

An Overview of the Upcoming libModSecurity

December 28, 2015 | Posted By Felipe "Zimmerle" Costa | Comments (0) 

libModSecurity is a major rewrite of ModSecurity. It preserves the rich syntax and feature set of ModSecurity while delivering improved performance, stability, and a new experience in easy integration on different.

libModSecurity - Motivations

While ModSecurity version 2.9.0 is available on different platforms (IIS, NGINX, etc...), It really favors an Apache Deployment. ModSecurity standalone is part of ModSecurity project and is basically a wrapper that packs requests from different formats into an Apache format, to later be processed by ModSecurity in the same fashion that it works on an Apache web server. That was certainly the fastest way to have ModSecurity running on different platforms but at the cost of performance and high amount of dependencies. Leading, for instance, to situations where NGINX users have to install Apache dependency in order to have the NGINX ModSecurity module working; see metabug ModSecurity/#661 (<https://github.com/SpiderLabs/ModSecurity/issues/661>) for further information. The growth of the project in terms of integration with scripting languages or adoption in other platforms such as IDSs becomes very difficult because of those limitations.

The addition of new operators or general features is also limited because ModSecurity 2.9 (Sec Language) uses the Apache configuration parser to be loaded. In other words, the language format and conditional syntax could not be extended because it was dependent on a 3rd party software, that may not have our features in mind when they make changes. Even worse, sometimes bugs that heavily affect us are not on the priority list of the Apache (https://bz.apache.org/bugzilla/show_bug.cgi?id=55910) (https://bz.apache.org/bugzilla/show_bug.cgi?id=55910)).

LibModSecurity – Our goals

In order to circumvent those limitations, a year ago we decided to develop a new version of ModSecurity. A version for which the primary goal was not to introduce new features, but to add the possibility of an easy expansion and integration.

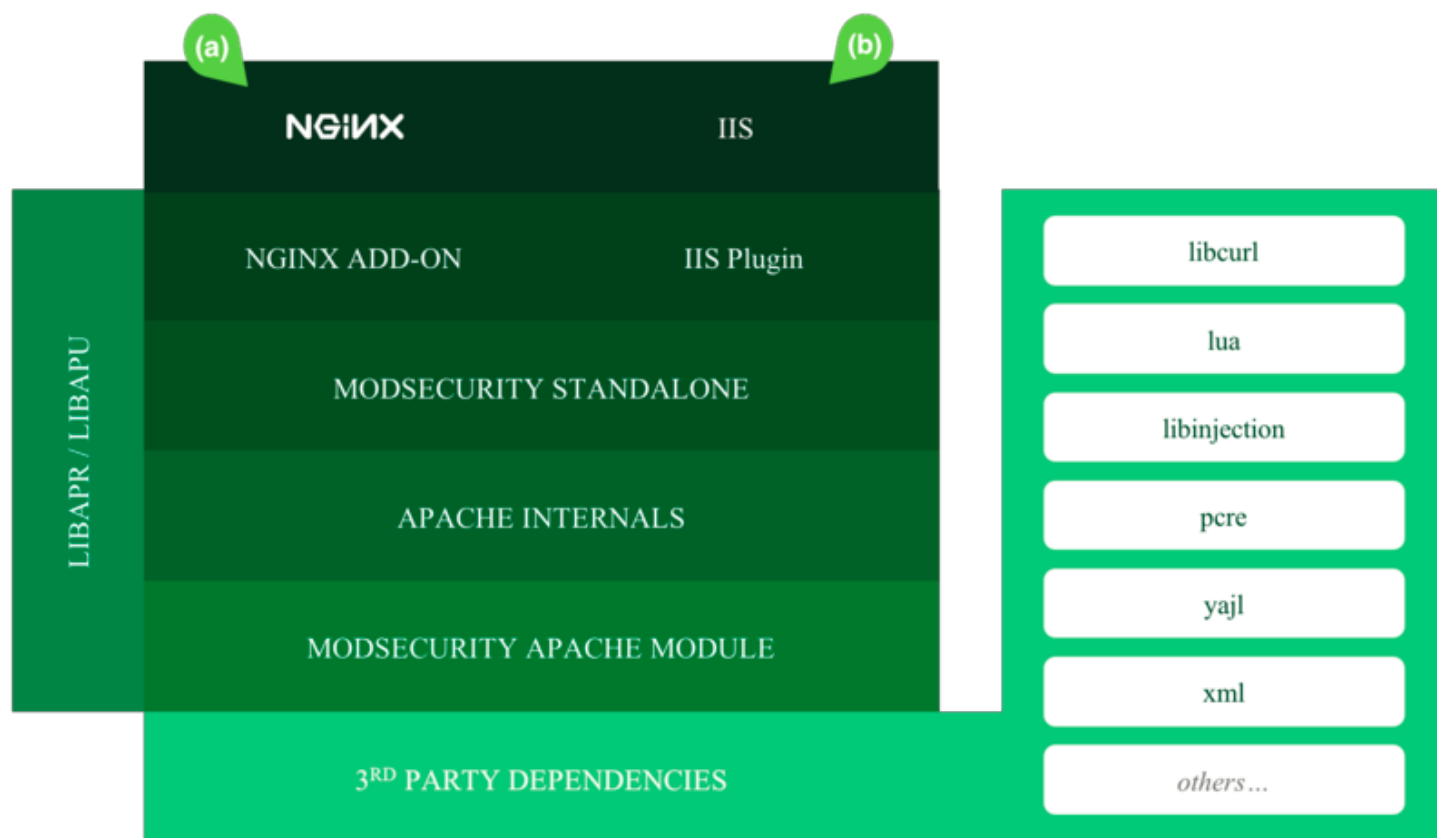
Due to the limitations of "ModSecurity standalone" and ModSecurity v2.9 architecture, we decided to move forward to implement something from scratch. By providing a new architecture while simultaneously supporting all the features of 2.9 we were able to remove many of the limitations that have burdened the project for the past few years.

In this blog post I plan to go over some of the details on why it was important to change the architecture and how some of those changes will lead to positive outcomes. Enjoy...

More about ModSecurity standalone architecture

As explained previously, the ModSecurity 2.x version has a high dependency on 3rd party projects including Apache. As it uses Apache internals directly, it was also using the libapr (Apache Portable Runtime). To highlight this point, we note that almost all memory allocation inside ModSecurity was allocated using libapr memory pools. This is, of course, not a problem, unless you want to remove APR.

To understand a little bit about the level of dependency of ModSecurity version 2.9 on 3rd party modules have a look on the **Figure 1**.



(<http://blog.spiderlabs.com/.a/6a0133f264aa62970b01b7c7fe12d5970b-pi>)

Figure 1. ModSecurity dependencies. (a) nginx extension. (b) IIS extension. Notice that both utilize the ModSecurity Standalone module.

At the right of **Figure 1** we have several dependencies that are bound to the operators or request body parsers (e.g. libxml), some of those are mandatory on ModSecurity version 2.9.x and the idea is that those will become optional in ModSecurity version 3. The availability of a given feature will depend on the existence of the dependency. But, that is the subject for another blog post :)

On the left side of the **Figure 1** are listed the dependencies that are mandatory for the ModSecurity v2.9.x core execution, without these core features of ModSecurity v2.9 cannot function. Removing the APR without removing other Apache dependencies was not be possible, as there are internal calls to the Apache API which demand data in the APR structure (APR memory pool). Additionally, removing just the Apache dependency also did not make sense, as it would need to be replaced by something Apache-like in any event, inheriting all those limitations. So the natural step was to move to a new architecture free of both Apache and APR.

libModSecurity

The first thing that comes to mind when discussing "refactoring" of the ModSecurity core, is the possibility to have a segmentation of what is the "core" and what code is required to interact with a given web server, or as we call it a "connector". Looking at the issues on GitHub it is difficult to tell which bug reports belong to each part of the code.

This monolithic design also puts pressure on us, its maintainers, when developing, testing, and packaging releases. This is because whenever we release in the current version, 2.x, everything needs to be released together (All platforms, even if there is no benefit to a given platform). Splitting the core from the "connectors" seems to be the right choice. Splitting the project give us numerous advantages, not only from the project planning perspective, but also the fact that the library can be easily ported, , and manipulated in this more modular architecture.

By splitting the project between "connectors" and "core", the core naturally become a **library**, and the connectors become consumers of the core library. This way, ModSecurity core becomes completely independent of the underlying web server.

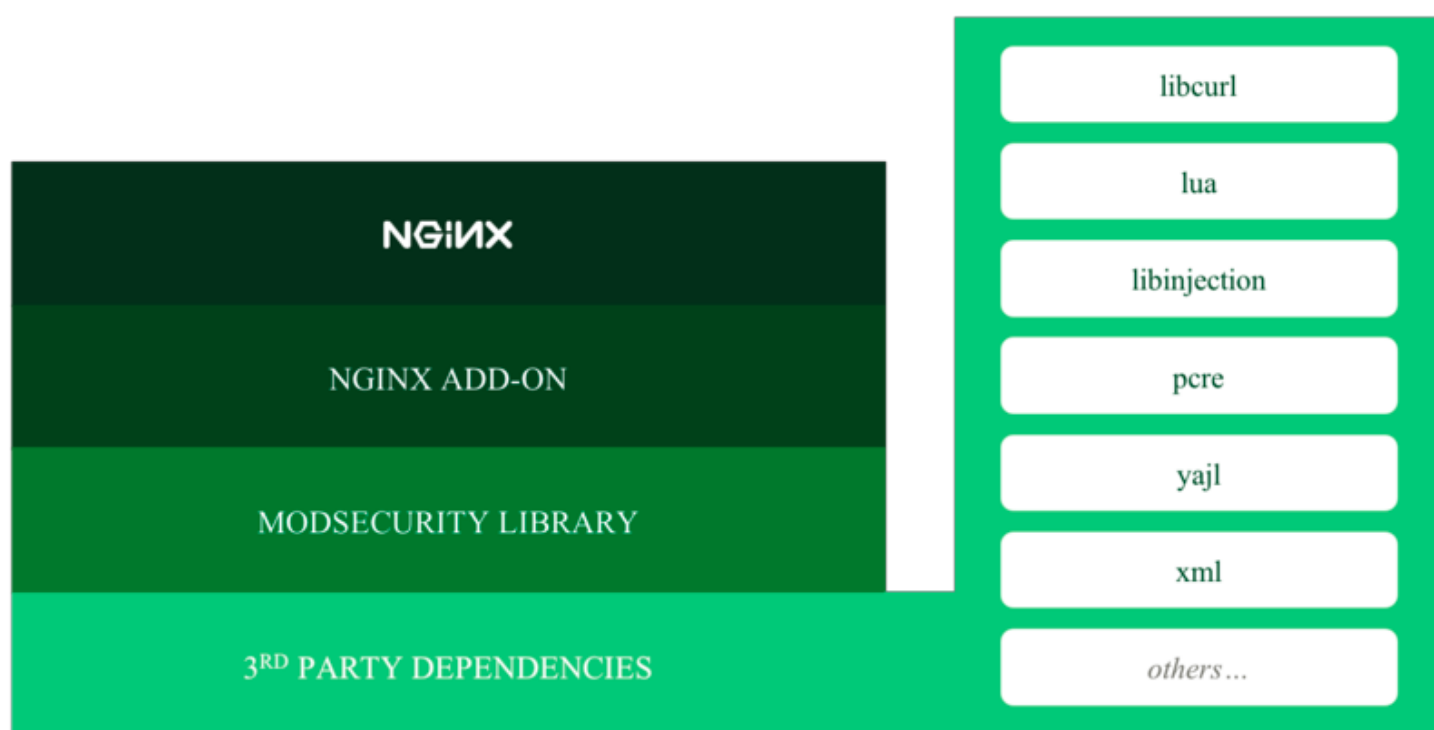
Another goal of the project was to reuse as much code as possible; this was done for two main reasons: (1) the code was tested and proven to work. (2) we didn't need to implement everything from the scratch. So we first implemented a very minimalistic LALR parser (SecLanguage) to read the rules and transform them into C++ objects in memory. That limited language was slowly expanded as we added more operators, transformations and variables.

During the development of the core, other important utilities were also created. Two of those utilities deserve special attention, namely, the **regression** and **unit test** utilities. Both have a very important role in the development of ModSecurity version 3 and probably they will be even more important for the maturity of the project. In the future there will be a specific blog post to cover the regression and unit tests inside ModSecurity version 3.

One of the challenges on ModSecurity version 3 was to rewrite the SecRules, but to return exactly the same results of ModSecurity v2.9.x, thereby avoiding a compatibility break. This meant that we also had to reproduce the corner cases and unexpected behaviors of v2.9.x. This was achieved, so far, by the utilization of the regressions tests, where the requests are mimicked into JSON and the expected results are established from analysis of ModSecurity 2.9.

Going beyond the test, we needed to see ModSecurity version 3 working in practice as part of a web server. We choose the NGINX web server to be the first to fully implemented ModSecurity version 3. For that we started a different GitHub project, called ModSecurity-nginx (<https://github.com/SpiderLabs/ModSecurity-nginx>).

Figure 2 contains the dependencies of "ModSecurity NGINX connector".



(<http://blog.spiderlabs.com/.a/6a0133f264aa62970b01b7c7fe1307970b-pi>)

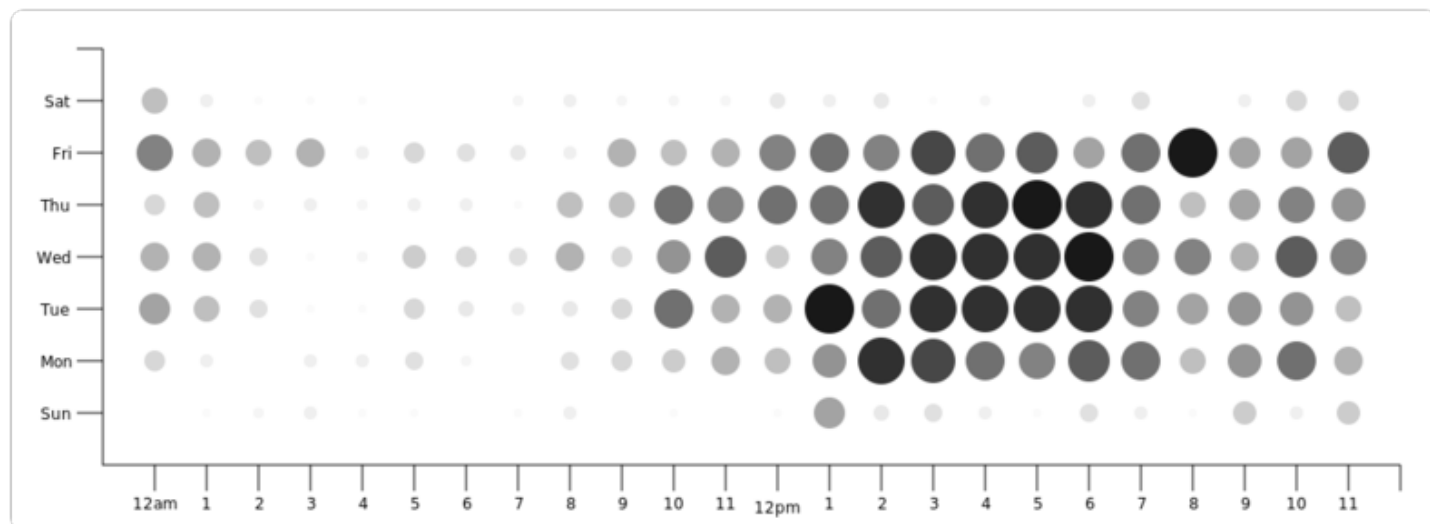
Figure 2. Dependencies of ModSecurity NGINX connector for libmodsecurity.

The ModSecurity NGINX connector, together with libModSecurity was presented at NGINX.conf 2015 (<https://www.nginx.com/nginxconf/schedule/#day2s30>). At the conference we discussed various aspects of the ModSecurity connector and we obtained valuable feedback from the community to shape development of the connector.

Current status?

Currently we have the major features of libModSecurity implemented and ready to use. Although we don't have support for collections yet, it is possible to load the OWASP core rule set version 3.0.0-dev.

So far, we are 374 commits ahead of master. Supporting almost all ModSecurity 2.x features. **Figure 3.** Contains a punch card with the commits to the project so far.



(<http://blog.spiderlabs.com/.a/6a0133f264aa62970b01bb08a2a0b3970d-pi>)

Figure 3. Commits punch card.

Testability

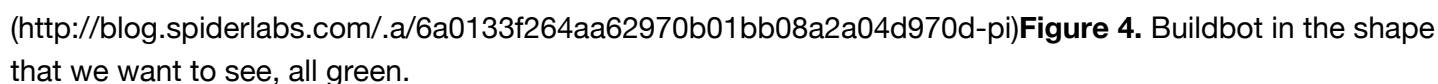
While starting with versions 3, one of the main concerns was the project quality and that is why most of the features contain regression tests and/or unit tests. These tests can be executed on the developer's machine but they are also executed on our BuildBots.

ModSecurity version 2.9.x already has its own regression and unit test utilities, however, they are very slow. A test with these utilities will take around 1 hour, has a dependency on Perl scripts, is Unix only, and is dependent on the web server.

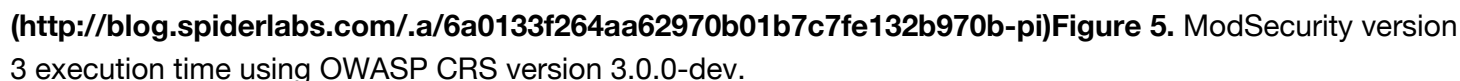
With version 3 we fork the tests used in 2.9 into a separate GitHub project, and we added it inside ModSecurity by the utilizing a Git subtree. Having the test cases migrated into a separate project gives us the flexibility to run different tests with different versions of ModSecurity.

We also created another subset of tests that is specific to the connector. This was done so we can segregate potential problems between the core and any connectors. In the specific case of NGINX, the regression tests were built as an extension of the NGINX server regression tests.

All these tests are executed for each commit that is performed on GitHub. This gives us the capability to promptly identify and fix any problem during the development of ModSecurity v3. It also gives us the ability to mimic problems without needing an entire web server configuration.



Performance is something that we are constantly measuring during the development of version 3. Instead of using the DebugLogs, with performance time stamps, we created SystemTap scripts that allow for real time instrumentations. SystemTap also allows the creation of flame charts. As demonstrated on **Figure 5**.



The **Figure 5** illustrates the mean times of each rule, grouped into different phases. Notice that the entire execution of the OWASP CRS took 483 microseconds. That specific subject deserves a creation of a specific blog post, together with the possibility of rules optimization.

As mentioned, the libModSecurity still isn't feature complete when compared to ModSecurity version 2.9.x. Your help is more than welcome to support that initiative. Currently, we have missing Connectors, Operators, Transformations, and Variables. How do you want to help?

As mentioned before, currently we only have connector for nginx. In other words, Apache users will not have any benefit from ModSecurity v3, at least, not yet.

The development of the connectors for IIS and Apache will start as soon as we have a version of ModSecurity v3 released, unless someone from the community starts to develop it :)

<https://www.trustwave.com/Resources/SpiderLabs-Blog/An-Overview-of-the-Upcoming-libModSecurity/>

A question that we frequently receive is how to start coding for ModSecurity. Well, this is a good opportunity. The missing features in ModSecurity versions 3 are not very difficult to, plus, many of them already contain a description of what needs to be done. Some even have the file already, and just need to be filled. For a complete list of missing features with descriptions, check here:

Although all the operators needed by the ModSecurity SLR commercial rules are already supported inside ModSecurity version 3, we just encourage the usage of ModSecurity version 3 to advanced users. ModSecurity version 3 was not released yet, thus, not considered stable. Once it is released it will be 100% compatible with our SLR commercial rules.

Testing!

At this stage of the development testing is very important. We count on the community to provide feedback and report any issue on ModSecurity version 3. We are aiming to have a release candidate soon, ideally with only a small number of issues.

Share:	 LinkedIn	 Facebook	 Twitter	 Embed	 Email
--------	-----------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------

Tags: (/Resources/SpiderLabs-Blog/?page=1&year=0&month=0)

Trustwave reserves the right to review all comments in the discussion below. Please note that for security and other reasons, we may not approve comments containing links.



(/)

Resources > SpiderLabs Blog (/Resources/SpiderLabs-Blog) >

Share:	 LinkedIn	 Facebook	 Twitter	 Email
--------	-------------------------------------------------------------------------------------------------	-------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------



0 Comments

Trustwave.com

1 Login

Recommend 1

Share

Sort by Best



Start the discussion...


Be the first to comment.

ALSO ON TRUSTWAVE.COM

WHAT'S THIS?

Custom Native Library Loader for Android


2 comments • a year ago



Antelox — Hi, nice hack! BTW, I'm trying it but I get this: Loading libs Lib loaded, getting dlysm calling get ...

Bring Out Your Dead: An Update on the PCI relevance of SSLv3


5 comments • 9 months ago



Owen Orwell — What's the issue with TLS 1.0, specifically? 1.1 and 1.2 are not supported on Windows XP, ...

CVE-2014-0050: Exploit with Boundaries, Loops without ...


2 comments • a year ago



Alex — Further to my previous comment I am guessing that the call to input.read() returns 0 bytes ...

Attacking Ruby Gem Security with CVE-2015-3900

5 comments • 6 months ago



Marek Cybul — Great research on this abandoned area.

Subscribe

Add Disqus to your site

Add Disqus

Add

Privacy

DISQUS

< Prev (/Resources/SpiderLabs-Blog/Neutrino-Exploit-Kit---One-Flash-File-to-Rule-Them-All/?page=1&year=0&month=0)

Next > (/Resources/SpiderLabs-Blog/3-in-1-Malware-Infection-through-Spammed-JavaScript-Attachments/?page=1&year=0&month=0)

Recent Posts

3-in-1 Malware Infection through Spammed JavaScript Attachments (/Resources/SpiderLabs-Blog/3-in-1-Malware-Infection-through-Spammed-JavaScript-Attachments/)

Dec 22, 2015 | Rodel Mendrez

Protecting Your Sites from Apache.Commons Vulnerabilities (/Resources/SpiderLabs-Blog/Protecting-Your-Sites-from-Apache-Commons-Vulnerabilities/)

Dec 21, 2015 | Assi Barak

Joomla 0-Day Exploited In the Wild (CVE-2015-8562) (/Resources/SpiderLabs-Blog/Joomla-0-Day-Exploited-In-the-Wild-(CVE-2015-8562)/)

Dec 18, 2015 | Assi Barak

TrustKeeper Scan Engine Update for December 16, 2015 (/Resources/SpiderLabs-Blog/TrustKeeper-Scan-Engine-Update-for-December-16,-2015/)

Dec 16, 2015 | Lolita Chandra

Mom Spies a Hack (/Resources/SpiderLabs-Blog/Mom-Spies-a-Hack/)

Dec 15, 2015 | Jonathan Yarema

Stay Connected



(<https://www.linkedin.com/groups/SpiderLabs-90640>)



(<https://twitter.com/spiderlabs>)



(<https://plus.google.com/+trustwave/posts>)



(<https://www.facebook.com/Trustwave>)



(<http://www.youtube.com/user/TheRealTrustwave>)



(</rss/SpiderLabs-Blog>)

Subscribe Now

Sign up to receive the latest security news and trends from Trustwave.

Subscribe

No spam, unsubscribe at any time.

Archive

2015 (147) (/Resources/SpiderLabs-Blog/?page=1&year=2015&month=0)

2014 (218) (/Resources/SpiderLabs-Blog/?page=1&year=2014&month=0)

2013 (238) (/Resources/SpiderLabs-Blog/?page=1&year=2013&month=0)
2012 (256) (/Resources/SpiderLabs-Blog/?page=1&year=2012&month=0)
2011 (109) (/Resources/SpiderLabs-Blog/?page=1&year=2011&month=0)
2010 (31) (/Resources/SpiderLabs-Blog/?page=1&year=2010&month=0)
2009 (5) (/Resources/SpiderLabs-Blog/?page=1&year=2009&month=0)
2008 (46) (/Resources/SpiderLabs-Blog/?page=1&year=2008&month=0)
2007 (38) (/Resources/SpiderLabs-Blog/?page=1&year=2007&month=0)
2006 (23) (/Resources/SpiderLabs-Blog/?page=1&year=2006&month=0)
2005 (19) (/Resources/SpiderLabs-Blog/?page=1&year=2005&month=0)
2004 (14) (/Resources/SpiderLabs-Blog/?page=1&year=2004&month=0)
2003 (17) (/Resources/SpiderLabs-Blog/?page=1&year=2003&month=0)

< Prev (/Resources/SpiderLabs-Blog/Neutrino-Exploit-Kit---One-Flash-File-to-Rule-Them-All/?page=1&year=0&month=0)

Next > (/Resources/SpiderLabs-Blog/3-in-1-Malware-Infection-through-Spammed-JavaScript-Attachments/?page=1&year=0&month=0)

Subscribe Now

Sign up to receive the latest security news and trends from Trustwave.

Email Address

Subscribe

No spam, unsubscribe at any time.



(<http://www.linkedin.com/company/trustwave>)



(<https://twitter.com/trustwave>)



(<https://plus.google.com/+trustwave>)



(<https://www.facebook.com/Trustwave>)



(<http://www.youtube.com/user/TheRealTrustwave>)



Copyright © 2015 Trustwave Holdings, Inc. All rights reserved.

[Terms of Use \(/Company/Legal-Notice/\)](#) | [Privacy Policy \(/Company/Privacy-Statement/\)](#)