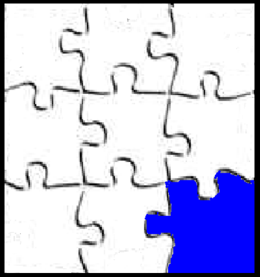
Client Puzzles

A Cryptographic Defense Against Connection Depletion Attacks

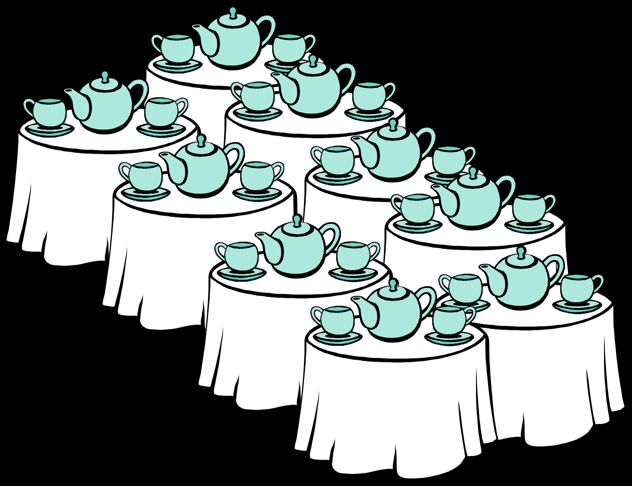


Ari Juels and John Brainard
RSA Laboratories

Connection Depletion: The Problem

How to disable a restaurant

Restauranteur



Saboteur

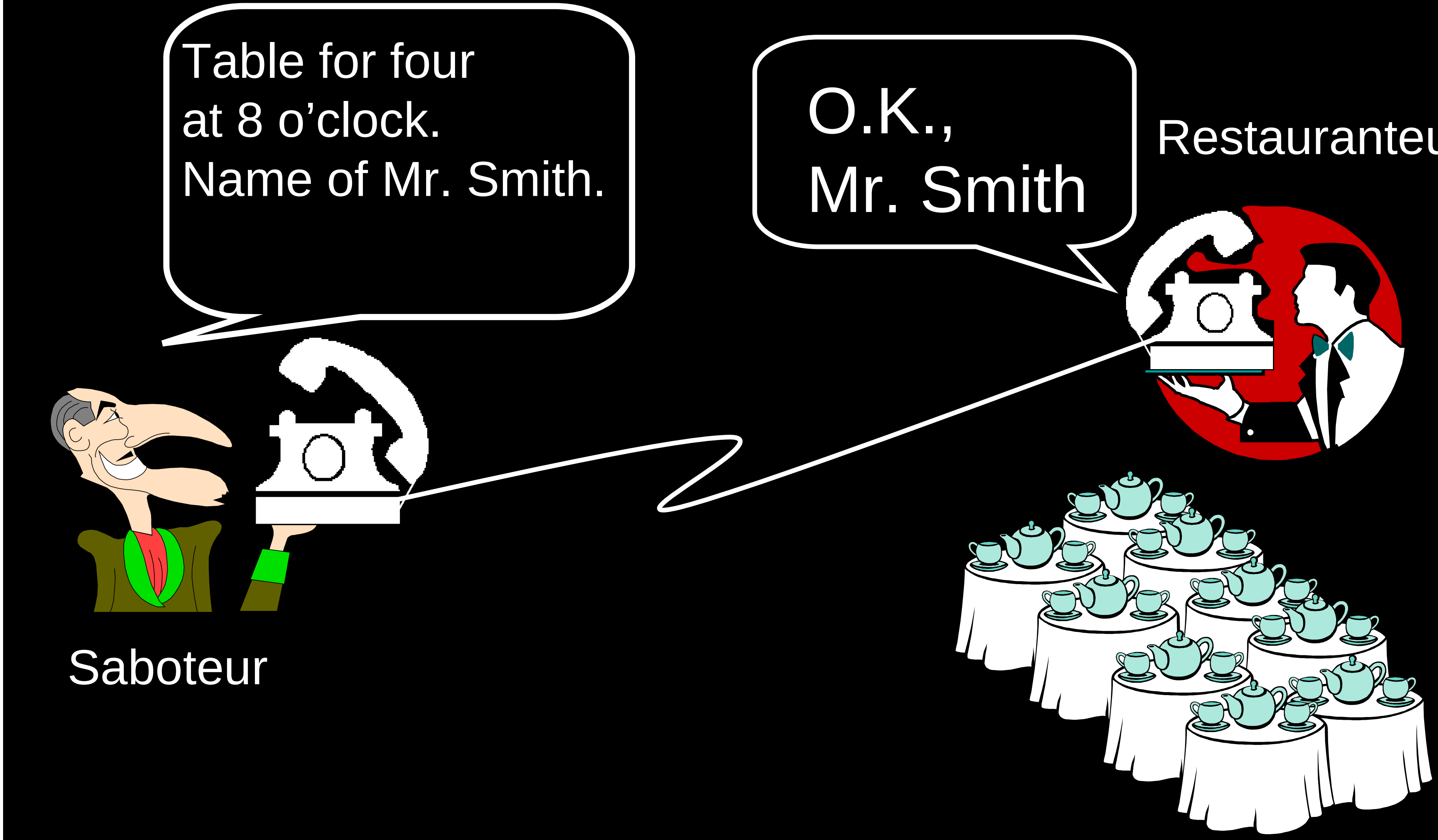


Table for four
at 8 o’clock.
Name of Mr. Smith.

O.K.,
Mr. Smith

Restauranteur

Saboteur

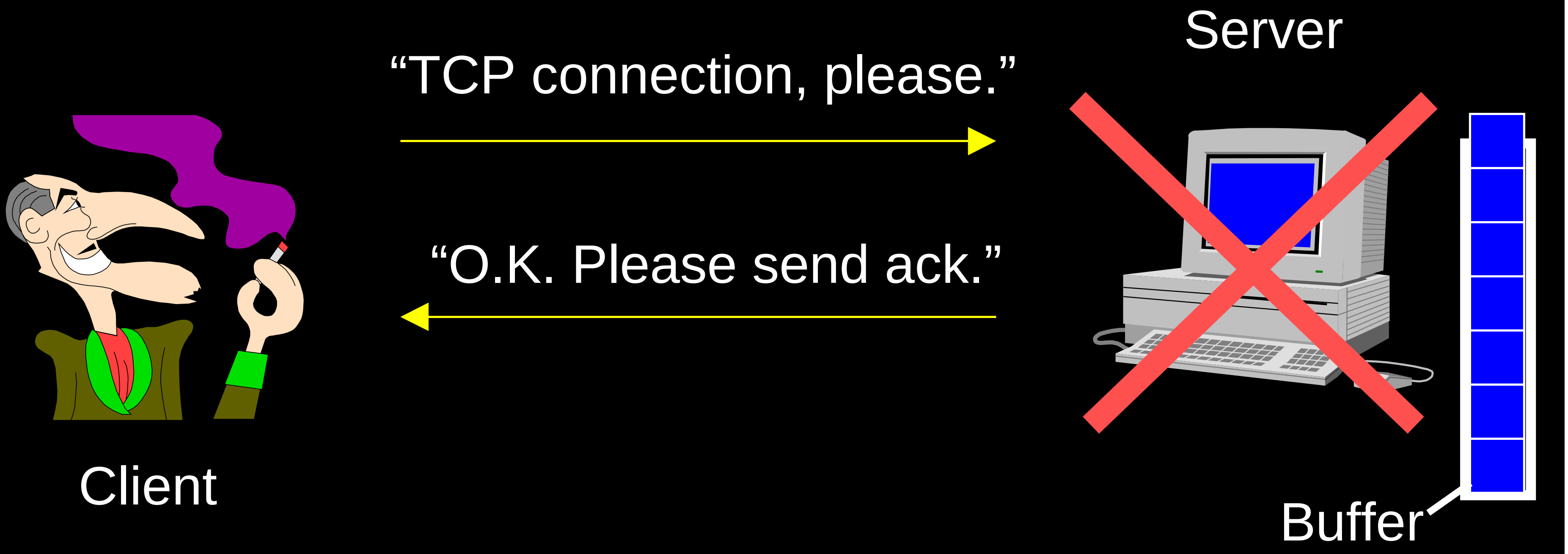
Saboteur vs. Restauranteur

Restauranteur

Saboteur

No More Tables!

An example: TCP SYN flooding



Server

“TCP connection, please.”

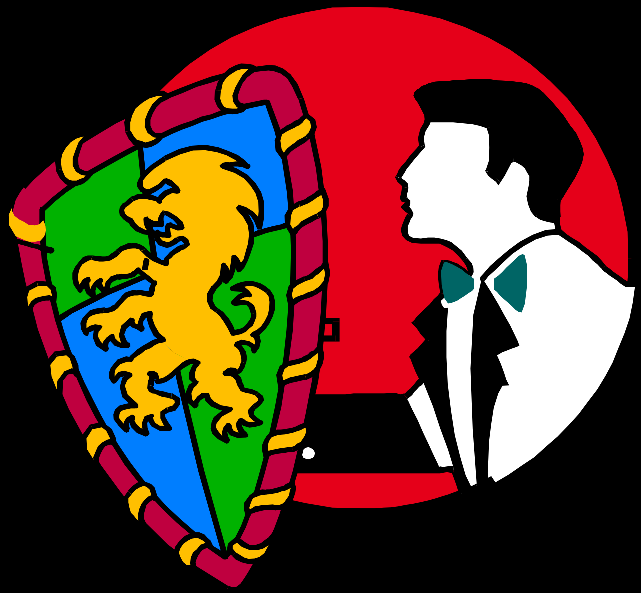
“O.K. Please send ack.”

Client

Buffer

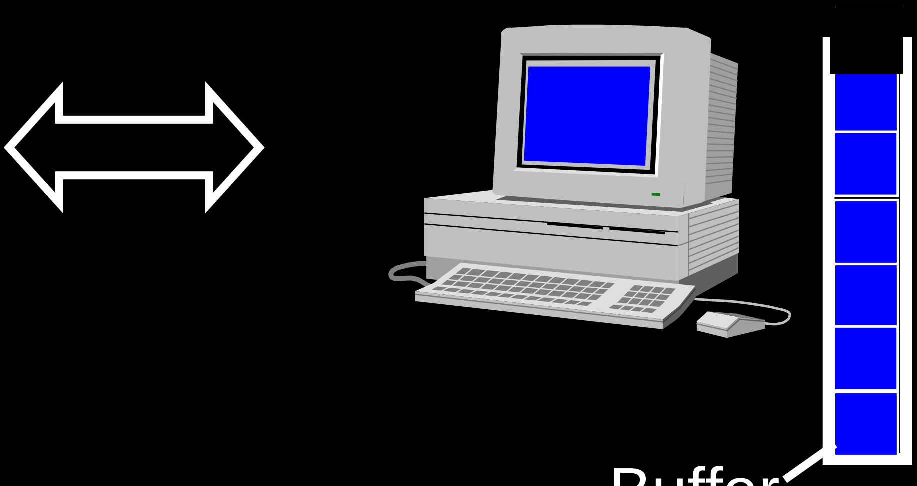
- **TCP SYN is a real-world problem**
  - **Panix, mid-Sept. 1996 (NYT)**
  - **New York Times, late Sept. 1996**
  - **Others**
- **Similar attacks may be mounted against e-mail, SSL, etc. -- resources other than memory**

Some defenses against connection depletion



IP Tracing (or Syncookies)

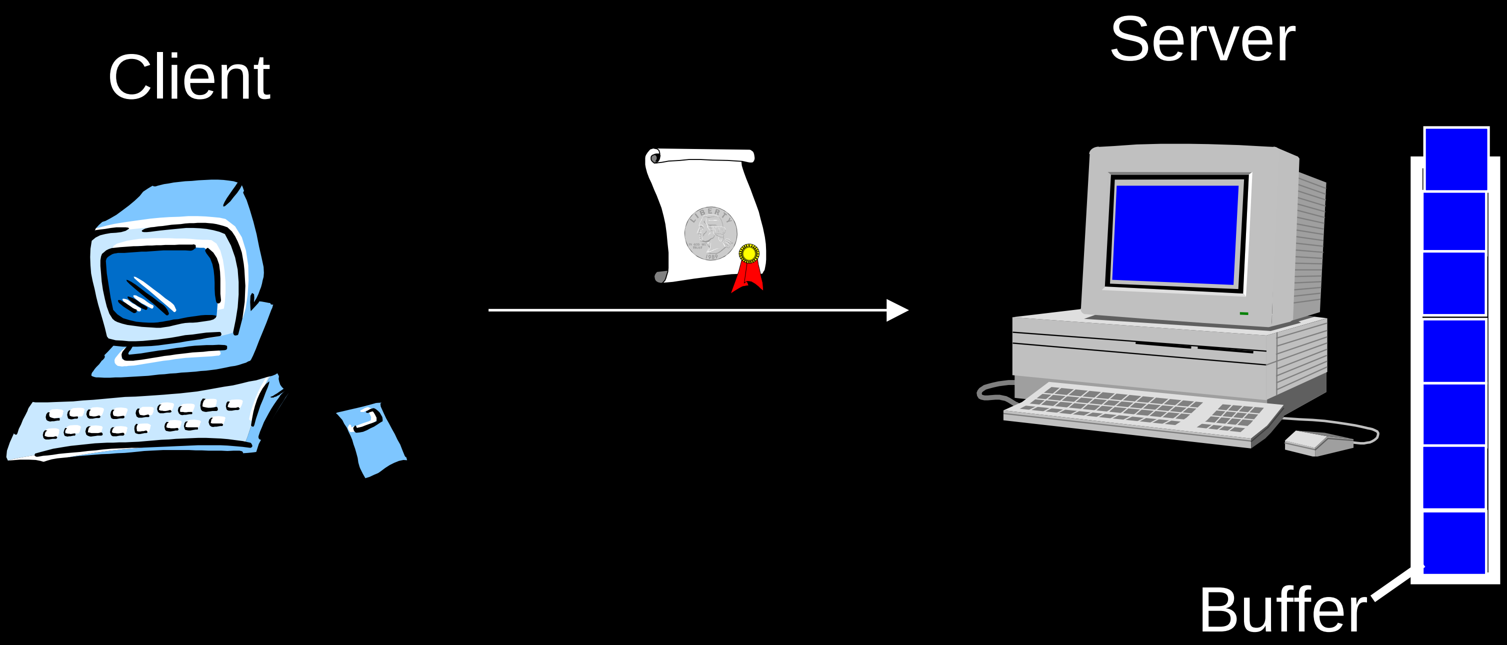
Hi. My name is 10.100.16.126.



Problems:

- Can be evaded, particularly on, e.g., Ethernet
- Does not allow for proxies, anonymity

Digital signatures



Client

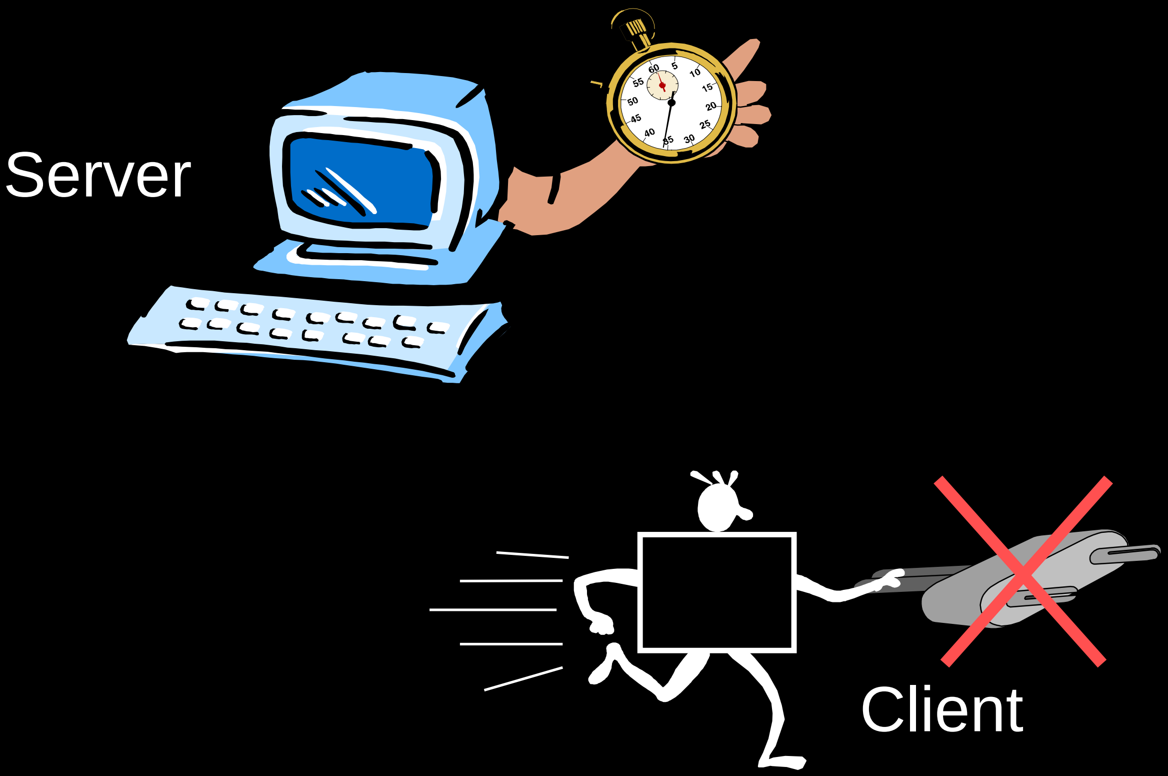
Server

Buffer

Problems:

- Requires carefully regulated PKI
- Does not allow for anonymity

Connection timeout (for buffers)

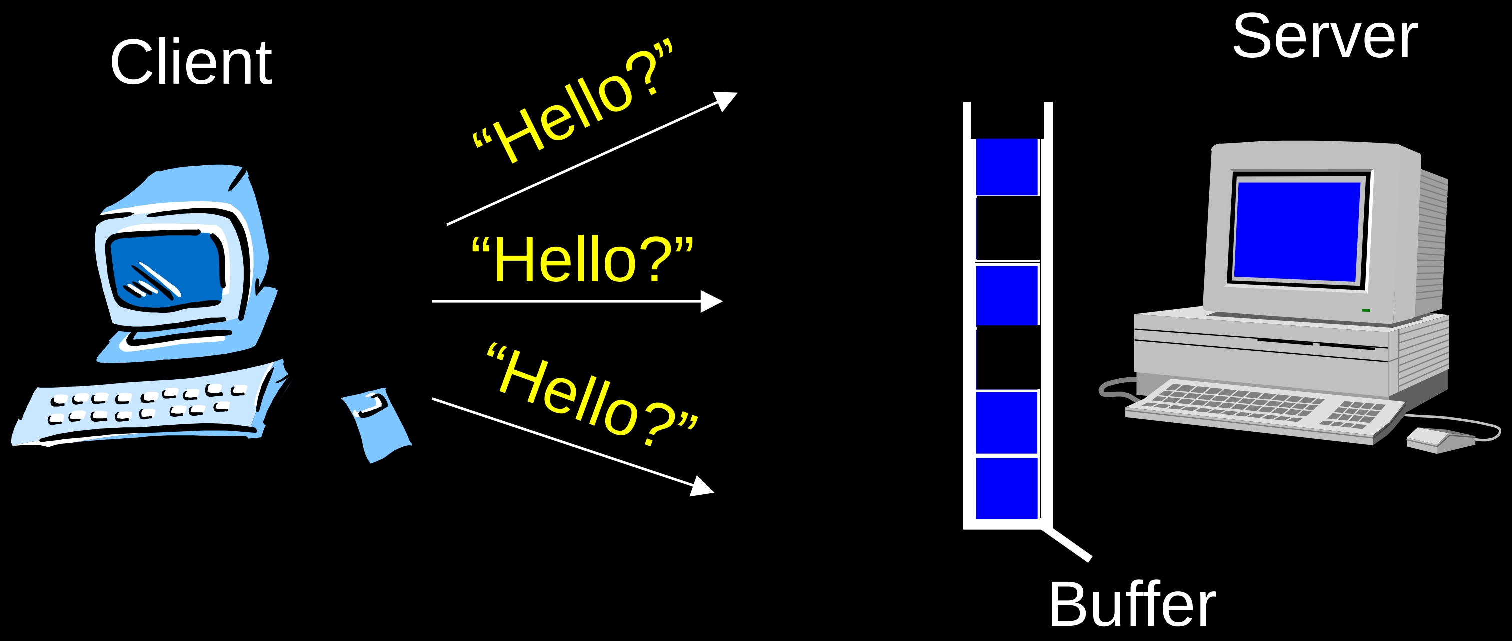


Server

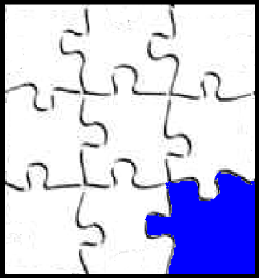
Client

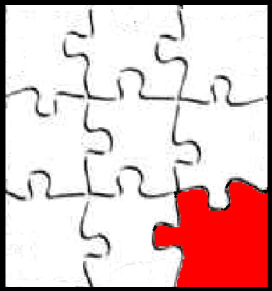
Problem: Hard to achieve balance between security and latency demands

Throw away requests at random

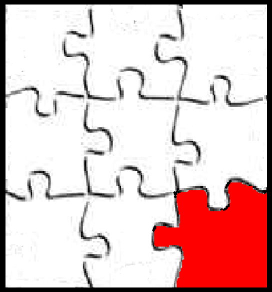


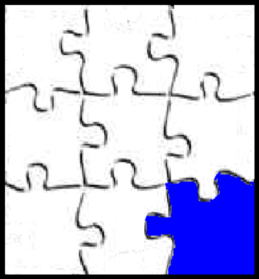
Problem: Legitimate clients must keep retrying in high volume attacks



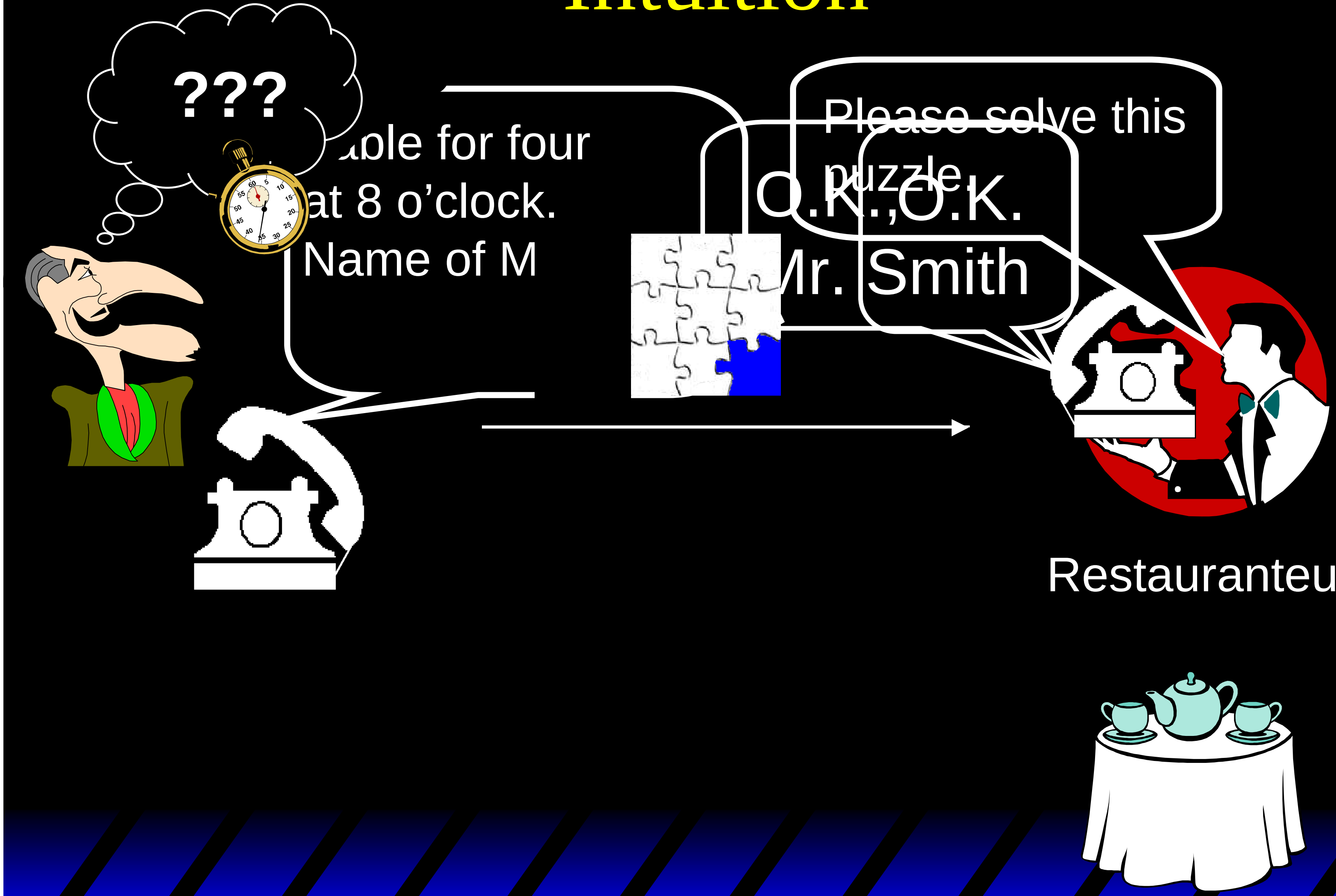


Our solution: *client puzzles*





Intuition


???
able for four
at 8 o'clock.
Name of M
Please solve this puzzle.
O.K.,O.K.
Mr. Smith
Restauranteur

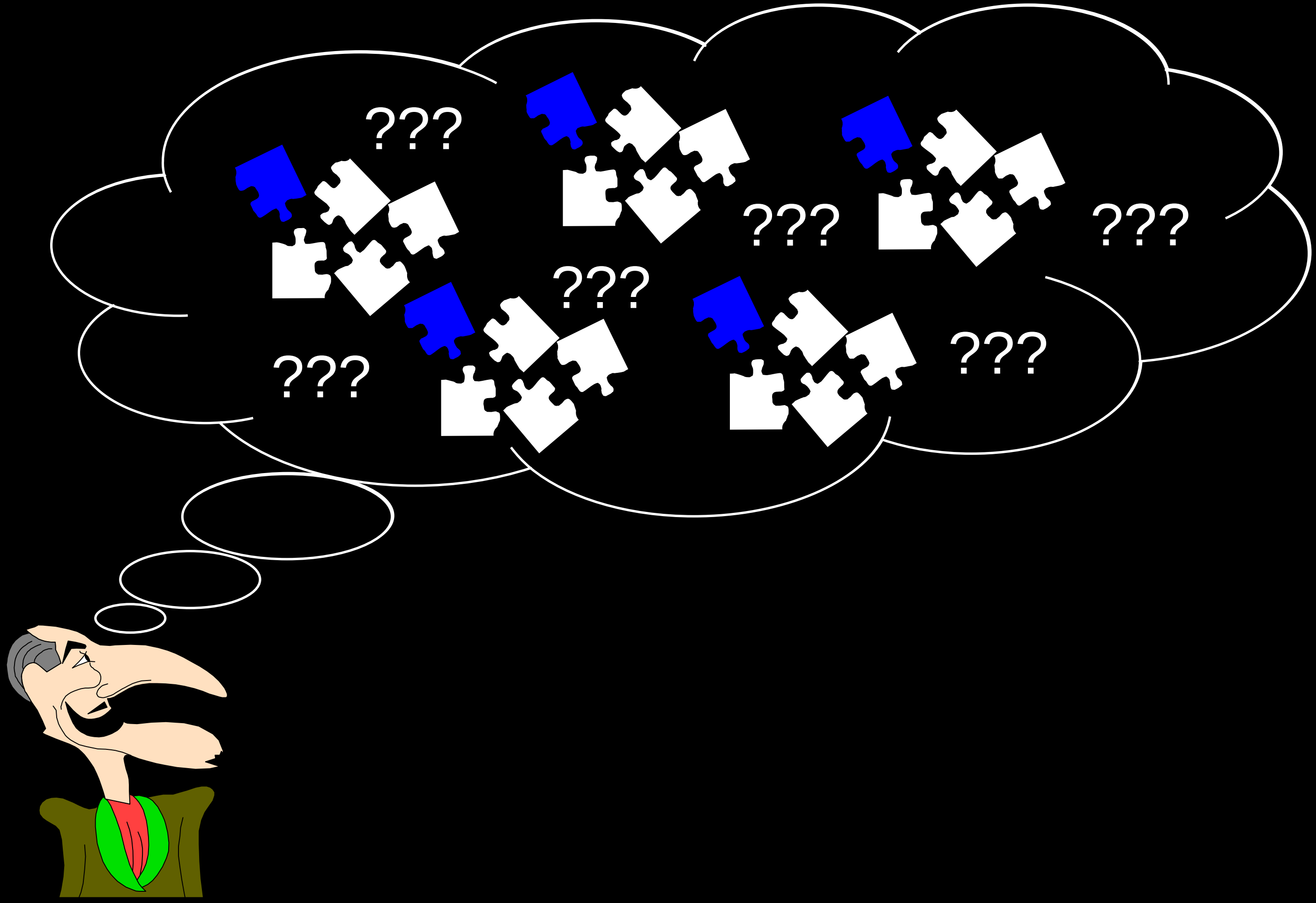
Intuition

Suppose:

- A puzzle takes an hour to solve
- There are 40 tables in restaurant
- Reserve at most one day in advance

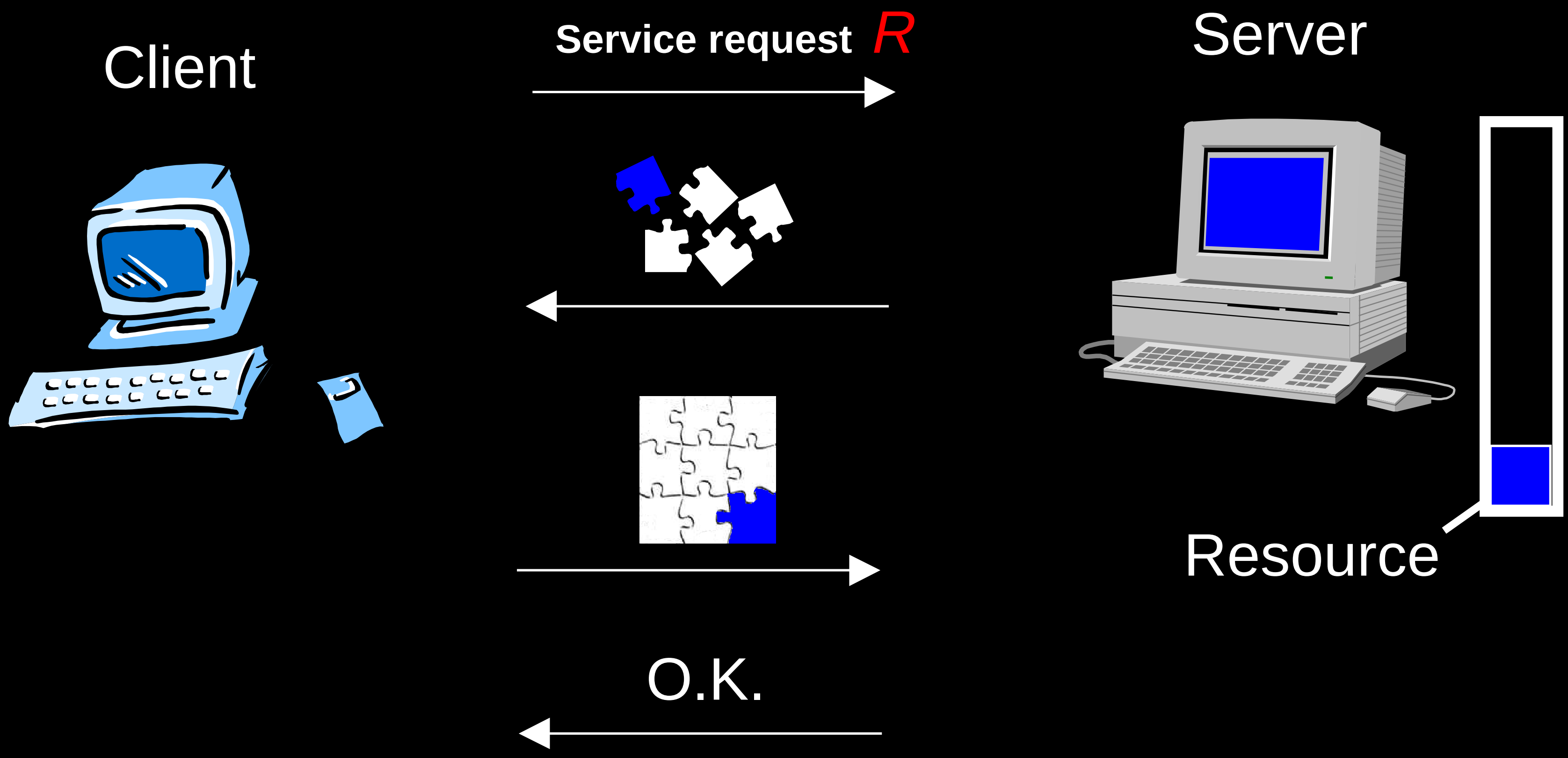
A legitimate patron can easily reserve a table, but:

Intuition



Would-be saboteur has too many puzzles to solve

The client puzzle protocol



Client

Service request R

Server

Resource

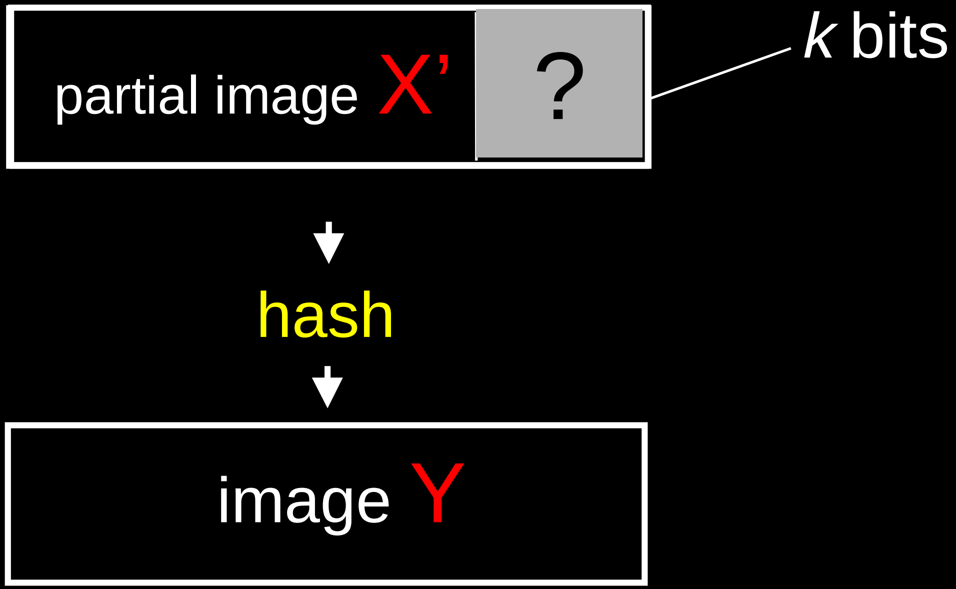
O.K.

Remarks

- Can use puzzles for any type of resource
- Only have to distribute puzzles when under attack
- Can scale hardness of puzzles depending on severity of attack

What does a puzzle look like?

Puzzle basis: partial hash inversion



partial image X’

?

k bits

hash

image Y

Pair (X’, Y) is *k*-bit-hard puzzle

Puzzle construction

Client

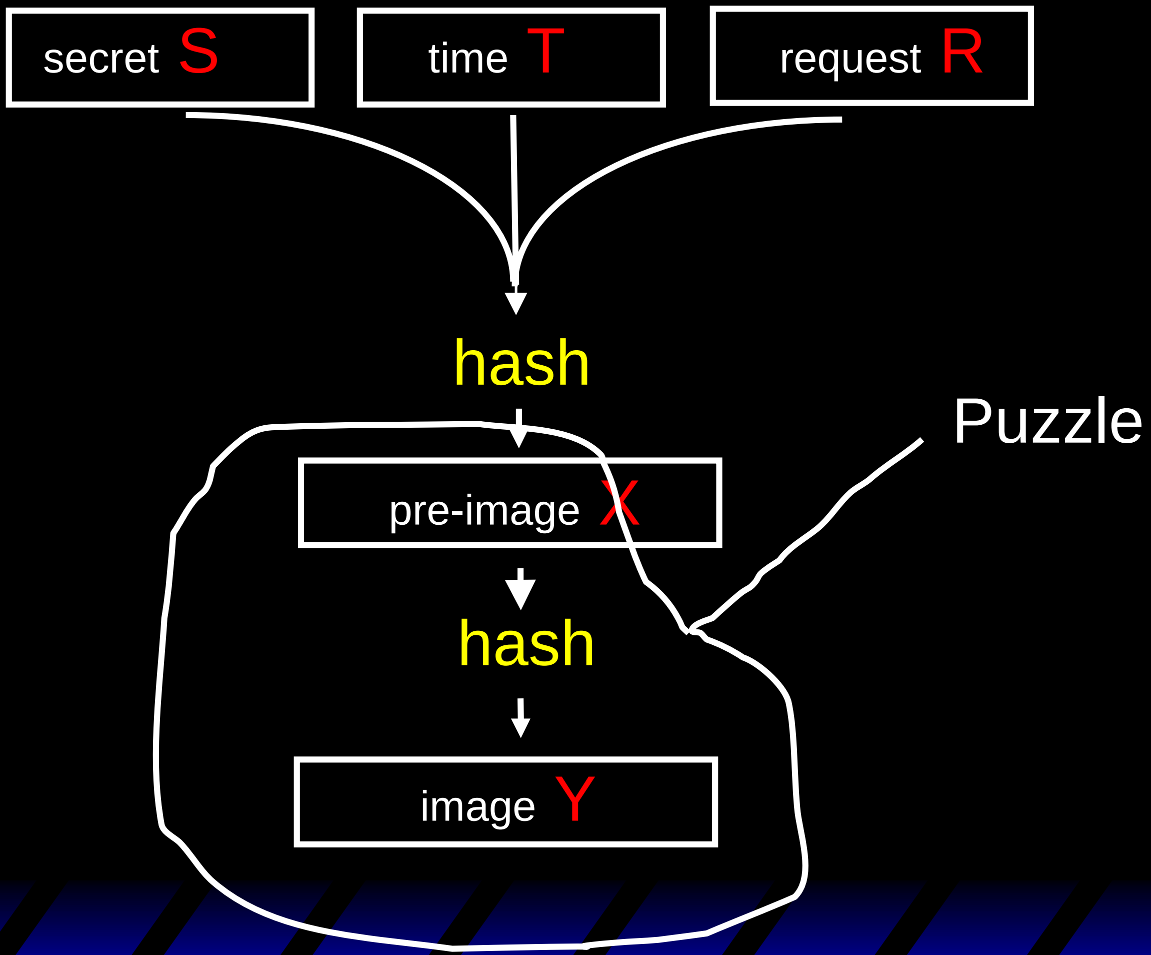
Server

Service request R

Secret S

Puzzle construction

Server computes:



secret S

time T

request R

hash

pre-image X

hash

image Y

Puzzle

Puzzle properties

- Puzzles are stateless (client provides *T* and *R* with puzzle)
- Puzzles are easy to verify
- Hardness of puzzles can be carefully controlled
- Puzzles use standard cryptographic primitives

Where to use client puzzles?

Some pros

Avoids many flaws in other solutions, e.g.:

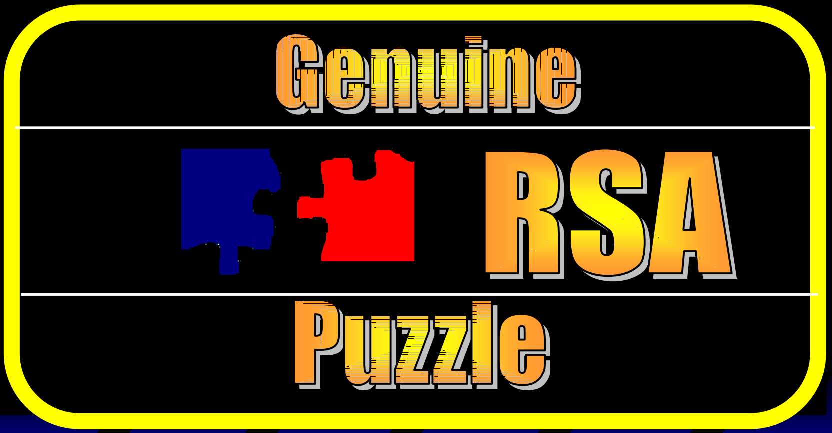
- Allows for anonymous connections
- Does not require PKI
- Does not require retries -- even under heavy attack

Drawback

- Requires special-purpose software, e.g., browser plug-in

Client puzzles seem most suitable for internal networks

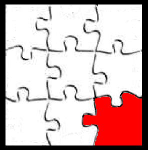
Candidate technology for RSA/Security Dynamics enterprise security servers



Conclusions

What’s in the paper

- Introduces idea of *puzzles* for on-the-fly resource access control



- Detailed puzzle and protocol description
- Discussion of overhead
  - How long to process puzzle solution?
  - How many extra tables?

Too

- Rigorous mathematical treatment of parameterization/security level
  - Solving puzzles is a probabilistic process -- attacker may get lucky
- *Protocol can be simplified and made more efficient*

More work on puzzles

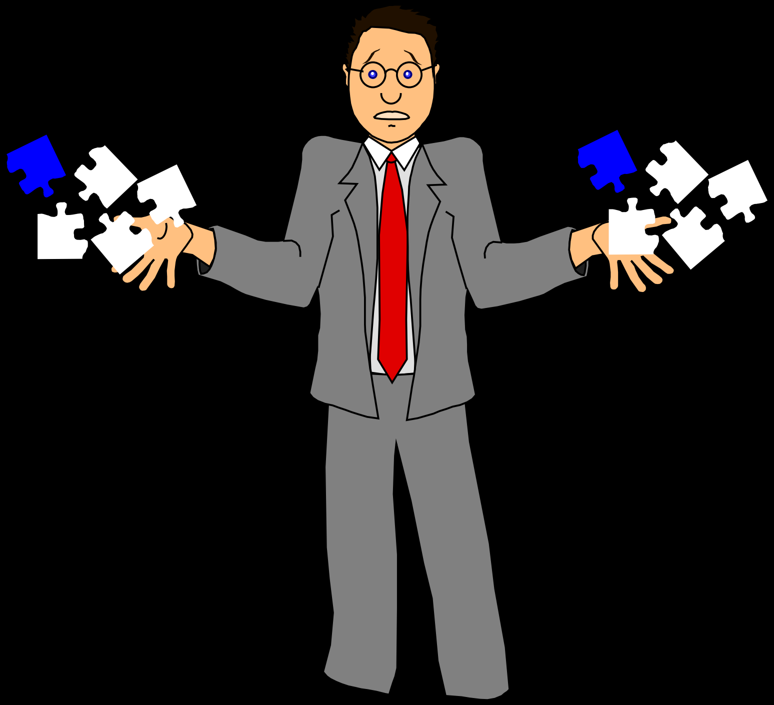
Puzzles have also been proposed for:

- Controlling spam (DN94, BGJMM98)
- Auditing server usage (FM97)
- Time capsules (RSW96)

More to be done

- How to define a puzzle? Search space vs. sequential workload
- Can puzzle construction be improved?
  - Replace hash with, e.g., reduced-round cipher
- Can puzzles be made to do *useful* work?

Questions?



e-mail: ari@rsa.com