April 4, 2005 -- Lecture 29

22C:169
Computer Security

Douglas W. Jones

Department of Computer Science

Amoeba

Andrew Tannenbaum's Amoeba System

Designed as a multicomputer OS

- *Commodity computers as components*
- *Commodity interconnect (ethernet)*
- *Abandons TCP/IP completely*

Amoeba was not designed for security

- *Innovative use of trapdoor functions*
- *cryptographic protection of capabilities*

Fundamental idea of Amoeba

Client-server communication model

Remote procedure call to server

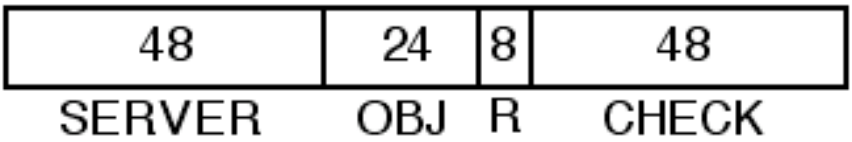
Servers implement classes of objects

An Amoeba capability packages

the right to call a particular server
with regard to a particular object
to perform a particular operation

eg: read a file using file server

Amoeba Capability Format



Server

48-bit public ID of server

Obj

24-bit id of object relative to server

R

8 bit access rights to object

Check

48-bit validity check on capability

Amoeba server addressing

Server has random private ID $ID_{PRIVATE}$

Everyone knows trapdoor function *f*

Server publishes $ID_{PUBLIC} = f(ID_{PRIVATE})$

This is the Server ID of the server!

Server says to kernel `register`($ID_{PRIVATE}$)

capabilities with server field ID_{PUBLIC}

now address this process as the server!

Amoeba message delivery

Each machine has network address cache

Mapping from ID$_{\text{PUBLIC}}$ *to location.*

On cache miss

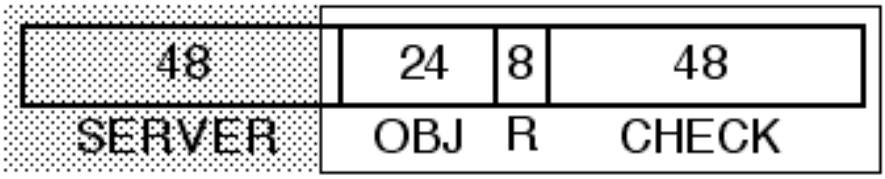
Broadcast "who has this server?"
or use registration server
or some combination.

On receipt of message with a bad ID

discard it and
report error so sender can clean cache

Amoeba server-side authentication

Most of capability belongs to the server!



Server

relates object ID to object itself

checks access rights

determines if capability is valid

Minimal server operation

Server maintains object table

object = *ObjectTable*[*capability.obj*]

Each object contains check field

valid if object.check = *capability.check*

Knowing object ID grants no access

unless correct check field is known

This scheme would be sufficient except

no support for access rights

Support for access rights

Simple scheme used if all rights are present

cap.rights = 11111111

Otherwise, valid if:

f (*cap.rights* || *obj.check*) = *cap.check*

f is a publicly known trapdoor function

Anyone may

compare capabilities

restrict rights from all rights to fewer

Only the correct server can

validate capability

restrict rights from less than all