- less is more when you start out.

So, with monitoring, there's a lot of different things you can monitor. Like I like to start out and say that monitoring your disk space, memory, and swap is probably one of the basic, most important.

Speaker 2: And amazingly, I've seen different government programs that burn millions and millions of dollars where they don't even have a simple shell script and a cron job that does a DF minus H and emails somebody the results. If they would have had that, they could have avoided a lot of downtime because they were caught not monitoring their disk space.

Monitoring: One Screen to Rule Them All



Monitoring: One Screen to Rule Them All

Collaboration is the key to DevOps, and likewise with monitoring.

- All of your monitoring statistics and alerts need to be visible from one place.
  - Avoids monitoring fatigue
  - Allows easy review of metrics
  - Prevents scrambling for the correct tool
- Most monitoring functionality can be achieved with a combination open-source tools and extended with plugins.

Along with StatsD/Graphite and ElasticSearch/Logstash/Kibana (ELK)

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

43

**043 Speaker 3: Yeah, and the monitoring point is you want to have one place you can look at the monitoring kind of like your collaboration. You want to have one place that will like wire into your collaboration system. You want to be able to know that when you're looking at a screen that you're going in one particular location that has all your monitoring metrics.

Speaker 2: And we're seeing the whole picture in one place that we can make appropriate business decisions.

Speaker 3: Exactly.

Speaker 2: To react.

Speaker 3: And there's different ways. There's different tools that are open source like Nagios. Probably, everybody's heard of that. Icinga, it's like a Nagios for open NMS. Then there's NC has created an open source StatsD/Graphite. And then you have your ELK stack. And just having these tools together, you get easy metrics. You don't have to scramble to figure out what you're looking for, what you have to grab. You can tune your metrics to avoid monitoring fatigue. And it just makes life a lot easier when you're trying. This is especially more important with operations. But development needs to know this as well because if they're trying to react to a security issue or a problem, you need to know what you're looking for, and you need to know metrics on how this happened, what the quantity was, and things like that.

Speaker 2: Just to hit back on your monitoring fatigue is where we have- we're overwhelmed with-- whoever the watcher is, they're overwhelmed with so many false positives they-- you open up their email, and they just see hundreds of emails. And they don't know which alerts are real or not. So, that's what monitoring fatigue is is where we're not seeing the signal through the noise.

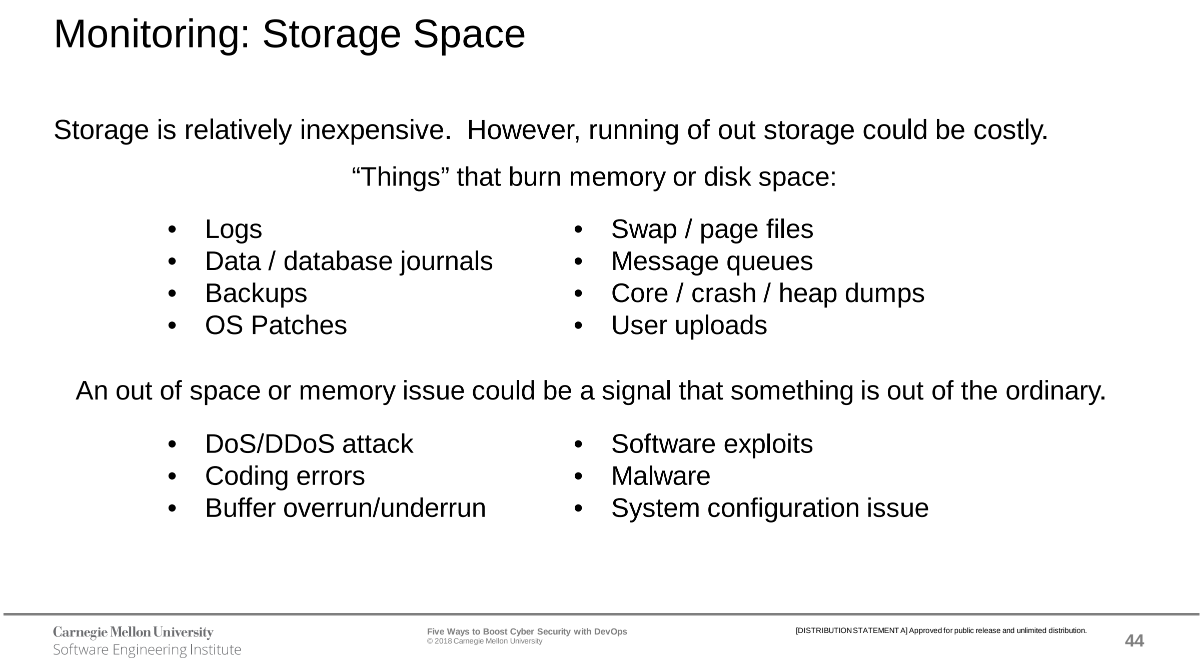
Speaker 3: Yeah, and a lot-- especially with like disk monitoring, if you're monitoring backup space on a database server, you could be using space and all of a sudden, you're at ninety percent, and all of a sudden, you drop to fifty percent because your archive logs are rolled back into the primary backup, depending on how your database is. So, you're like oh, I don't need any space anymore. Then it just happens again. And then you can see this trend and say oh, I really need to add more disk space to my database.

Speaker 2: Yeah, I think that's a very important part with monitoring is to under-- whenever we institute monitoring, and this applies not only to monitoring our systems, but also monitoring our development. We have very few-- very rarely do we have any idea of what good looks like whenever we're looking at this data. And it's very helpful to look for trends in the data and see if we're going from fifty percent to seventy-five percent overnight, we know that that's probably a problem where something's filling up a log or something to that respect.

But yeah, and I just wanted to-- it also applies to if we're monitoring our DevOps environment. How many builds per day, mean time to-- MTTR, mean time to recovery of a pipeline, number of unit tests, number of failed tests. Out of gate, most orgs don't have any idea of what they should be looking for. And it's a learning process on what numbers they should be looking at and what thresholds make sense because just blindly looking at MTTR and demanding that that's one hour. That's not necessarily a good thing.

Speaker 3: No, not at all.

Monitoring: Storage Space



Monitoring: Storage Space

Storage is relatively inexpensive. However, running of out storage could be costly.

“Things” that burn memory or disk space:

- Logs
- Data / database journals
- Backups
- OS Patches
- Swap / page files
- Message queues
- Core / crash / heap dumps
- User uploads

An out of space or memory issue could be a signal that something is out of the ordinary.

- DoS/DDoS attack
- Coding errors
- Buffer overrun/underrun
- Software exploits
- Malware
- System configuration issue

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

44

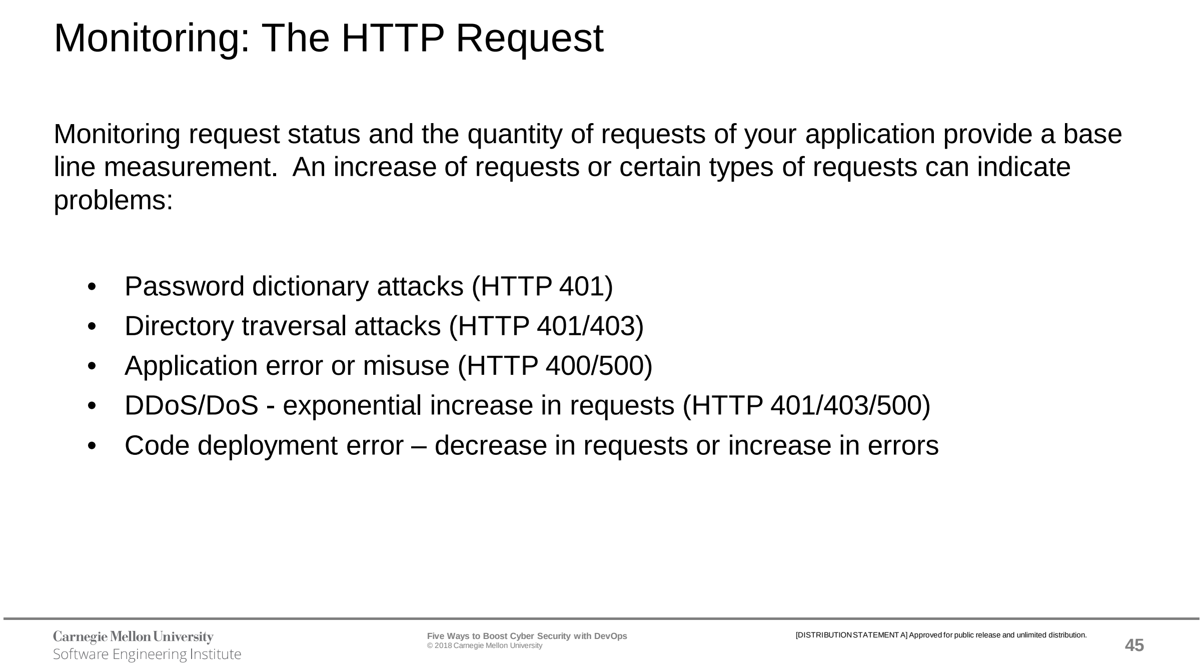
**044 So, part of this storage space is you have a lot of different occasions where logs are. And logging and running out of logs and storage could be just you need more space. Or you could have somebody performing a DDOS attack. You could have coding errors that are like filling up your disks. Malware, you could have exploits, buffer overruns, underruns, just a ton of things that could cause this. Some are security related, and some aren't necessarily security related.

Speaker 2: Hopefully, more often than not, they're not security related.

Speaker 3: Yes.

Speaker 2: But we cannot ignore that this is a very strong signal that there might be a security incident occurring.

Monitoring: The HTTP Request



Monitoring: The HTTP Request

Monitoring request status and the quantity of requests of your application provide a base line measurement. An increase of requests or certain types of requests can indicate problems:

- Password dictionary attacks (HTTP 401)
- Directory traversal attacks (HTTP 401/403)
- Application error or misuse (HTTP 400/500)
- DDoS/DoS - exponential increase in requests (HTTP 401/403/500)
- Code deployment error – decrease in requests or increase in errors

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

45

**045 Speaker 3: And the other thing that can be a good monitoring aspect is your-- if you have a web application, you have-- you monitor your web requests. A lot of times, if you get a lot of 401s, not everybody remembers their password on the same try, but you might have a couple of those a day. But if you all of a sudden have fifty thousand failed requests, that could be an indication that you have issues with somebody doing a dictionary attack.

Speaker 2: I think that's why it's so important to have unified data so that we're-- we have visibility of what changed in the system. If we're seeing a spike in these invalid requests, is it a bug? Or is it an indication of something happening externally?

Speaker 3: Exactly, and the other thing that's important is like error 500s is the same thing with security is that could be a bug that could turn into a security issue.

Speaker 2: Absolutely.

Speaker 1: So, we are at noon right now. So, a couple people may have to drop off. Can we wrap up with some parting shots or some conclusions from you guys before we have to sign off?

Speaker 3: Sure, I think we're probably-- we're good if you want to-

Speaker 2: Yeah, the main takeaways from this are number one focus is collaboration and communication. The tooling is the sexy thing, and the automation and tooling is the thing that people can grasp onto. But at the core of DevOps and security is to increase people talking and working together with aligned goals. Start small, start with thin horizontal slices to try out things the whole way through the lifecycle in order to shift left learnings. And I don't know.

Speaker 3: Yeah, you don't want to bite off an entire-- you don't want to automate all the things. Just like if you get a project, you don't want to sit down and right five million unit tests because-- you can do that. But it's probably-- you're not going to see any benefit out of it.

Speaker 2: Limited value add.

Speaker 3: Exactly, so the idea is if you're starting out a project fresh, it's really a good time to implement DevOps because you can do each step of the way. If you're trying to integrate it, you need to take the snowball effect. You need to start small and then gradually increase and get more involved, everybody more involved with the project.

Speaker 2: Yeah, and also very important to apply the scientific method and run experiments, make changes, and define a way to measure the outcomes of changes and determine whether or not something new that we're doing, whether-- determine-- have some sort of measuring stick to determine if the new things that we're doing are working or not because there is no secret sauce to DevOps. It varies so much depending on the context that experimentation is paramount.

Speaker 3: Yeah, and it could a difference between organizations. It could be a difference between applications. It could be a difference between even people. Some people learn in different ways. Some people have different preferences. So, it's just a matter of finding out what works in your environment.

Speaker 1: Doug and Aaron, thank you very much, excellent talk. We appreciate your time today.

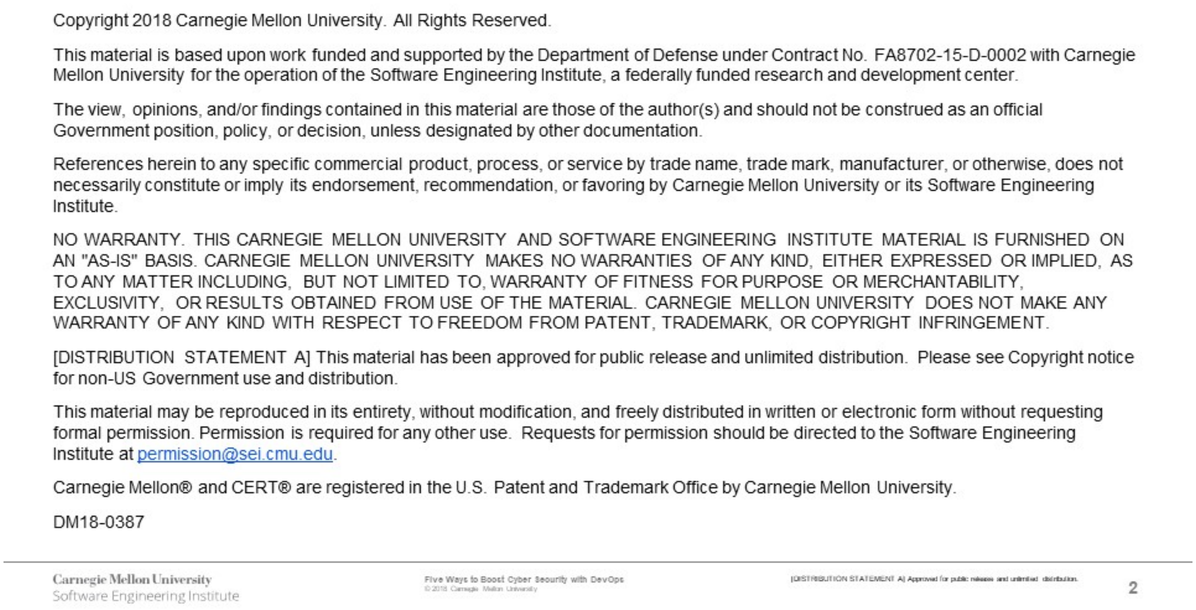
Speaker 3: Thank you, Shane.

Speaker 1: And just to close out,

just a reminder for everybody to please complete the survey upon exiting today's presentation. And the link to that survey is at the top of the chat tab within your platform. And just one final plug, for those of you in the D.C. area, the SEI's hosting a symposium next week with the theme of Agile and DevOps. So, if you found some value in today, I know Aaron is speaking at that next week. So, we will send out a link there. And the best part of that symposium, it is free to attend. So, we'll send out some more information. Hopefully, you can attend that as well.

And finally, our next webcast will be May 9th. And our topic will be block chain, your questions, our answers. Thanks, everyone. Have a great day.

Copyright 2018.



DM18-0387

Carnegie Mellon University
Software Engineering Institute

Five Ways to Boost Cyber Security with DevOps
© 2018 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

2