NETWORK SECURITY PRACTICES – ATTACK AND DEFENSE

Reconnaissance – Vulnerability Scanning

Reconnaissance

- Systematic and methodical understanding of a system's security posture
  - Domain names, IP addresses, routers, servers,…
- Network Scanning
- **Vulnerability Scanning**
- LAN Reconnaissance
- Wireless Reconnaissance
- Custom Packet Generation

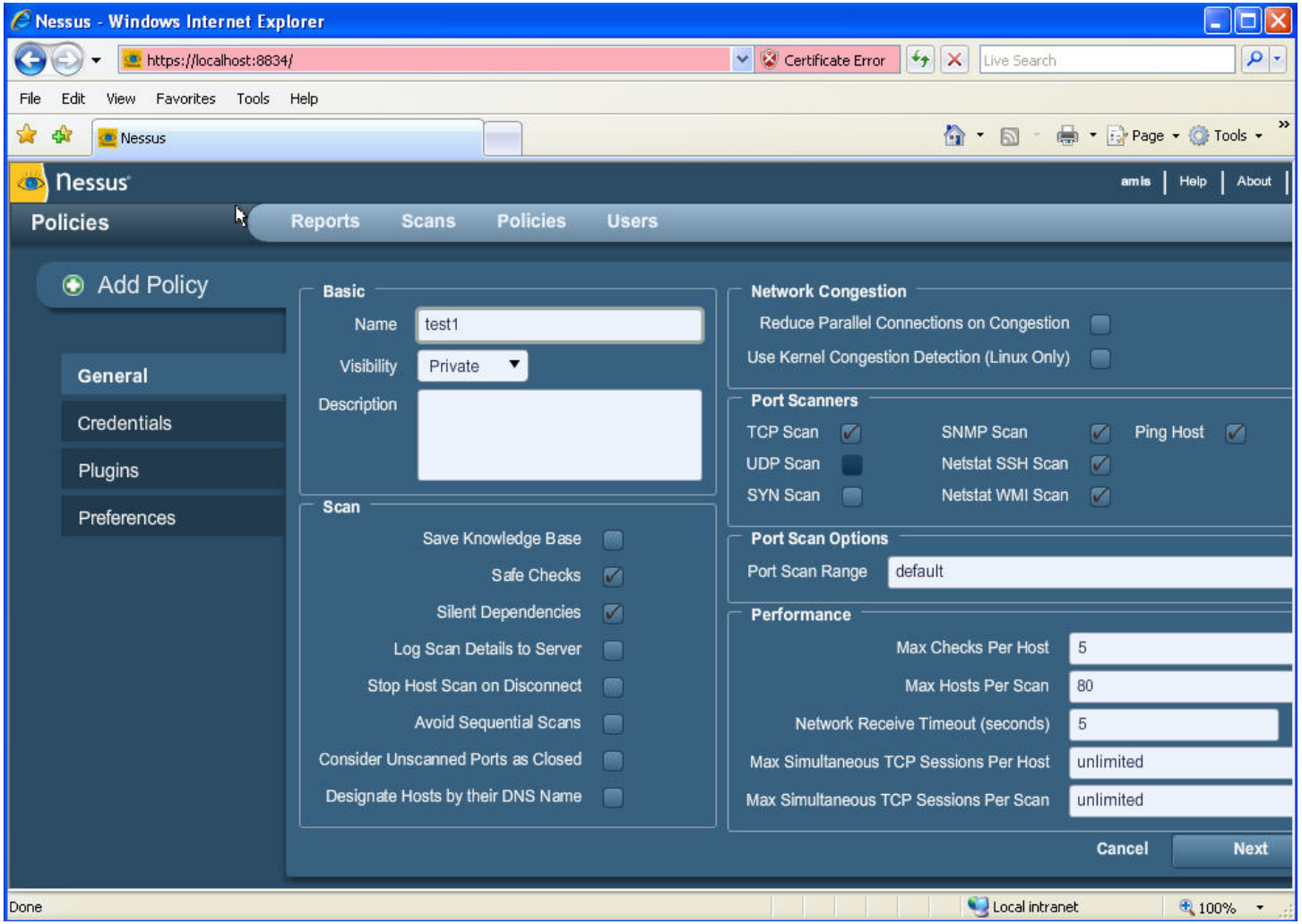
Vulnerability Scanning

- Look for known vulnerabilities in known products
- An *intrusive* test tries to exercise the vulnerability, which can crash or alter the remote target.
- A *non-intrusive* test tries not to cause any harm to the target. (by checking versions / options)
  - Pros and Cons between the two?
  - (intrusive) vulnerability scanning ≠ penetration test
  - Good idea to use vulnerability scanning to test IDS ?

Nessus

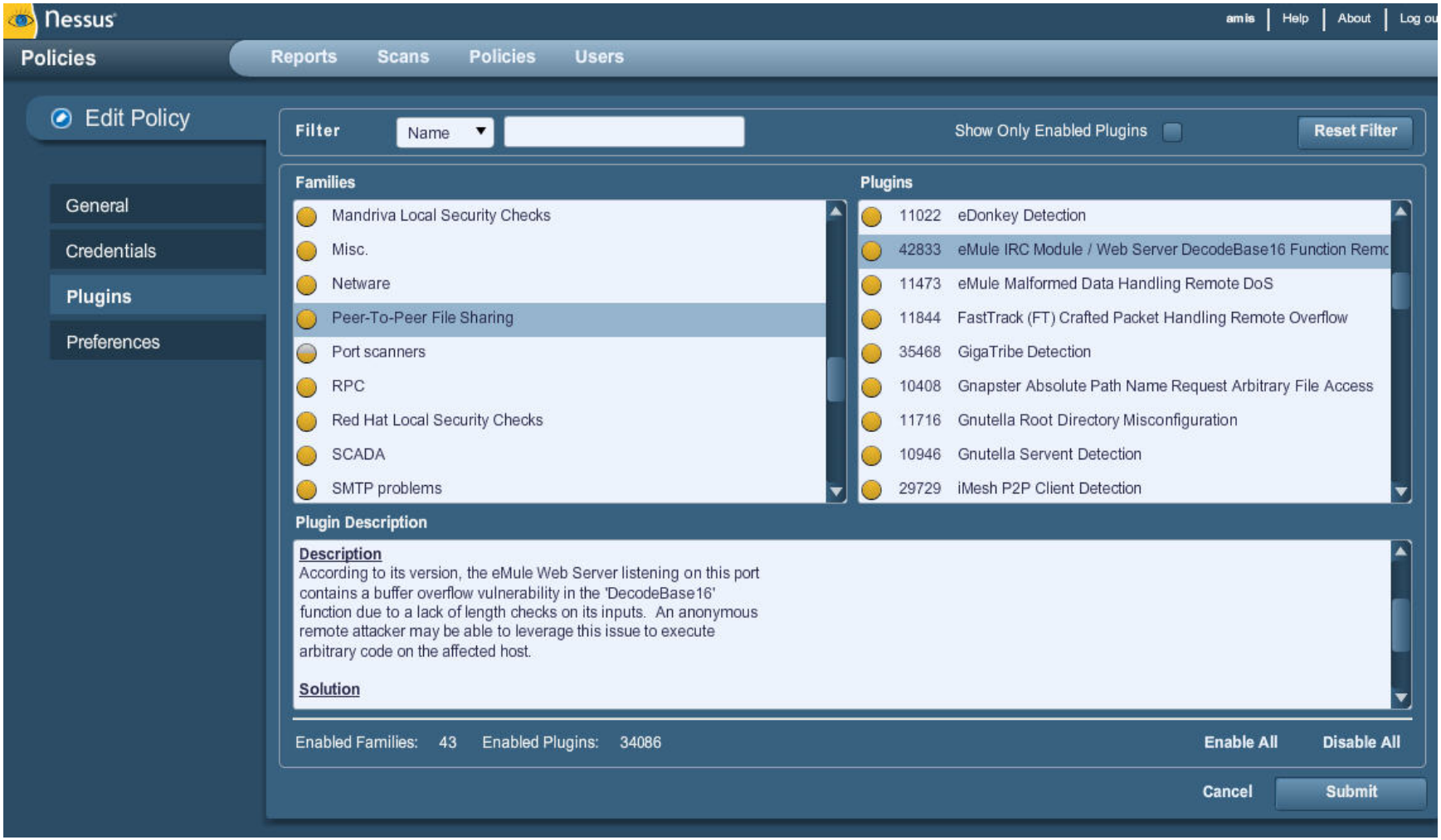
- The best known tool in vulnerability scanning (http://www.nessus.org)
- Local Vulnerabilities
  - Require login (SSH/SMB) to local machines
  - Check for security vulnerabilities, file permissions, configuration files
- Network Scan
  - Intrusive / Non-intrusive
- Client-Server architecture
  - Can control the scan engine (server) remotely
  - Scan engine can inspect multiple machines on the network

Nessus / Scan Policy

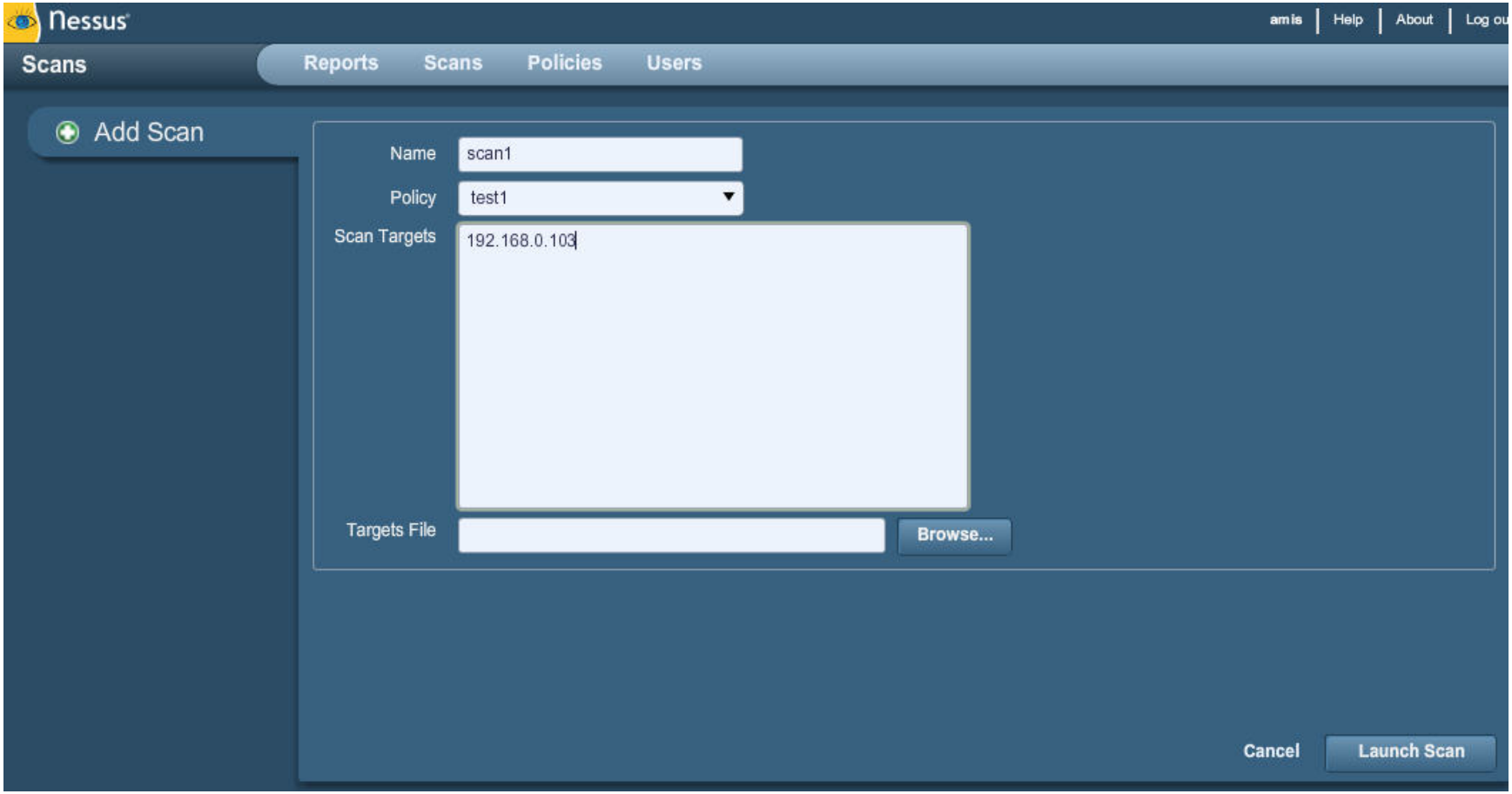


Nessus / Plugins

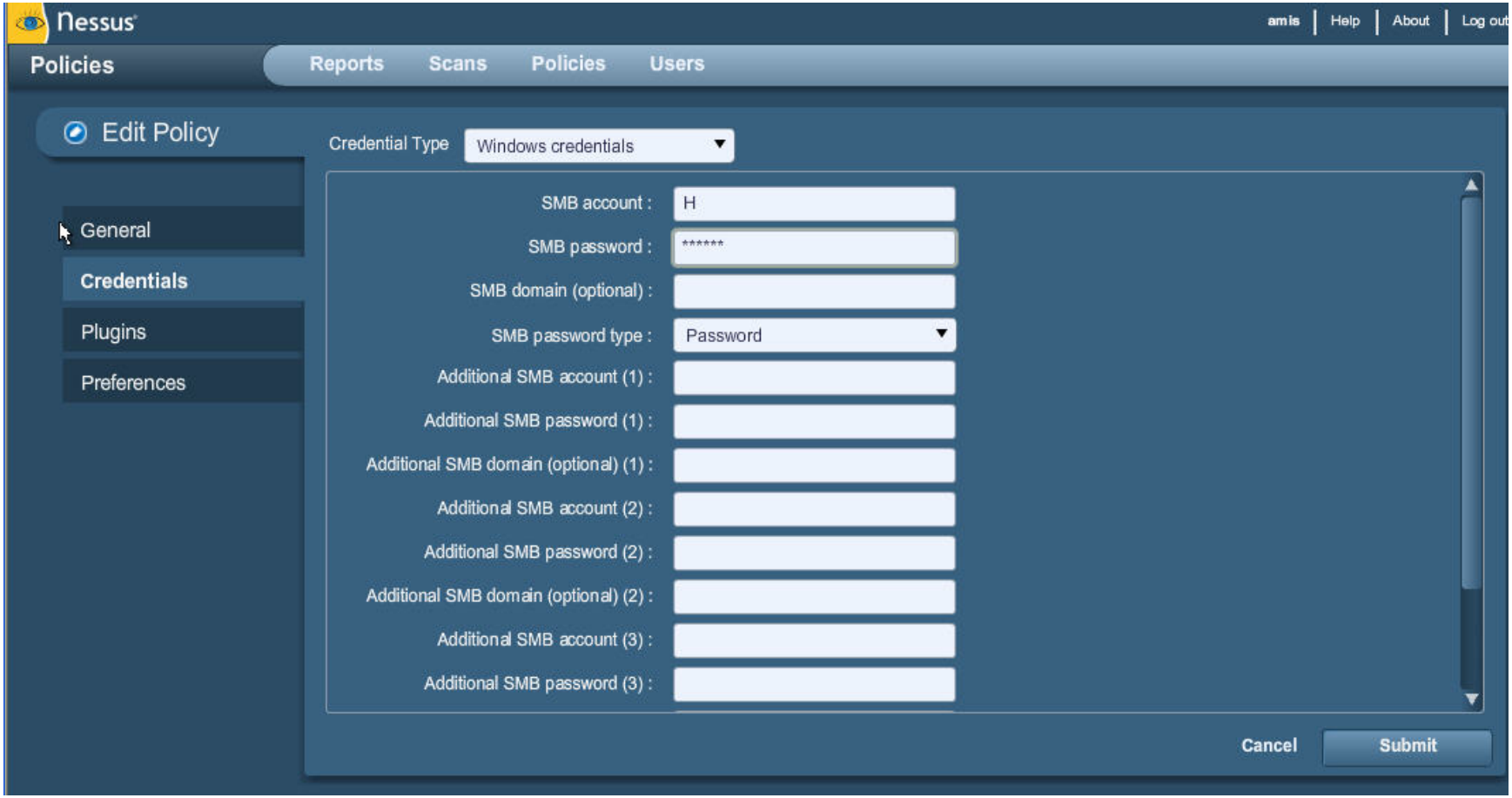
Nessus / Plugins



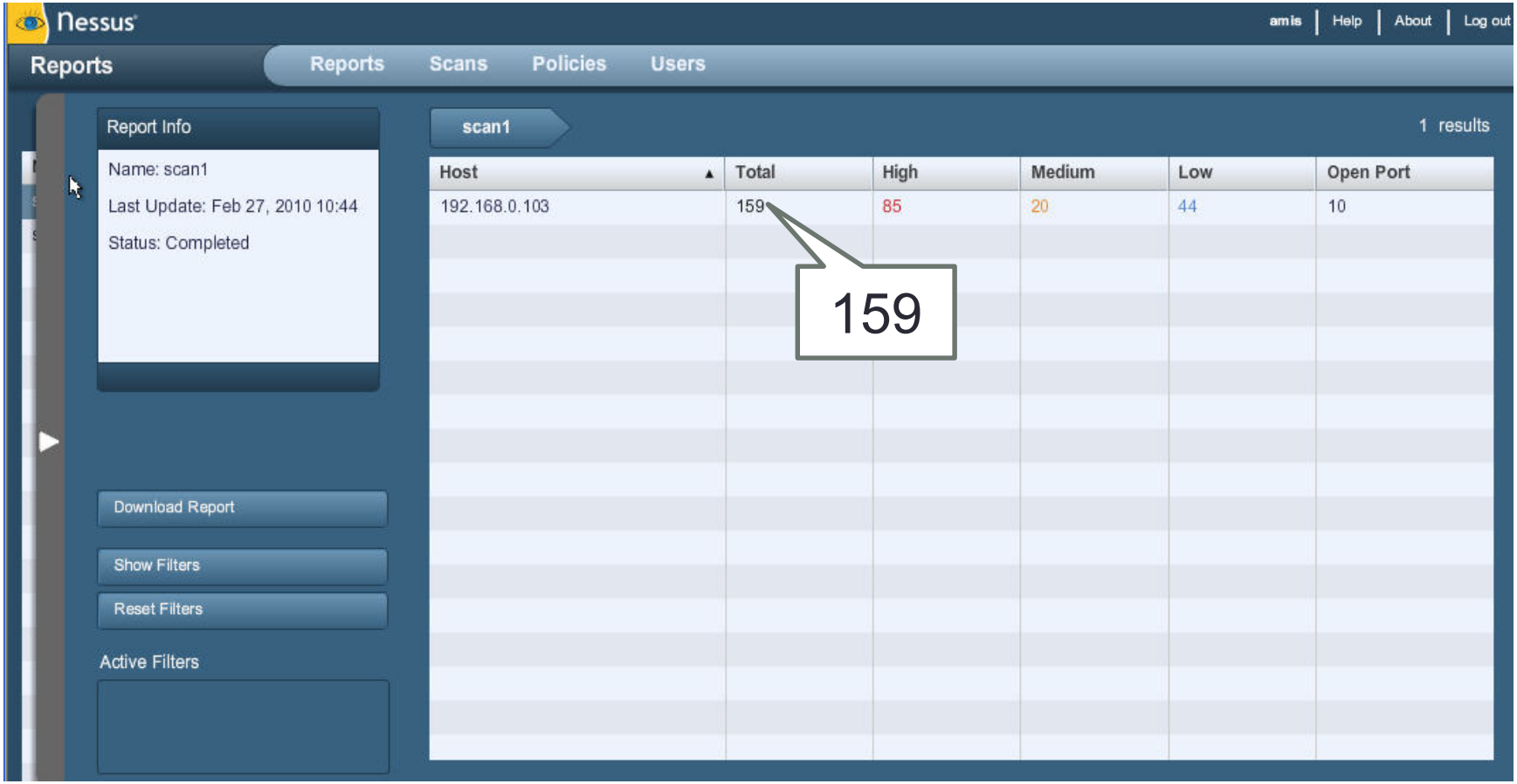
Nessus / Add Scan



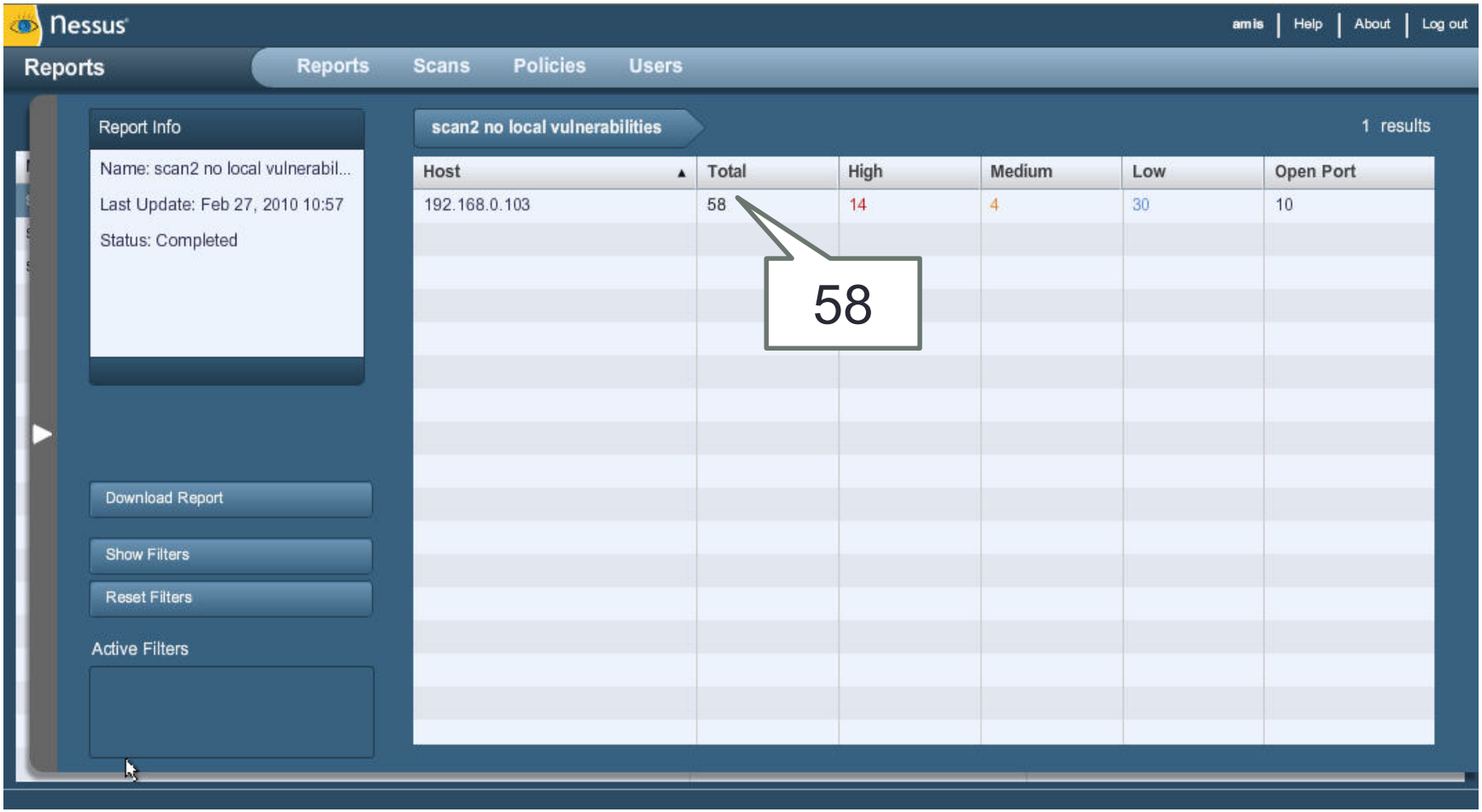
Nessus / Credentials



Nessus / Scan summary



Nessus / Scan Summary (no local vulnerability check)



Nessus / Detailed scan result

MS08-067: Microsoft Windows Server Service Crafted RPC Request Handling Remote Code Execution (958644) (uncredentialed check)

Synopsis:
Arbitrary code can be executed on the remote host due to a flaw in the 'Server' service.

Description:
The remote host is vulnerable to a buffer overrun in the 'Server' service that may allow an attacker to execute arbitrary code on the remote host with the 'System' privileges.

Risk factor:
Critical

CVSS Base Score:10.0
CVSS2#AV:N/AC:L/Au:N/C:C/I:C/A:C

Solution:
Microsoft has released a set of patches for Windows 2000, XP, 2003, Vista and 2008 :
http://www.microsoft.com/technet/security/bulletin/ms08-067.mspx

Plugin ID:
34477

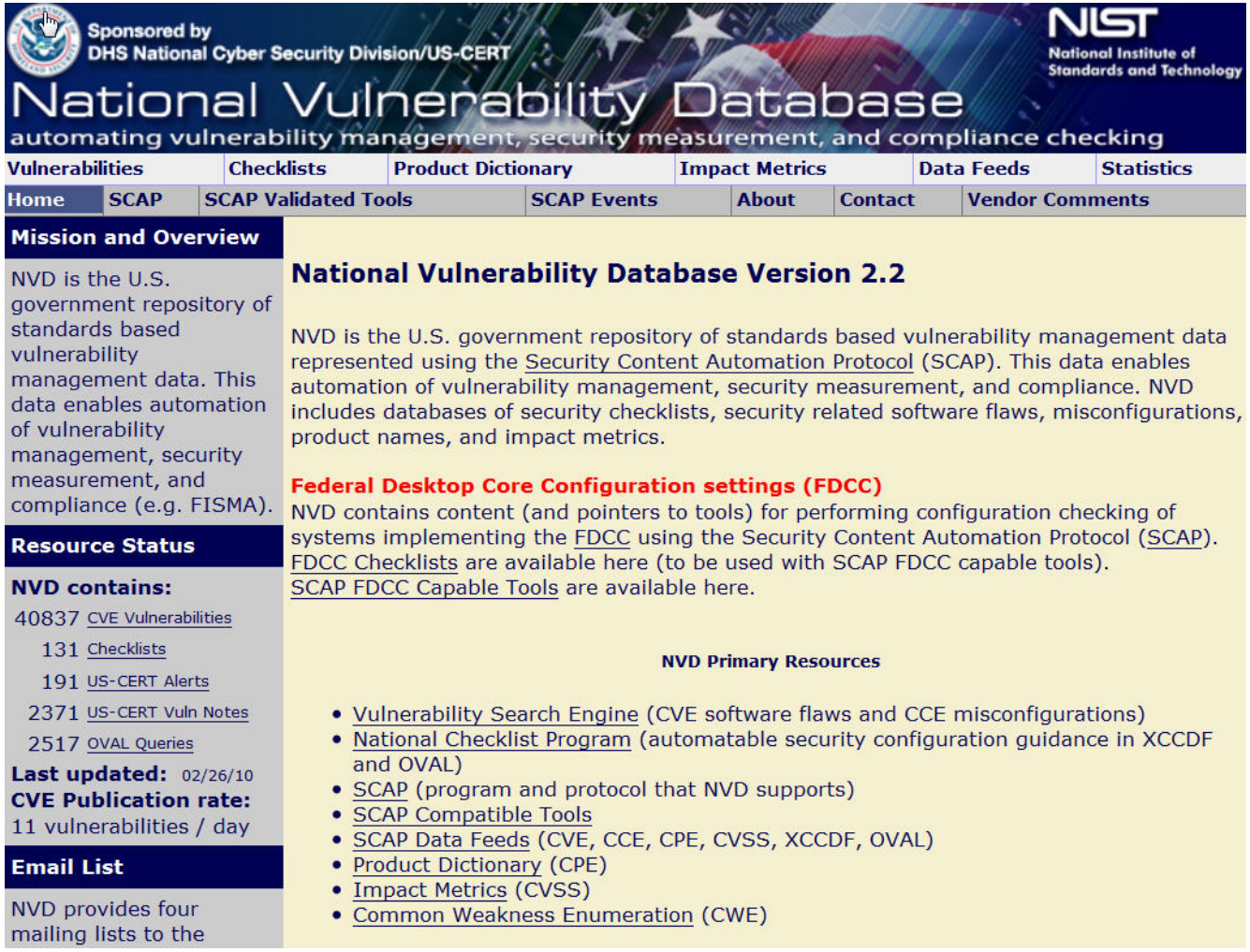
CVE:
CVE-2008-4250

BID:
31874

Other references:
OSVDB:49243

National Vulnerability Database

- http://nvd.nist.gov/



NVD: Security Configuration Checklist

Home | SCAP | SCAP Validated Tools | SCAP Events | About | Contact | Vendor Comments

National Checklist Program Repository

The National Checklist Program (NCP), defined by the NIST SP 800-70 Rev. 1, is the U.S. government repository of publicly available security checklists (or benchmarks) that provide detailed low level guidance on setting the security configuration of operating systems and applications. NCP is migrating its repository of checklists to conform to the Security Content Automation Protocol (SCAP). SCAP enables standards based security tools to automatically perform configuration checking using NCP checklists. For more information relating to the NCP please visit the information page or the glossary of terms.

Search for Checklist using the fields below. The keyword search will search across the name, and summary.

Tier: Any.....
Target Product: Any.....
Product Category: Any.....
Authority: Any.....
Keyword: Search

Checklist Results

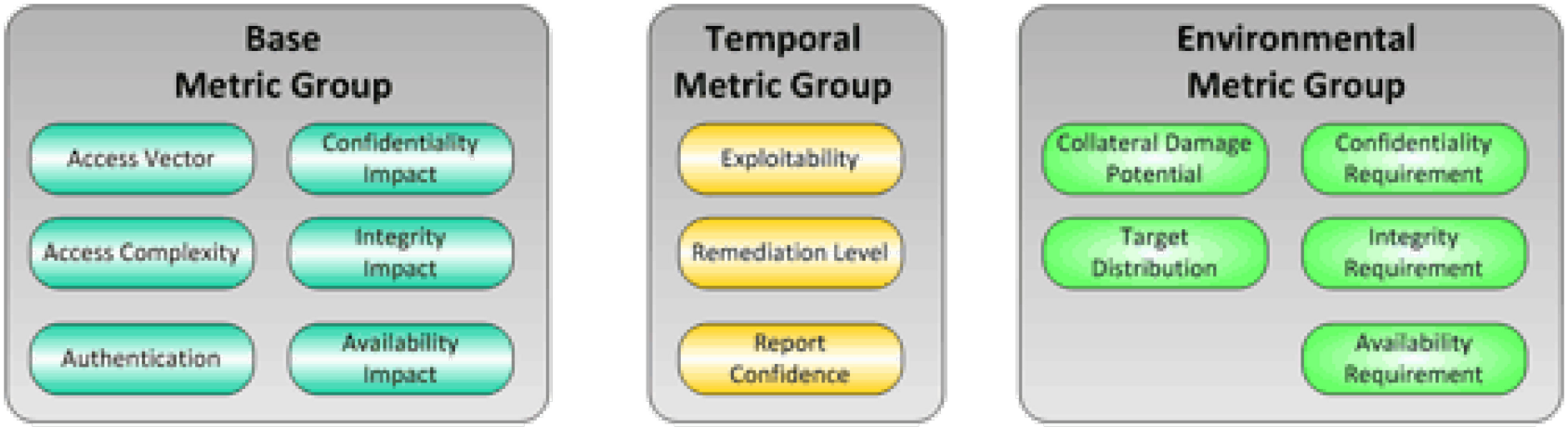
Tier	Target Product	Product Category	Authority	Publication Date	Checklist Name (Version)	Resources
IV	• Microsoft Internet Explorer 7	• Web Browser	• OMB	06/19/2008	FDCC IE7 (1.2)	• SCAP Content - OVAL 5.3 • SCAP Content - OVAL 5.4 • GPOs • Prose
IV	• Microsoft Windows Vista	• Operating System	• OMB	06/19/2008	FDCC Windows Vista (1.2)	• SCAP Content - OVAL 5.3 • SCAP Content - OVAL 5.4 • GPOs • Prose
IV	• Microsoft Windows Vista	• Operating System	• OMB	06/19/2008	FDCC Windows Vista Firewall (1.2)	• SCAP Content - OVAL 5.3 • SCAP Content - OVAL 5.4

NVD: Security Configuration Checklist

```
<definition id="oval:gov.nist.fdcc.ie7:def:1198" version="1" class="compliance">
      <metadata>
            <title>Disable Automatic Install of Internet Explorer Components</title>
            <affected family="windows">
                  <platform>Microsoft Windows XP</platform>
                  <platform>Microsoft Windows Server 2003</platform>
                  <platform>Microsoft Windows Vista</platform>
                  <product>Microsoft Internet Explorer 7</product>
            </affected>
            <reference source="http://cce.mitre.org" ref_id="CCE-3518-8"/>
            <reference source="cce.mitre.org/version/4" ref_id="CCE-684"/>
            <description>This Disable Automatic Install of Internet Explorer components setting prevents
            Internet Explorer from automatically installing components.</description>
      </metadata>
      <criteria>
            <extend_definition comment="Microsoft Internet Explorer 7 is installed"
            definition_ref="oval:gov.nist.fdcc.ie7:def:627"/>
            <criterion comment="Computer Configuration\Network\Internet Explorer\Disable Automatic Install of
            Internet Explorer Compoents" test_ref="oval:gov.nist.fdcc.ie7:tst:3956"/>
            <!--                        <criterion comment="Computer Configuration\Network\Internet
            Explorer\Disable Automatic Install of Internet Explorer Compoents"
            test_ref="oval:gov.nist.fdcc.ie7:tst:3957"/>-->
      </criteria>
</definition>
```

CVSS (Common Vulnerability Scoring System)

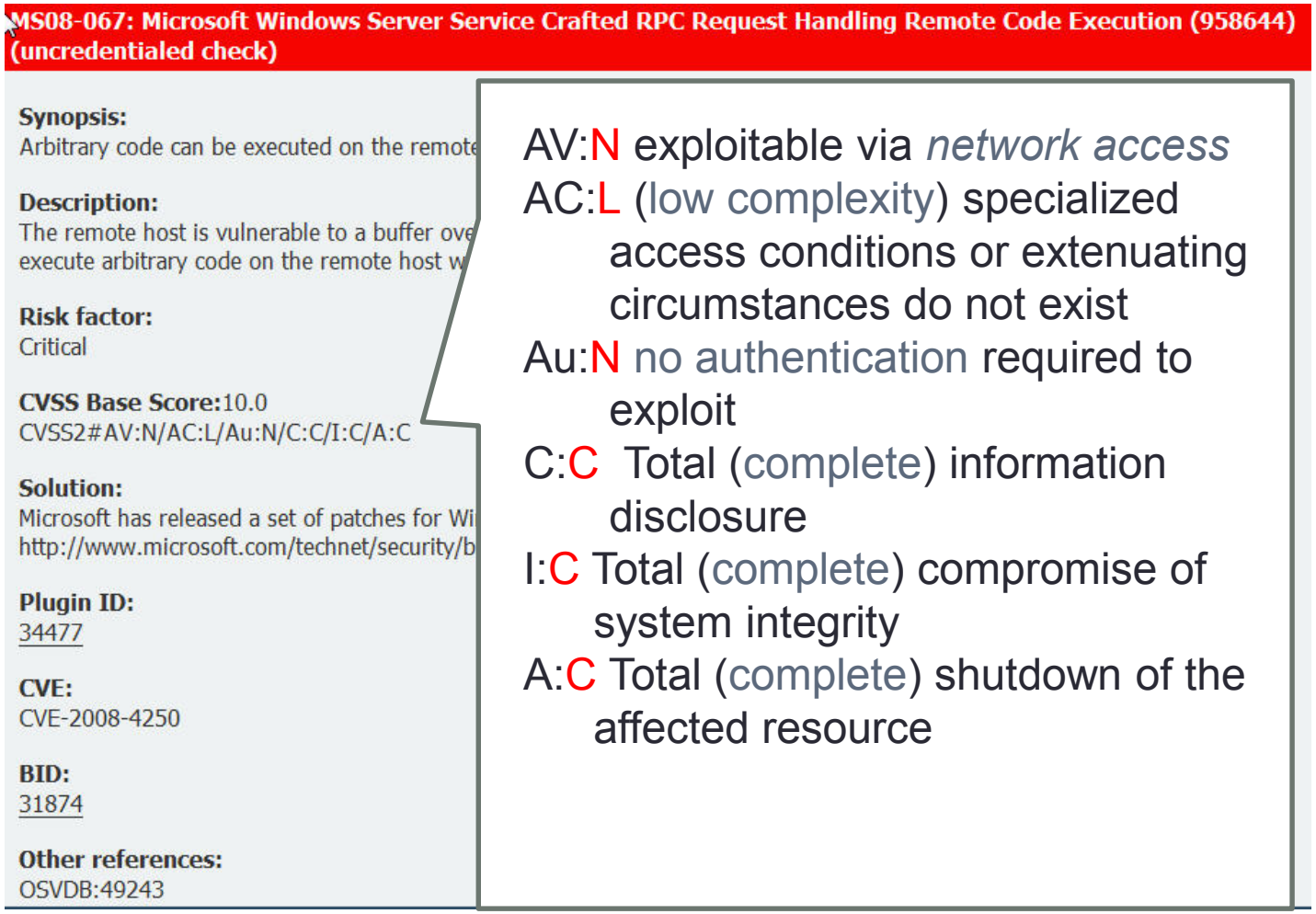
- http://www.first.org/cvss/cvss-guide.html
- Too many vulnerabilities to deal with
  - Prioritize the vulnerabilities and remediate the most risky ones first
- Three metric groups



CVSS / Metric Groups

- **Base**: represents the intrinsic and fundamental characteristics of a vulnerability that are constant over time and user environments.
- **Temporal**: represents the characteristics of a vulnerability that change over time but not among user environments.
- **Environmental**: represents the characteristics of a vulnerability that are relevant and unique to a particular user's environment.

CVSS Score and Vector



CVSS Score and Vector

```
BaseScore = round_to_1_decimal(((0.6*Impact)+(0.4*Exploitability)-1.5)*f(Impact))

Impact = 10.41*(1-(1-ConfImpact)*(1-IntegImpact)*(1-AvailImpact))

Exploitability = 20* AccessVector*AccessComplexity*Authentication

f(impact)= 0 if Impact=0, 1.176 otherwise

AccessVector     = case AccessVector of
                        requires local access: 0.395
                        adjacent network accessible: 0.646
                        network accessible: 1.0

AccessComplexity = case AccessComplexity of
                        high: 0.35
                        medium: 0.61
                        low: 0.71

Authentication   = case Authentication of
                        requires multiple instances of authentication: 0.45
                        requires single instance of authentication: 0.56
                        requires no authentication: 0.704
```

Nessus Plug-in Code Example

- C:\Program Files\Tenable\Nessus\nessus\plugins\zope.nasl

```
if(description)
{
   script_id(10447);  script_version ("$Revision: 1.21 $");  script_cve_id("CVE-2000-0483");
script_bugtraq_id(1354);
script_xref(name:"OSVDB", value:"347");
 script_name(english:"Zope < 2.1.7 DocumentTemplate Unauthorized DTML Entity
Modification");
[.....]
  script_category(ACT_GATHER_INFO);
   exit(0);
}

# The script code starts here
[.....]

port = get_http_port(default:80);
banner = get_http_banner(port:port);

if(banner)
{
    if(egrep(pattern:"^Server: .*Zope 2\.((0\..*)|(1\.[0-6]))", string:banner))
          security_hole(port);
}
```

ACT_GATHER_INFO : the script will be launched among the first. You know it will not harm the remote computer.
ACT_ATTACK : the script will attempt to gain some privileges on the remote host. It may harm the remote system (if it tests a buffer overflow for instance)
ACT_DENIAL : the script will attempt to crash the remote host
ACT_SCANNER : the script is a port scanner

Nessus Plug-in Code Example SMTP OPEN Relay

```
[.............]
send(socket: soc, data: strcat('HELO ', src_name, '\r\n'));
smtp_recv_line(socket: soc);
for (i = 0; soc && (from_l[i] || to_l[i]); i ++)
{
  mf = strcat('MAIL FROM: <', from_l[i], '>\r\n');
  send(socket: soc, data: mf);
  l = smtp_recv_line(socket: soc);
  if (! l || l =~ '^5[0-9][0-9]')
  {
    smtp_close(socket: soc);
    soc = smtp_open(port: port, helo: domain);
  }
  else
  {
    rt = strcat('RCPT TO: <', to_l[i], '>\r\n');
    send(socket: soc, data: rt);
[.....................................................]
    l = smtp_recv_line(socket: soc);
    if (l =~ '^2[0-9][0-9]')
    {
      flag = 1;
[.............]
```

Smtp_relay2.nasl

HELO localhost
MAIL FROM: <nessus@localhost>
RCPT TO: <nessus@domain.com>

Check for 200~299 response code (server agrees to deliver the email)

Nessus Plug-in Code Example / Horde File Disclosure

```
  script_category(ACT_ATTACK);

[.....................................]
  # Try to exploit the issue to read a file.
  #
  # nb: Horde 3.x uses "/services"; Horde 2.x,
"/util".
  foreach subdir (make_list("/services", "/util"))
  {
    if ("util" >< subdir) file = "horde.php";
    else file = "conf.php";

    r = http_send_recv3(method:"GET",
      item:string(
        dir, subdir, "/go.php?",
        "url=../config/", file, "%00:/&",
        "untrusted=1"
      ),
      port:port
    );
[.....................................]
```

horde_url_file_disclosure.nasl

Actually exploit the vulnerability to see if *go.php* allows accessing arbitrary files (horde's config files)

Other Vulnerability Scanners

- Nikto
  - Dangerous files / CGIs, outdated versions of servers, misconfigurations in a web server
- HP WebInspect
  - Web Application Vulnerability Scanning
    - Cover Adobe Flash, JavaScript/AJAX, .Net, PHP, Cold Fusion,…

Summary

- The scanners covered thus far interact with the interfaces of a production system to detect vulnerabilities
  - Check for banners and versions
  - Look for files / settings that make system vulnerable
  - Exploit vulnerability and see if it works
- Deal with known vulnerabilities

Alternative Approaches for Vulnerability Scanning

- Static code analysis for vulnerabilities
  - http://en.wikipedia.org/wiki/List_of_tools_for_static_code_analysis
  - http://www.armorize.com/
- Software testing
  - Black-box / grey-box / white-box testing
  - http://en.wikipedia.org/wiki/Software_testing#Testing_methods
- Can identify “potential” vulnerabilities