

CS 378 - Network Security and Privacy
Spring 2009

Homework #2

Due: 2pm CDT (in class), April 23, 2009

YOUR NAME: _____

Collaboration policy

No collaboration is permitted on this assignment. Any cheating (*e.g.*, submitting another person's work as your own, or permitting your work to be copied) will automatically result in a failing grade. The Computer Sciences department code of conduct can be found at <http://www.cs.utexas.edu/academics/conduct/>

Late submission policy

This homework is due at the **beginning of class** on **April 23**. All late submissions will be subject to the following policy.

You start the semester with a credit of 3 late days. For the purpose of counting late days, a "day" is 24 hours starting at 2pm on the assignment's due date. Partial days are rounded up to the next full day. You are free to divide your late days among the take-home assignments (3 homeworks and 2 projects) any way you want: submit three assignments 1 day late, submit one assignment 3 days late, *etc.* After your 3 days are used up, no late submissions will be accepted and you will automatically receive 0 points for each late assignment.

You may submit late assignments to Vitaly Shmatikov (TAY 4.115C — slide under the door if the office is locked). **If you are submitting late, please indicate how many late days you are using.**

Write the number of late days you are using: _____

Homework #2 (50 points)

Problem 1 (4 points)

Give a short snippet of C code which contains a single call to a `libsafe`-protected `strcpy`, and yet is vulnerable to a buffer overflow attack as a result of this call.

Problem 2 (5 points)

Program shepherding is a recently proposed method for preventing control flow hijacking (*e.g.*, overwriting the return address on the stack with the address of the attack code). Because it is very difficult to prevent memory corruption caused by buffer overflow, program shepherding instead prevents the transfer of control to the malicious code. In particular, it prevents execution of code masquerading as “data” (*e.g.*, stored on the stack) and ensures that library functions can only be entered through their declared entry points.

You can learn more about program shepherding from this paper:

<http://www.cag.lcs.mit.edu/rio/security-usenix.pdf>

Give at least **three** buffer overflow-based exploits that program shepherding would *not* prevent.

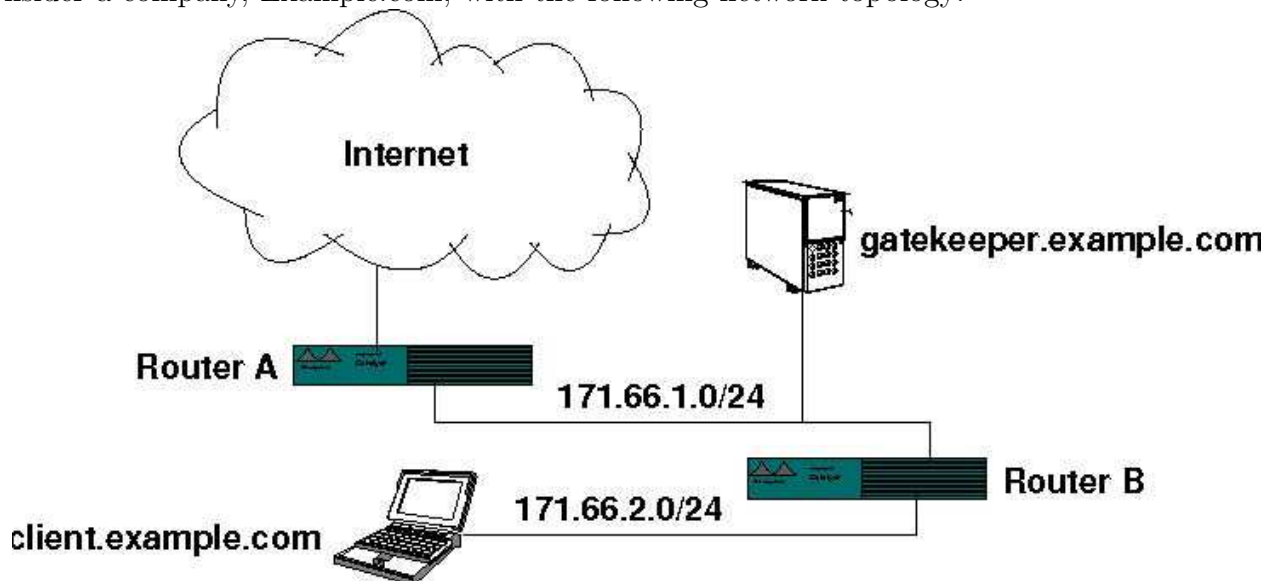
Problem 3 (6 points)

Using a rule syntax presented in class, write the rules for a packet-filter firewall which jointly implement the following security policy:

- All incoming connections are blocked, except...
- ...incoming TCP connections to SMTP on mail server (IP address 1.1.1.1).
- ...incoming TCP connections to HTTP and HTTPS on Web server (IP address 1.1.1.2).
- ...incoming TCP connections to FTP server (IP address 1.1.1.3).
- All outbound connections are allowed.
- Telnet access is blocked (because it sends unencrypted passwords).

Problem 4

Consider a company, Example.com, with the following network topology:



Example.com is worried that a Trojan infecting one of its internal machines may steal the source code of their product and send it outside the internal network. Therefore, every employee is issued a small hardware authentication device. Any communication to the outside world must be authenticated by a human typing in a security code computed by and displayed on this device.

To enforce the policy, the administrators set up a single machine, `gatekeeper.example.com`, that can talk both to internal company machines and to the rest of the Internet. Employees can log into `gatekeeper` from internal machines using SSH and their hardware authentication device. From `gatekeeper`, they can SSH to the rest of the Internet. Internal machines are on a separate subnet (`171.66.2.0/24`) and can exchange packets with `gatekeeper` but not with the outside world. Machines on the outside Internet should not be able to SSH to `gatekeeper`.

Problem 4a (5 points)

Describe how to enforce this policy with stateless packet filtering on Router A and/or Router B. Describe the precise packet filtering rules you would put in place at each router.

Problem 4b (4 points)

After several days of this new policy, employees become annoyed that many applications seem to lock up for periods of a minute or so. It is suspected that the problem is caused by attempts to create TCP connections to the outside world, which instead of failing instantly take approximately one minute. After all, clients' TCP implementations treat packets dropped by the firewall policy just the same as packets dropped because of congestion—they back off and keep trying.

To solve the problem, the administrators re-configure their routers not just to drop packets silently, but in certain cases to send packets back to the source of a dropped packet. Describe precisely what the routers can send back to make prohibited outgoing TCP connections fail quickly. (Assume they cannot make any changes to the TCP implementation)

on clients.)

Problem 4c (4 points)

After the fix from the previous part, things improve somewhat, but applications are still locking up because DNS lookups to the outside world are also taking a long time to fail. To solve the problem, the administrators run a caching resolver on gatekeeper and configure all internal clients to use gatekeeper as their DNS nameserver. The administrators figure that since DNS is a read-only protocol, it is safe to allow internal machines to query for IP addresses of hosts anywhere on the Internet, as long as any actual communication to those IP addresses is blocked by the routers.

Explain how a clever Trojan with access to the secret source code on `client.example.com` can collude with another machine on the Internet to leak the source code, even without access to the hardware authentication device.

Problem 5 (4 points)

Acme Security Inc. is promoting `Hackz-Be-Gone`, the universal anti-virus software. Acme's marketing literature asserts that `HBG` can correctly determine whether a program is a virus. More precisely, given any program `P`, `HBG(P)` returns *True* if and only if `P`, when executed, will replicate itself and infect other executables on the host machine.

Do you believe them? If no, write a program `P` on which `HBG` will *always* produce the wrong answer. If yes, how do you think they are doing it? Assume that `P` can execute `HBG` on any input.

Problem 6

The Random Constant Spread (RCS) model has been proposed for mathematically describing the spread of worms in the Internet. Let $\alpha(t)$ be the fraction of all vulnerable machines compromised at time t . The RCS model posits that the spread of a random-scanning worm is governed by the differential equation $\frac{d\alpha}{dt} = K\alpha(1 - \alpha)$, where K is the “infectiousness” factor. This equation says that the initial infection compromises K other machines at time 0, and at time t each infection compromises $K(1 - \alpha(t))$ other machines. The closed-form solution of the above equation is:

$$\alpha(t) = \frac{e^{K(t-T)}}{1 + e^{K(t-T)}}$$

where T is the constant of integration.

Problem 6a (4 points)

If $T = 12$ and $K = 1.8$, at what time are half of all vulnerable machines infected? At what time are all machines infected?

Problem 6b (4 points)

Now suppose $K = 0.9$. At what time are half of all vulnerable machines infected? At what time are all machines infected? What is different from the scenario in Problem 6b?

Problem 7 (10 points)

Acme Security Inc. has also developed a new host-based intrusion detection systems. For the purposes of this problem, assume that the following three possibilities are mutually exclusive: either the host is normal, or rookit-infected, or spyware-infected. Assume that 1% of all hosts are infected with rootkits, 2% with spyware, and the remaining 97% are normal.

Acme computed the following accuracy rates for its product:

Actual state of the host	How the host is diagnosed		
	Rootkit	Spyware	Normal
Rookit-infested	85%	5%	10%
Spyware-infested	5%	90%	5%
Normal	5%	5%	90%

For example, when Acme analyzes a rookit-infected host, it correctly classifies the infection with probability 85%, misclassifies it as spyware with probability 5%, and misclassifies the host as normal with probability 10%.

When Acme announces that a host is infected with spyware, what is the probability that this is a false positive? Give your calculations.