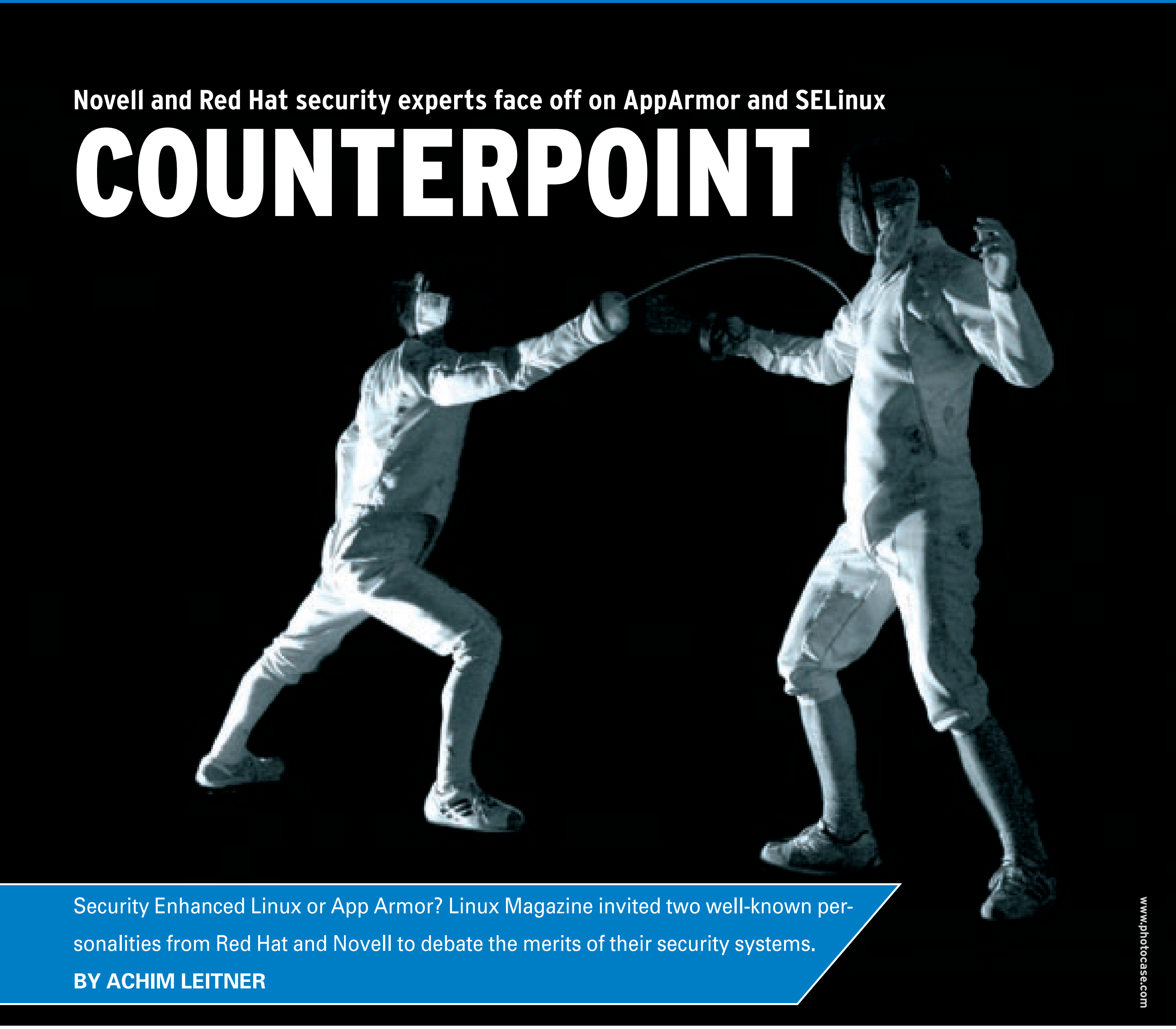
Novell and Red Hat security experts face off on AppArmor and SELinux

COUNTERPOINT



www.photocase.com

Security Enhanced Linux or App Armor? Linux Magazine invited two well-known personalities from Red Hat and Novell to debate the merits of their security systems.

BY ACHIM LEITNER

Novell and Red Hat are currently doing battle to establish their respective products as competitive protection systems for Linux. Whereas Red Hat adopted SELinux years ago, Novell introduced their AppArmor protection system after acquiring Immunix. Both systems are licensed under the GPL, both aim to make Linux more secure, and both give administrators more control over applications privileges.

We asked spokesmen from Novell and Red Hat to explain why their security system is the best. Crispin Cowan, who came to Novell from Immunix, will be talking first about the advantages of AppArmor. Then Daniel Riek will explain why Red Hat will be sticking with SELinux.

Crispin Cowan, Novell

AppArmor[1] and SELinux have similar goals of improving Linux security, but the goals differ in detail. AppArmor secures individual applications against latent defects, and protects an entire system against a particular threat such as network attack, by protecting all applications that face the network. SELinux instead sought to control the whole system, including assuring properties like information flow, and SELinux paid the price in the complexity of the resulting software. The Strict Policy that SELinux first provided was found to be too strict to be usable, and so SELinux actively moved towards the AppArmor model with the Targeted Policy, which simulates AppArmor's

Figure 1: Crispin Cowan: "Simplicity is the soul of security…SELinux seems to have been designed to meet the NSA's desire for arbitrarily complex policy at the expense of usability…AppArmor was designed for usability – to meet the needs of most Linux users."

Crispin Cowan

Crispin Cowan was the CTO and founder of Immunix, Inc., which was recently acquired by Novell. Dr. Cowan now works as an architect for Novell with respect to security for the Linux platform and applications that Novell offers for Linux, and with particular attention to the AppArmor product that came with the Immunix acquisition. Dr. Cowan developed several host security technologies, including the StackGuard compiler defense against buffer overflows and the LSM (Linux Security Modules) interface in Linux 2.6.

Dr. Cowan also co-invented the “time-to-patch” method of assessing when it is safe to apply a security patch. Prior to founding Immunix, he was a professor with the Oregon Graduate Institute Department of Computer Science and Engineering. He holds a Ph.D. from the University of Western Ontario and a Masters of Mathematics from the University of Waterloo.

per-application access control model. AppArmor lets administrators confine applications in familiar terms: you specify the application to be confined and the files to be accessed with absolute path names, followed by familiar *r*ead and *w*rite access modes. Groups of files are granted using traditional shell wildcards, so */home/*/public_html/**.html r* grants read access to all .html files in everyone’s *public_html* tree.

SELinux instead applies labels to files and processes and defines security policy in terms of which labels can access which other labels. Labeled access controls are an established technique from the 1970s, however labels in the general case significantly hinder usability:

- You must label the file system and create security policy as separate steps, creating a circular dependency for the user between specifying labels and specifying policy.
- Some applications such as *tar* do not preserve labels, so data archived and restored with tar will lose its labels.
- NFS mounted filesystems cannot support labels, so the whole file system gets a single label. Thus all network file systems get an all-or-nothing policy decision: each application can either access the entire file system or none of it.

Simplicity is the soul of security: the more complex a system is, the more likely it is to be configured badly. Worse, if a security policy cannot be understood, then it is no policy at all; it is a black box that you hope provides some protection, but you really don’t know.

Much Simpler

AppArmor is considerably simpler than SELinux. This can be seen in this video from Fosdem[2], where an Apache policy is built in 5 minutes. The AppArmor Apache profile is 133 lines, while the corresponding SELinux Apache policy is 826 lines. Magnus Runesson reports he was able to port AppArmor to Ubuntu in less time than it took him to comprehend and modify an SELinux policy.

Despite AppArmor’s relative simplicity, it can also provide security protection that SELinux cannot. AppArmor provides for sub-process confinement of portions of a process, something which SELinux has recently added. However, AppArmor also comes with an Apache module to use this feature, so that users can create AppArmor profiles for things as small as a perl script executed by mod_perl, or even an individual PHP page. I know of no other technology that can confine individual PHP pages.

No Need for Changes

AppArmor is transparent to the application. No application modification is needed to use AppArmor, except for sub-process confinement, which requires some cooperation from the protected process. That cooperation can be achieved using a module if the application supports modules. If AppArmor is abruptly removed, the system continues to function *identically* the way it worked with AppArmor in place, except that it is now more vulnerable to attack.

SELinux can only apply some of its features to un-modified applications; the full feature set is only available if you re-link the application to libselinux, which is feasible for open source applications, but problematic for proprietary enterprise applications.

App Armor Preferred

AppArmor and SELinux both provide high quality security. But SELinux seems to have been designed to meet the NSA’s desire for arbitrarily complex policy at the expense of usability. Poor usability is critical, because it often causes security to not be deployed at all, and SELinux is often disabled when users find the policy too difficult to manage. AppArmor was designed for usability – to meet the needs of most Linux users, both home and enterprise. Try it for yourself: AppArmor is available for Slackware, Ubuntu, Gentoo, Red Hat, Pardus, and integrated into all new editions of Suse Linux for the x86, x86-64, Itanium, Power, and Z-series architectures.

▶ Daniel Riek, Red Hat

SELinux applies strict MAC-based access controls at kernel level (see the article on SELinux). It mititages the impact of successful attacks, guarantees the confidentiality of data, and fulfills complex security demands thanks to context-dependent domain changes.

The first company to announce SELinux support in a commercial product was Novell, although this did not mean that they provided a policy suitable for production use. At this point, the policy was not suitable for a widespread market: too strict, too many restrictions for user application, and certainly somewhat over the top. It was Red Hat that launched the first mature, and production-capable product. Every Red Hat Enterprise Linux 4 installation, and Fedora installation, enables SELinux for central network services by default.

Global Community

SELinux is supported by a large and active community. Besides non-commercial users and providers, the community includes Red Hat, IBM, HP, NSA, DOD, Tresys, and Trusted Computing Systems. These organizations all cooperate on im-

Figure 2: Daniel Riek: “...would you want your credit card data stored on a server whose security policy was created by a novice user using Yast and AppArmor’s complain mode?”

proving policies, developing a powerful auditing infrastructure and policy development tools [4], providing troubleshooting support, and advising users.

In contrast to this, Novell dropped SELinux last year, and started to promote AppArmor, which it had recently acquired from Immunix, as a more simple alternative. Instead of investing in cooperation within the OSS community, and helping to make SELinux easier to use, Novell decided to fork the security architecture in Linux, and hand off responsibility to developers and users, in an approach that reminded Dan Walsh, the head of SELinux development at Red Hat, and others, of the legacy Unix issues [5].

The AppArmor FAQ has this to say about security: "Using the YaST GUI, it's a straightforward task for novice users to develop security profiles, while power users retain the flexibility they need to create finely-tuned profiles." But would you want your credit card data stored on a server whose security policy was created by a novice user using Yast and AppArmor's complain mode? In contrast to this, Red Hat focuses on enterprise use. Software vendors provide verified SELinux policies that customers configure within the framework of validated parameters.

Does More

Of course App Armor is easier to configure because it addresses a far smaller group of security problems. The FAQ even boasts that AppArmor does not guarantee data confidentiality, in contrast to SELinux, claiming that this feature is only useful to secret services. Not a word is lost about credit card data, customer data, medical records, accounts data, Basel II, and Sabranes-Oxley compliance…

And the claim that you need to rebuild applications for SELinux is misleading. With SELinux, the security context after launching a new process depends on who launched the process in which context. There is no need to change the application to do this, and security contexts are clearly defined. Only very few programs require modifications.

AppArmor's sub-process restrictions allow you to run, for example, PHP scripts via mod_php in a context different from the context of Apache itself, although both run within the same process. The FAQ mentions that this is only possible with a special version of Apache with modifications by Novell. In other words, AppArmor needs to rebuild, too, from time to time.

The design of AppArmor has enormous disadvantages: there is nothing to stop malevolent code injected by an attacker into the PHP context from running in the Apache context later. After all, they use the same memory sector. Code hit by an exploit can't give you security! Thus, this scenario will permit escalation of privileges – that's a bug not a feature.

Daniel Riek

Daniel Riek (Figure 2) has been Product Manager Red Hat Enterprise Linux since the beginning of 2006. He joined the company in mid-2003. Before moving to product development, Riek was a Solution Architect and provided pre-sales customer advisory services.

Riek founded ID-PRO, an Internet and GNU/Linux service provider, while he was studying computer science at at the University of Bonn, Germany; and the company grew to become an international player. In 2001, Riek moved from ID-PRO to the French free software service provider, Alcove, where he was responsible for activities in Germany and mainly dealt with key accounts from IT and banking. Riek was a member of the LIVE Linux association's board for many years, and the organization's spokesperson.

Labels vs. Pathnames

The situation with the filesystem argument is similar: SELinux uses security labels, which are stored as extended attributes for filesystem objects. Novell views this as an extension that is only supported by specific filesystems. In fact, extended attributes are a standard feature that just a few filesystems lack, and thus more of an argument against Novell's favorite, Reiser FS.

AppArmor uses pathnames in its profiles, but they can't guarantee security. Whereas SELinux's inode-linked security labels refer to actual filesystem objects, App Armor's pathnames use an abstraction layer that doesn't necessarily reflect the real filesystem. Symbolic links are a simple example of the multi-faceted issues. An object can use multiple pathnames and thus be governed by different policies. The question is if this can still be considered Mandatory Access Control.

A Question of Flexibility

The claim that AppArmor is more flexible than SE-Linux is not based on factual evidence. Admittedly, an AppArmor configuration can be modified more quickly, because it defines a less secure system. But this has nothing to do with flexibility. AppArmor's unidimensional profile design does not give you the same level of security and flexibility that SELinux's dynamic security context changes do. A program can run with different privileges depending on who launches it and from which context. This allows for extremely flexible security profiles.

The SELinux architecture is also suitable for security designs beyond MAC. And the MLS and MCS implementations provide ample evidence that the design works (see the article on SELinux). Both store attributes as extended filesystem attributes and thus support seamless integration with the SELinux policy.

SELinux Preferred

SELinux is the most consistent implementation of Mandatory Access Control in a standard product today. It derives from a fundamental understanding of the way attacks on IT systems work. This said, hackers are always one step ahead in the race to discover the next major exploit. The architecture needs to take this into account and guarantee that even a successful hack does not cause serious problems.

The decision of SELinux or AppArmor is the choice between a comprehensive security architecture on the one hand, and local ad hoc improvements on the other. The typical security demands made by Red Hat Enterprise Linux users can only be truly met by the more complex SELinux. ■

INFO

[1] AppArmor: *http://www.opensuse.org/AppArmor*

[2] AppArmor video: *ftp://ftp.belnet.be/pub/mirror/FOSDEM/FOSDEM2006-apparmor.avi*

[3] SELinux: *http://www.nsa.gov/selinux/*

[4] Developing SELinux policies: *http://selinuxnews.org*

[5] Interview with Dan Walsh: *http://danwalsh.livejournal.com/424.html*