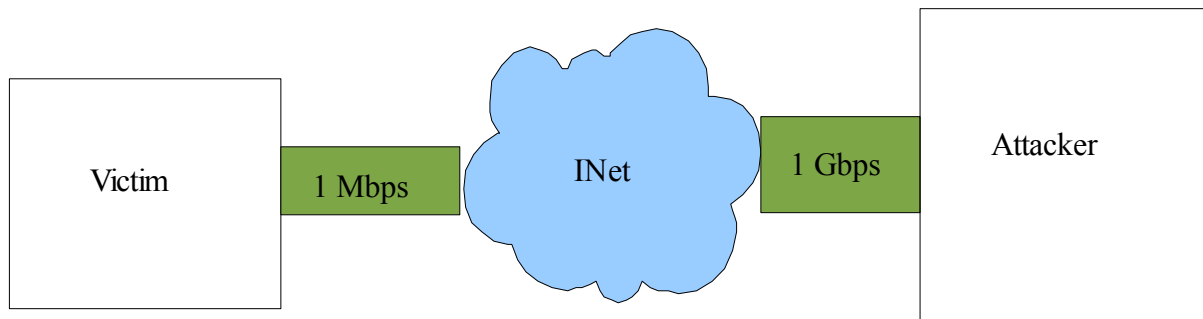


DENIAL OF SERVICE ATTACKS (Lecture on 4/17/07)

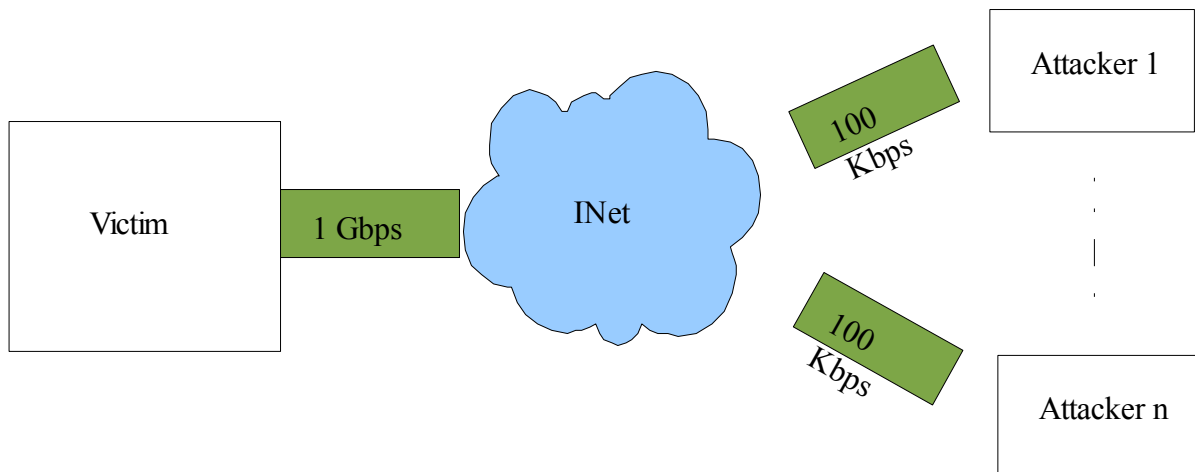
- Crashing Program
- Resource Exhaustion
 - Network Bandwidth
 - Memory
 - OS Resources (File Descriptors)
 - Disk
 - CPU

Some Network DoS attacks:



The attacker floods the victim with requests that overwhelm the victim's b/w resource

When victim has greater b/w than the attacker

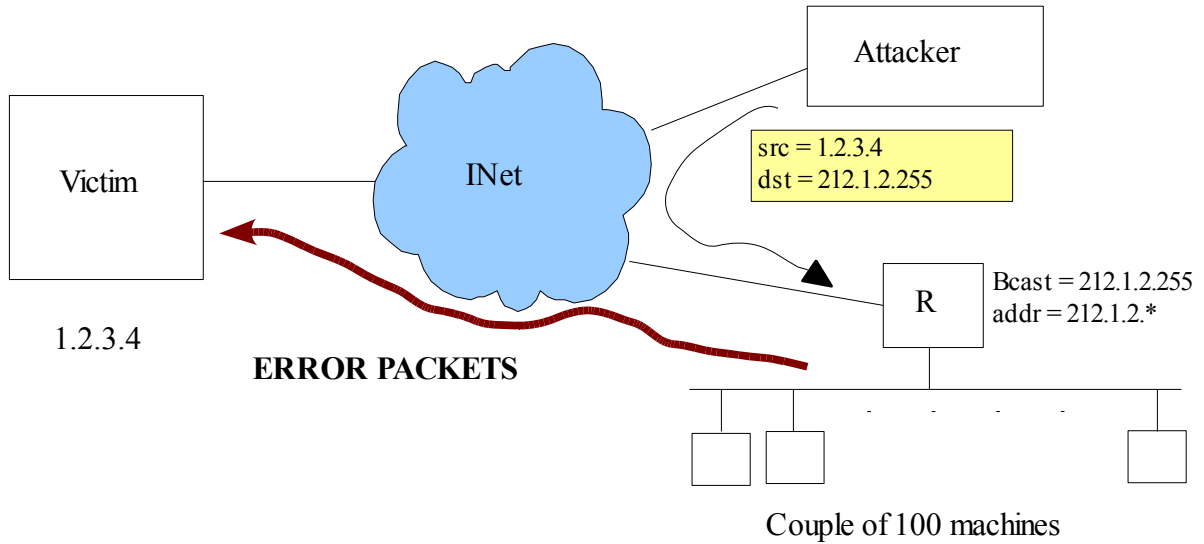


The attacker machines (also called zombies, bots or slaves) are obtained by exploiting some vulnerability and spreading worms

NOTE: More attacker machines => harder to distinguish attacker

An 'Old School' attack – SMURF

In all the bandwidth DoS attacks the attacker needs to somehow escalate its bandwidth to beat the victim (if the attacker doesn't have sufficient b/w). SMURF was an old trick that no longer works thanks to good network administration policies which do not allow broadcast packets from hosts outside the LAN.

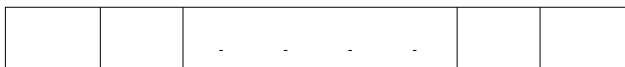


SMURF attack: Gives 200 times bandwidth !

CPU DoS

- Algorithmic Complexity Attacks

Example: Hash Table



$$i = \text{hash}(x)$$

append(hashtable[i], x)

- hash function is typically fixed and public
- easy for an attacker to find collisions

Attack:

1. Choose many inputs x_1, x_2, \dots, x_n that all hash to 0.
2. Interact with the server to cause it to insert x_1, x_2, \dots, x_n into its hash table
3. Force the server to do "H.T. Lookups".

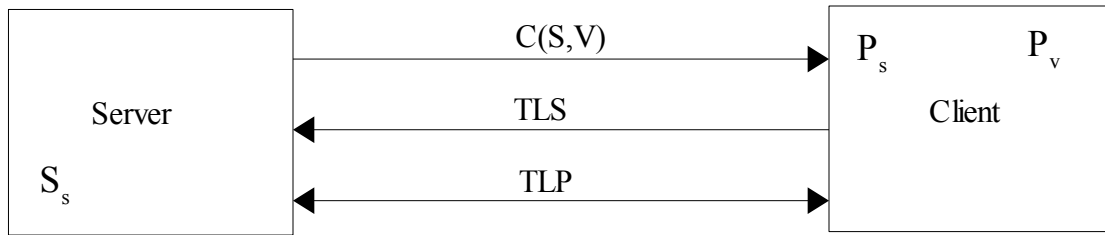
Idea is to bridge the gap between average case complexity and the worst case complexity.

A similar attack can be devised for binary trees (making them unbalanced)

Solution: Use a randomized hash function (that takes a random seed upon initialization).

[READ Universal Hash Functions]

TLS/SSL (CPU DoS)

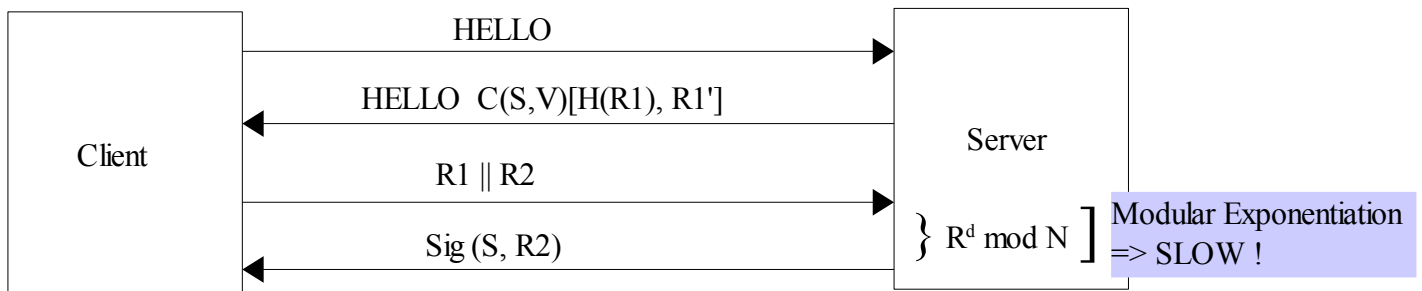


v : Could be a third party such as Verisign

$C(S,V)$: Certificate of the server

TLS Guarantees:

- Secrecy
- Authenticity
- Server Identity



Client:

1. Verify Certificate and find $R1$
2. Send $R1$ and $R2$ (a new random number for signature verification)
3. Verify signature

$R1'$ is the first few bits (say 64) of $R1$.

- Modular Exponentiation in software
- Typical Desktop ≈ 100 modular exponentiations per sec
- Couple of 100 challenges ($R2s$) and server slows down a lot.

NOTE: $R1$ is optional (for backward compatibility)

CLIENT PUZZLES

Problems:

- Doesn't distinguish real from attacks
- Must be configured per system
- Discriminate against slow clients

With puzzles the attacker needs to have at least as much MIPS as the server for launching an attack.