Red Hat SELinux

Walsh: **Dan J. Walsh, Lead SELinux Developer**

1. Introduction and Overview

Walsh: Hello, my name is Dan Walsh. I'm the lead SELinux developer at Red Hat. I've been working on SELinux for about three years now, and I'm going to give you a talk, basically an introduction to SELinux, and some of the past, present, and future.

2. 0-Day Exploits

Now, when software is delivered, the first day you install that software is sort of the 0-day. The problem is that hackers or crackers can basically find vulnerabilities in the software, and at some later point that vulnerability gets known to the wider community, and at a later point then it gets back to the vendor and the vendor is able to create a security patch for the exploit, and then that will get sent out to the customers and then finally the vulnerability gets plugged and installed on that.

But what do you have to protect yourself during the time between when you install the software for the first time and the vulnerability gets fixed by the vendor? At Red Hat we decided to work on a new product called SELinux, or a new way of running the operating system called SELinux that would help you protect against those 0-day exploits.

3. Security misconceptions

Where is the good stuff on a computer system? The good stuff on a computer system is not in the system part, it's in the user's home directories. That's where my credit card information is; that's where my top secret documents are; they're in my home directories. I want to protect user space from systems space. The bad guys are out in system space. They've broken into the system through some application that I'm running on the system through some network connection on the system that have broken in, so how do I protect my user space against the system space?

People are always trying to protect route. Okay, how do I protect a user from becoming root? Well, a lot of times what the hackers are now after is not necessarily a root account, what they're actually after is the ability to become, get onto the machine and then use the machine as a spam forwarder. So you just need a regular account in order to be able to send an email out of the system. So if I break in to your machine and I can forward email through it, I don't care about getting to that, so we're also trying to prevent people trying to break into the system and being able to spam mail out.

So my question to the audience would be, which is more important, a user that can read the etc/shadow file, or an application, say my Apache server, is able to read my home directory and get access to my credit cards? In my opinion, the Apache web server getting access to my credit cards is a lot more important to me.

4. What is SELinux?

So what is SELinux? SELinux is your last line of defense on the system. SELinux is mandatory access control: that's all it is. It isn't encryption; it isn't any of the other security things; it's mandatory access control. Current UNIX systems and current, actually, Windows systems, they use what they call discretionary acts of control. What that means is that the user has discretion or has the ability to change access capabilities on every file on the system that he owns. The problem, also, is that every application that he runs also has the access, so if I, as a user, run, say, the Mozilla application, that Mozilla application has full access to my home directories and, again, has access to my credit card information. So every application that I run has the same access that I, the user, has. Now, if you take that to the systems side, every application that runs as root has full access to what the root user and root has control over the entire system. So when I run a service on a machine and it's running as root, and that machine gets hacked into, that application gets hacked into by an attacker, he now has full access over the entire machine.

So what SELinux does is it will basically lock that down, eliminate that capabilities. So root compromise, even if I get to a full root compromise on a system, I can lock it down to only being able to do what that application was designed to do. So if they break into my Apache web server, I can lock it down so that all they can do is what Apache was designed to do.

So, a little history about SELinux. SELinux has been developed for about 15 years by the National Security Agency. When they were designing NSA, they designed a flexible architecture called Flask--Flask security architecture--and a couple of design think goals was to separate the security policy from the actual enforcements. So they wanted to have a separate policy, a separate rules database, so that they had some flexible way of defining security goals of the system. What they wanted this to be, was also be transparent to the applications and users of the system. So they didn't want to have to rewrite every single application on the system, and they wanted it to be somewhat transparent to the user, so they wouldn't even know that they're running an SELinux system. The real goal was to basically remove the full power of root to a breakdown of power of root so it doesn't have as much power, and applications running on the root would be locked down.

5. Where should you run SELinux?

Next slide is to talk about where should you run SELinux? Obviously any application that runs on the Internet is going to be subject to far more attacks when you're not protected by firewalls, so SELinux becomes much more important when you're running a box on the pure internet. As you move inwards to the boxes, you come to the corporate network, and these would be your firewall boxes, your application server, stuff that's in your demilitarized zone or your DMZ. Basically any application that now has a connection to the internet, even if it's coming through a firewall, that application could be vulnerable to attack, so SELinux should be running the app.

We also believe that you should be protecting with SELinux on the internal network. There's been several studies that show about 80% of all attacks come from inside your companies, so an employee of the company decides to attack, say the payroll system, you want to protect your infrastructure applications on the machine. Once we get away from infrastructure, it's basically on the individual servers. Those machines are vulnerable. Again, that's where I have my good stuff, that's where I have my credit cards, so I want to protect all the way down to the desktop.

Now, with SELinux, we're able to run the range of different security policies because, as we talked about before, the flexible architecture, so you can run a different security policy, say on your laptop, that you might want to run on your firewalls. So you can run different types of security architectures on these platforms.

6. SELinux Key Components: Kernel

There are three components that we're going to talk about what makes up SELinux. The major one is the kernel. SELinux was first developed, as I said earlier, on the flask architecture but NSA quickly realized that the goal of getting this into a mainstream operating system, was that people weren't going to go out and adopt their operating system, so they decided to move it to Linux. NSA still wanted to get better uptakes. So NSA came together with Red Hat probably about three and a half years ago and wanted to talk to us about perhaps adopting SELinux into Red Hat's operating system. People feel that Linux is more secure than Windows environment, so we wanted to do as much as possible to make it approval being correct.

So, at the same time that NSA wanted to get into a mainstream operating system, Red Hat wanted to improve their security, so we had sort of a perfect storm, and we got together with NSA to bring SELinux out to the mainstream. First thing we did is to work with the NSA kernel developers to get the main modifications accepted into the upstream kernel. As many of you might be aware, Red Hat prefers to have all modifications work with the upstream kernel and not have lots of patches, so one of the requirements we had with NSA was that the patches had to get into the mainstream kernel. So about three years ago, the 2.6 kernel adopted most of the patches and SELinux into it.

One of the main problems with, when they tried to get it accepted into the kernel, was the upstream maintainers of the kernel felt that it was too SELinux-specific. So they wanted to add all the excess control hooks into the operating system, but they didn't want to be purely SELinux-centric, so they forced them to basically break the SELinux central part of the code into a global module, and they put the hooks into the mainline kernel. So that this developed what they call the Linux security module, so you actually have the mainline kernel has all the hooks in it, and a global module for SELinux wave enforcing those hooks. Other vendors and other applications have been written to use LSM put pretty much the most mainstream use of LSM is through SELinux module.

7. SELinux Key Components: Policy

The second component of SELinux is policy. SELinux was designed from the ground up to be a strict policy system. There are no deny rules; there are only allow rules. Everything is denied by default. It's designed to have minimum privilege for each daemon or domain. It also is designed in to have separate domains for each application that a user runs, or a lot of applications that a user runs, so for instance, Firefox, Evolution, ssh, different applications run by a user are locked down in a strict policies system.

When we came out with Fedora Core 2, which is first operating system to support SELinux, we had strict policy on by default, and we'll talk about the history of Fedora and how SELinux worked, but we found very quickly that it was difficult to enforce in a general purpose operating system. So figuring out the way each user of Fedora wanted to run their operating system was impossible, so we couldn't really define in the user space all the rules. I get so many bug reports because people were setting up their systems in such different ways that the number one question was, "How do I turn SELinux off?" So we basically had to go back to the drawing board, and we came up with a new type of policy. We decided to work on a new type of policy, and that policy is targeted policy.

Targeted policy is a system where everything is allowed by default and then we lock down certain domains. What we wanted to do is protect the doors and windows of the system. By that I mean to lock down anything that's listening on the network port for incoming communication.

By default we came up with a new domains. Again, we still had the SELinux architecture that basically said everything is denied by default, and we came up with a new domain, a new way of running processes and that was called unconfined process. Basically an unconfined process runs under the same ways it would it would if SELinux wasn't running on the machine at all.

We also came up with targeted domains, a targeted process; for instance, the Apache domain or a name D, DHP, post-gres, so when a targeted domain gets started up through say a net, that process transitions into a lockdown context, so we're basically still protecting user space from the targeted domains.

8. SELinux Key Components: Applications

The last part of SELinux key components is applications. One of the goals of SELinux was that applications and users were not going to have to be SELinux-aware. In a Fedora distribution, there's about 1500 packages, and only about 30 to 40 of those applications or RPM packages are actually SELinux-aware. The ones that have to be SELinux aware are basically to show the user some of the components of SELinux that are running on the system. So for instance, to see what the file contacts or the security contexts of a file arg the

LS command had to be made SELinux aware. Also the login programs basically are set, and users' context initial security context on the system.

Lastly, certain applications are being made SELinux-aware, which the goal being those applications have access control rules in them and they're guarding against information flow, so things like DBUS, X Windows, different applications are being made SELinux-aware so they can take advantage of the SELinux architecture.

9. How SELinux enforces security policy

So how does SELinux enforce security policy? The main idea here is that every single file on the system, every single process on the system, each device on the system has a security context associated with them. So if we go through a user logging onto the system, when a user logs onto the system, the first thing a login program does is associate a security context with that user. Now that user goes out and he runs an application, say the Mozilla application in a strict policy environment. When the user goes to run that application he basically requests to the kernel that he wants to execute the Mozilla application. The kernel looks at the context of the user and looks at the context of the Mozilla application and checks in its rule database to see if the user is able to execute the Mozilla application. If it allows that user to execute the Mozilla application, the Mozilla application will now start up in a new context, usually something like user Mozilla t. All these queries are going through the kernel and going through the loadable security module SELinux security module.

To give you an example of how SELinux works, we can talk about Apache. Apache is probably the most complicated application that's in targeted policy at this time. There was no SELinux awareness in the Apache application, so it's totally unmodified. We decided that people want to run Apache in different ways. Some people might want to run Apache in a really tight, locked-down security system. Say all they want to do is allow Apache to display web pages in var/www/html, so we would call that high security. So we want to allow some way for an administrator to turn up the security on a system to say Apache an only read that page's. You might want to have a medium-level security Apache server which can run cgi-scripts on it, or you might have a low-level Apache webserver that can read, say, Apache pages from users' home directories.

So what we want to do is basically allow sys admin to a configurist system. We don't want to have to have them writing policy, but we want to be able to turn Booleans or make decisions on the way that his Apache system is being configured. Now if a cracker or hacker breaks into the system and breaks into Apache, here we have only access to what Apache was defined to be allowed to do, so even if he was able to somehow get a root account on the system, and you were running under high level security, he would only be able to read files that were in the var/www/html directory, he wouldn't be able to touch the user's home directory, he wouldn't be able to touch ftp password, he wouldn't be able to touch any of the files: just what Apache was designed to do.

10. Configuring policy

The way to do this configuring is we added the concept of Booleans, which are basically, if-then-else rules that are in policy. It allows you to turn on and turn off different sections of policy through the use of Booleans. Secondly you also are allowed to change the file context on system. We also have an application that allows you to adjust these Booleans through human readables, so this is a system configs security level which allows you to basically configure your Booleans through a graphical interface.

11. Writing policy

Advanced users of SELinux have the ability to write policy, to be able to configure the system. This would be, say you're a third party vendor and you wanted SELinux to protect against your targeted domain, you might want to write some policy. Also certain systems, sys admins might find a problem with the policy and that it is not able to get certain things done in their certain, so they might want to write sort of a minor modification to policy. So Red Hat and some of our partners have courses available for writing policy and there are several companies that are coming up, contracting companies, that would allow them to come in and write policy for you, but most of the tools are available for you to be able to write your own policy.

12. SELinux in Fedora Core

Fedora Core 2 was the first mainline operating system that had a mandatory access control in it. The problem is that we ship this with this strict policy so we ended up having to ship Fedora Core 2 with SELinux turned off by default. Users had the ability to turn it on and it came with strict policy, and in Fedora Core 3 we came up with the concept of targeted policy that I talked about earlier, and we were able to turn it on by default, which was a major success. That was the first mainline operating system with mandatory access control turned on by default.

Finally this past spring Fedora Core 4 came out and Fedora Core 4 came out with targeted policy and we had jumped from the original 10 targets that we supported in Fedora Core 3, now to almost 70 targets that are currently protected by SELinux, and strict policy is still available to be used.

13. SELinux in Red Hat Enterprise Linux 4

SELinux on Red Hat Enterprise Linux 4: this was based off of Fedora Core 3 platform, and it's on by default, the default policy is targeted. You see a list up here on the slide that talks about the different domains that we locked down on the system, and there's about 10 of them that are currently locked down on Red Hat Enterprise Linux 4. You can get strict policy supported through Red Hat but it requires a professional services agreement. We also provide training services through our Red Hat Learning Services, and we provide policy writing services.

14. SELinux in Government

Next slide, we're going to talk about SELinux and government. Who is developing SELinux? Again, SELinux is sort of an open source project. It isn't just a bunch of people at Red Hat that are developing SELinux. The major developers of SELinux are the National Security Agency is still working on it. They're the upstream maintainers of the product. Obviously Red Hat is a major developer. We work with DoD. We've gotten together with different people in DoD to figure out what their needs are, and they've contributed back information to it.

We also work with our major partners, IBM and HP. We also work with Traces corporation who does most of the policy work. They're really the lead developers of the new design and policy, and they're working sort of really advanced development team on how we're going to use SELinux in the future. And then TCS corporation is the first one to ship, they've actually worked with Fedora and they've turned on MLS capabilities in the Fedora operating system. We're using a lot of their stuff as we move towards higher levels of certification in the government.

What we're working on right now is new policies, so obviously we're working, you're hearing different talks about MLS: multi-level security. We're also developing a new concept, what we call multi-category security, and basically what MCS does is uses pretty much the same architecture that MLS does, excect that we don't support the Bell-La Padula model with it. The goal really with it is as we studied MLS we found that MLS has been very unsuccessful in non-government markets. We came out with the idea of MCS, which basically allows you to have a single sensitivity level but have multiple categories. Therefore, users have discretion over what categories they're able to assign to files and processes, and then use SELinux to lock down those systems.

15. SELinux in Fedora Core 5

SELinux futures, as I said before, SELinux and Fedora Core 5 is going to have mostly MLS and MCS capabilities turned on. There are going to be additional targets for targeted policy. Our goal is to have better controls over user space, so some of the strict policy capabilities have been improved. We're also looking to lock down things like the executable stacks. So the ways hackers break into system are people using vulnerable applications that basically allow for libraries to have executable stacks, or someone to use a buffer overflow attack. We can actually protect against certain types of attacks with SELinux controls.

We're also looking at better controls over the network. Basically, how do you extend SELinux controls to network-based applications. We're also working with Traces corporation, is the idea of loadable policy modules. Basically in current SELinux the entire policy module is one monolithic package that gets loaded into the kernel. With some of the new technology that we're working on now will allow vendors to provide policy modules that can be loaded and unloaded when applications are installed. We're also working with NSA and Traces on what we're calling reference policy, basically going back to the policy that's sort of evolved over the last couple years and breaking it down and studying where mistakes have been made and coming out and calling reference policy, and that's all plugged into this global policy module capabilities.

16. SELinux futures: Red Hat Enterprise Linux 5

So SELinux futures and Red Hat Enterprise Linux 5 timeframe. Again, we're trying to get to EAL 4 with LSPP. We'd be incorporating the Fedora Core 5 features into SELinux, so if you want to start playing with some of the MLS features and some of the LSPP features of SELinux now, you can start testing with the Fedora Core Rawhide version and Fedora Core 5, which should be coming out by the end of the year.

Eventually we'd also like to get to using Trusted X Windows system. You're basically taking some of the SELinux capabilities and SELinux policy decision-making and moving it to X Windows, and eventually get to things like compartmentalized workstation. You get to the highest levels of security that are available in a mainstream operating system.

17. Resources

There are several websites that are dedicated to SELinux; you can go to the NSA website to get it. There are mailing lists that discuss SELinux, there are chatrooms that discuss SELinux, and we're all available to help you work with SELinux. Thank you very much.

[END RECORDING]