CS 161 Computer Security
Fall 2005 Joseph/Tygar/Vazirani/Wagner MT 2 Soln

Problem 1. [Firewalls and Network Threats] (30 points)

List and **explain** three network threats that a firewall does not protect against. (If a threat only applies to certain types of firewalls, then explain why this is the case.)

Sample threats: (1) Attacks against open ports, such as buffer overrun attacks against unblocked services; (2) Malicious code or attacks carried in email or web traffic (many firewalls do not scan or examine email and web payloads); (3) Attacks on the firewall itself (e.g., trying to penetrate the firewall code by exploiting a buffer overflow in the firewall's packet parsing code); (4) Internal attacks by malicious insiders; (5) Attacks from compromised internal machines against other internal machines (e.g., a laptop becomes infected with a worm, which tries to infect other inside hosts)—applies to perimeter firewalls; (6) Attacks from compromised machines which have a VPN or other tunnel through the firewall—applies to perimeter firewalls; (7) Denial of service attacks against the network link or the firewall itself.

Grading: 10 point per threat, broken down as 5 points for the threat and 5 points for the explanation.

Problem 2. [Zero-Knowledge Proofs] (20 points)

Let (N, e) be Alice's RSA public-key and (N, d) be her private key. Suppose that Bob claims to have a signed message from Alice: he claims to have $s = m^d \bmod N$ for some particular $m \bmod N$ (which he reveals). Bob wishes to prove to Charlie that he has this signed message, without revealing any information about s. The following are the first two steps in a protocol by which Bob can provide a zero-knowledge proof of knowledge about s:

- Bob selects a random number $r \bmod N$ and computes $t = r^e \bmod N$. He sends $t \bmod N$ to Charlie.
- Charlie randomly chooses one of two challenges: I) He asks Bob to send him Alice's signature on t, namely $t^d \bmod N$. II) He asks Bob to send him Alice's signature on $m \cdot t$, namely $(m \cdot t)^d \bmod N$.

1. Fill in the last two steps of the protocol. i.e. how does Bob respond to each challenge. And what should Charlie do to check each response.

    - *Bob sends I) r or II) $r \cdot s \bmod N$, according to Charlie's challenge.*
    - *Charlie checks that I) $r^e = t \bmod N$. II) $(r \cdot s)^e = t \cdot m \bmod N$.*

    Grading: 8 points, broken down as 2+2 for what Bob sends (cases I+II) and 2+2 for what Charlie checks. No credit for telling Bob to send $t^d \bmod N$ or $(m \cdot t)^d \bmod N$ (Bob doesn't know d).

2. This protocol is zero knowledge, in the sense that even a cheating verifier gets no information about the original signed message s. Recall that the key step in proving this is showing that there is a simulator who, *without* knowledge of s, can create the transcript of Charlie's interaction with Bob with probability 1/2 regardless of which of the two challenges Charlie issues. Show how the simulator can achieve this goal.

> - *The simulator flips a fair coin to guess whether the verifier will ask for I or II in the third message, picks a random number* $r \bmod N$, *and sends to the verifier: I)* $r^e \bmod N$ *or II)* $r^e \cdot m^{-1} \bmod N$ *(choosing between the two according to its coin flip).*
> - *The simulator receives the verifier's challenge. If the simulator guessed the challenge incorrectly, give up (this happens with probability* $1/2$*). Otherwise, continue.*
> - *The simulator sends r to the verifier.*
> - *Finally, the simulator outputs the transcript of its interaction with the verifier (assuming it hasn't given up).*
>
> *Grading: 12 points. 6–7 points for noticing that you can answer both challenges, if you know in advance which challenge you will be given. 0 points for always sending* r^e *and giving up or rewinding if the verifier asks for challenge II (a dishonest verifier might always for challenge II).*

Problem 3. [Firewall Deployments] (30 points)

Explain the strengths and weaknesses of each of the following firewall deployment scenarios in defending servers, desktop machines, and laptops against network threats.

(a) A firewall at the network perimeter.

> *Example strengths: (1) Mediates all incoming traffic from external hosts and can protect against many attacks by outsiders; (2) Easier to manage and to update policies, because of single central location; (3) Protects against some kinds of DoS attacks launched from the outside.*
>
> *Example weaknesses: (1) No protection against malicious insiders; (2) No protection for mobile laptops while they are connected to other networks; (3) No protection if laptops get infected while travelling and then spread infection when they re-connect to our internal network.*
>
> *Grading: 7 points total, broken down into 3 points for naming at least one valid strength, 4 points for at least one valid weakness.*

(b) Firewalls on every end host machine.

Example strengths: (1) Protects against malicious insiders and infected internal machines as well as outside attackers; (2) Protects laptops even while they are travelling and connected to other networks; (3) May be easier to customize firewall protection on a per-machine basis.

Example weaknesses: (1) Potentially more difficult to manage policies, due to the number of machines whose rulesets must be configured and updated; (2) Uncooperative users may be able to modify settings or disable firewalls on their own machines, and viruses/worms may be able to do the same to machines they infect; (3) Potentially less resistant to DDoS, since DoS attacks can still flood internal network links; (4) Depending upon firewall configuration, may block legitimate internal traffic and/or make some internal services harder to use.

Grading: Same as (a).

(c) A network perimeter firewall and firewalls on every end host machine.

Example strengths: (1) Layered defense provides redundancy in case one firewall fails; (2) Can easily update policy against external attacks if a new threat develops, which gives some time to update the rulesets on internal hosts. See also strengths (a)(1) and (b)(1)–(3).

Example weaknesses: (1) Potential for overblocking of legitimate traffic, since traffic flows only if permitted by both firewalls. See also weaknesses (b)(1), (b)(4).

Grading: 6 points, 3 points for at least one valid strength, 3 points for at least one valid weakness.

Problem 4. [Classified Computing] (20 points)

(a) List two examples of covert channels, **other** than the three examples given in the lecture notes: existence of a file, system paging behavior, and system load. Explain how an adversary could take advantage of each of your examples.

Examples: (1) Number of pending jobs in print queue (e.g., send a 0 bit by printing nothing, a 1 bit by printing many documents); (2) Timing of locks or shared resources (e.g., sender: 0 = do nothing, 1 = acquire lock or resource); (3) Disk access latency (e.g., 0 = do nothing; 1 = issue many disk writes); (4) Presence/absence of a network packet (e.g., 0 = do nothing; 1 = visit a web site).

Grading: 10 points total, 5 points per covert channel, broken down as 3 points for naming a channel (e.g., a method of communication), 2 points if it is covert (not overt).

We also accepted side channels, although strictly speaking a covert channel usually represents deliberate communication (sender and receiver are two malicious parties colluding to transmit data) whereas a side channel usually refers to unintentional leakage (sender is honest but unintentionally leaks secrets; receiver is malicious).

(b) Two professors are running applications on a classified multi-user system. Professor Tygar is running the Quake game, and Professor Wagner is running a Top Secret application. Who should get higher priority on a multi-user machine? Explain your answer.

Valid answer #1: Tygar should receive higher priority, to prevent the system load from being used as a covert channel (otherwise the speed at which Quake runs depends on Wagner's behavior, which means that Wagner could leak secrets to Tygar).

Valid answer #2: Both receive a fixed percentage of system resources, to prevent the system load from being used as a covert channel. For example, Quake always receives exactly 50% of CPU time, whether or not Tygar is using the system at the time.

Grading: 10 points total, 5 points for a correct statement of who gets which priority, 5 points for explaining why (to prevent system load from being used as a covert channel).

(c) Why is it difficult to implement systems supporting covert channel prevention that perform well? Explain your answer.

Every resource that is shared among multiple users represents a possible covert channel. Pre-allocating such resources with a fixed schedule leads to a loss of performance; while trying to dynamically multiplex access to such resources on the fly in a way that leaks nothing is difficult. Also, there are many shared resources, and it is hard to identify them all.

Grading: 10 points for a full answer. Partial credit for several common answers.