HOW TO BREAK INTO z/OS VIA USS, TCP/IP, AND THE INTERNET

Stu Henderson (stu@stuhenderson.com)

ABSTRACT

Many mainframes are now connected to the Internet, with security that relies on integration of USS, TCP/IP, Websphere, and other daemon security, plus MVS security and software such as RACF, ACF2, or TopSecret. In this session, Stu shows you techniques to attack mainframes through this path. He also shows how to defend against such attacks, demonstrating once again that IBM gives us the most secure, flexible, and scalable UNIX, TCP/IP and web server commonly available, if only we implement the tools IBM gives us.. This session is a logical follow-on to Stu's session "How to Break into z/OS Systems".

AGENDA

- **Introduction and Background Architecture**
- **Breaking In Via USS**
- **Breaking In Via TCP/IP**
- **Breaking in Via TCP/IP Daemons**
- **Summary and Call to Action**

INTRODUCTION AND BACKGROUND ARCHITECTURE

- **IBM Has Added UNIX to the Mainframe, Under the Control of MVS. This Version of UNIX Was Originally Called OMVS, But Is Now Called USS (UNIX System Services).**

- **We Will Claim Here That It Is the Most Secure, Most Scalable, and Most Flexible UNIX Around, If You Implement It Right.**

On Top Of USS, IBM Has Added:

- **TCP/IP (Transmission Control Protocol / Internet Protocol), the Standard Communications Protocol for UNIX and for the Web**

- **And On Top of TCP/IP, Several Other Programs Which Are Paths Into Your System, Including FTP (File Transfer Protocol), Websphere (a Web Server Like Apache or IIS) , and Others.**

Background

- **IBM Made This All Extremely Secure, But Only If You Implement It Properly.**
- **Today If You're a Black Hat, We'll Show You How to Break Into It If the Security Tools IBM Gives Us Aren't Implemented Properly.**
- **If You Are A White Hat, We'll Show You the Tools to Use. (Hurry, Hurry!)**

Unlike Break-In Presentations for Other UNIXes:

- This One Won't Make It Easy for You to Break In
- We Haven't Yet Found Security Flaws in USS, So We Show You the Security Architecture and Likely Attack Points
- The Best Bet May Be to Find Basic Steps Not Taken or to Attack Through the Daemons

The Biggest Sources of Problems:

- **The MVS Sysprog Had to Teach Himself UNIX, TCP/IP and Much More Overnight.**
- **He Had a Tight Deadline, and He Gets Rewarded for Meeting Deadlines and Keeping Production Going.**
- **The Security Officer Didn't Know It Was Happening, and Doesn't Know UNIX, etc.**

This Is a Work In Progress

- **We'll Show You How the Security Works for: USS, TCP/IP, Its Daemons**
- **We'll Show You the Attack Methods We've Discovered and the Defenses Against Them**
- **Examples are Based on RACF, but ACF2 and TopSecret Provide the Same Support. References to SAF refer to which ever of these three security software products is installed.**
- **We Invite You to Add to the List**

When MVS Starts Up:

- **He Starts VTAM, JES, RACF, etc. and USS**
- **USS Reads His Control File to Get the Name of the TCP/IP Started Task, and Starts It**
- **TCP/IP Reads His Control File to Learn the Names of the Started Tasks for His Daemons, Such as FTP and Websphere**

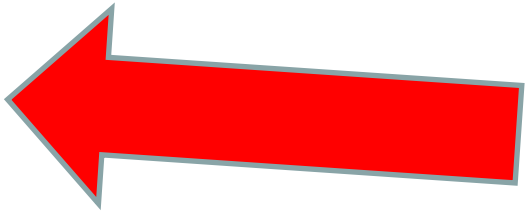
Each of the Daemons Can:

- **Read a Control File to Learn His Options**
- **For FTP (File Transfer Protocol), the Control File Might Be FTP.DATA**
- **For Websphere, the Control File Might Be /etc/httpd.conf**

To Gather Info for an Attack

- **Learn the IP Addresses of the Computer (Like Phone Numbers, Used for Routing) (Do This by Issuing a whois Request for www.mytarget.com)**
- **Learn the Ports that Are Open (One per Daemon or Application) by PINGing**
- **Browse the Control Files by FTPing**
- **But Let's Start By Attacking USS First**

Three Break-In Paths

1) **USS** 

2) **TCP/IP**

3) **TCP/IP Daemons**

1) USS Break-Ins

- How USS Security Works
- Break-In Approaches
- Protections

UNIX Security Uses Two Control Blocks:

- **The USP (User Security Packet) Has All the Info About a User's Identity. It is Similar to an ACEE.**

- **The FSP (File Security Packet) Has All the Info About Who Can Read, Write, and Execute One File. It is Similar to a Dataset Rule.**

When IBM Put USS on the Mainframe, Under the Control of MVS, They:

- **Replaced the /etc/passwd File with a Call to RACF or Other Security Software to Build the USP**
- **Kept the UNIX File System and Its Security Intact (Including FSPs)**
- **Added Callable Services, Routines in MVS and RACF That UNIX Programs Can Call**
- **Added Calls to RACF in Resource Classes Such as UNIXPRIV and FACILITY and APPL to Finetune Security**

Other Versions of UNIX:

- **Rely on a file called /etc/passwd (and also /etc/shadow) to establish a user's identity. This is a common source of UNIX security weakness.**
- **By replacing these files with a call to RACF, IBM has improved UNIX security on the mainframe.**
- **Note that ACF2 and TopSecret Perform the Same Functions as RACF for USS**

UNIX (Including OMVS or USS) Identifies Each User with a Numeric UID. (Not a Userid, But a UID)

- As in RACF, users are grouped into groups. Each group is identified, not by a group name, but by a numeric GID.

- RACF associates RACF userids with OMVS UIDs by keeping the UID in the OMVS segment of the RACF User Record. Guess where RACF keeps the GIDs!

IN GENERAL, USS DOESN'T LET YOU DO ANYTHING UNLESS

- YOU CAN COME UP WITH A RACF USERID WITH A VALID UID, CONNECTED TO A GROUP WITH A VALID GID.

- CONTROLLING THE GRANTING OF UIDs AND GIDs IS:

THE WAY TO CONTROL ACCESS TO USS.

The Superuser Privilege

- **Superuser Lets You Issue Any Command and Do Anything to Any File. You Get superuser in Three Ways:**
  - **A UID of 0 gives you superuser. (A GID of 0 does not!) (NOT RECOMMENDED!!)**
  - **Read access to FACILITY Class rule BPX.SUPERUSER allows you to switch to superuser status whenever you want.**
  - **A TRUSTED or PRIVILEGED Started Task**

RACF USER AND GROUP RECORDS

User Record	Group Record	Group Record	Group Record
USER29	GRPQ	GRPD	GRPX
GRPQ, GRPD, GRPX			
UID(17)	GID (532)	GID (612)	GID (27)

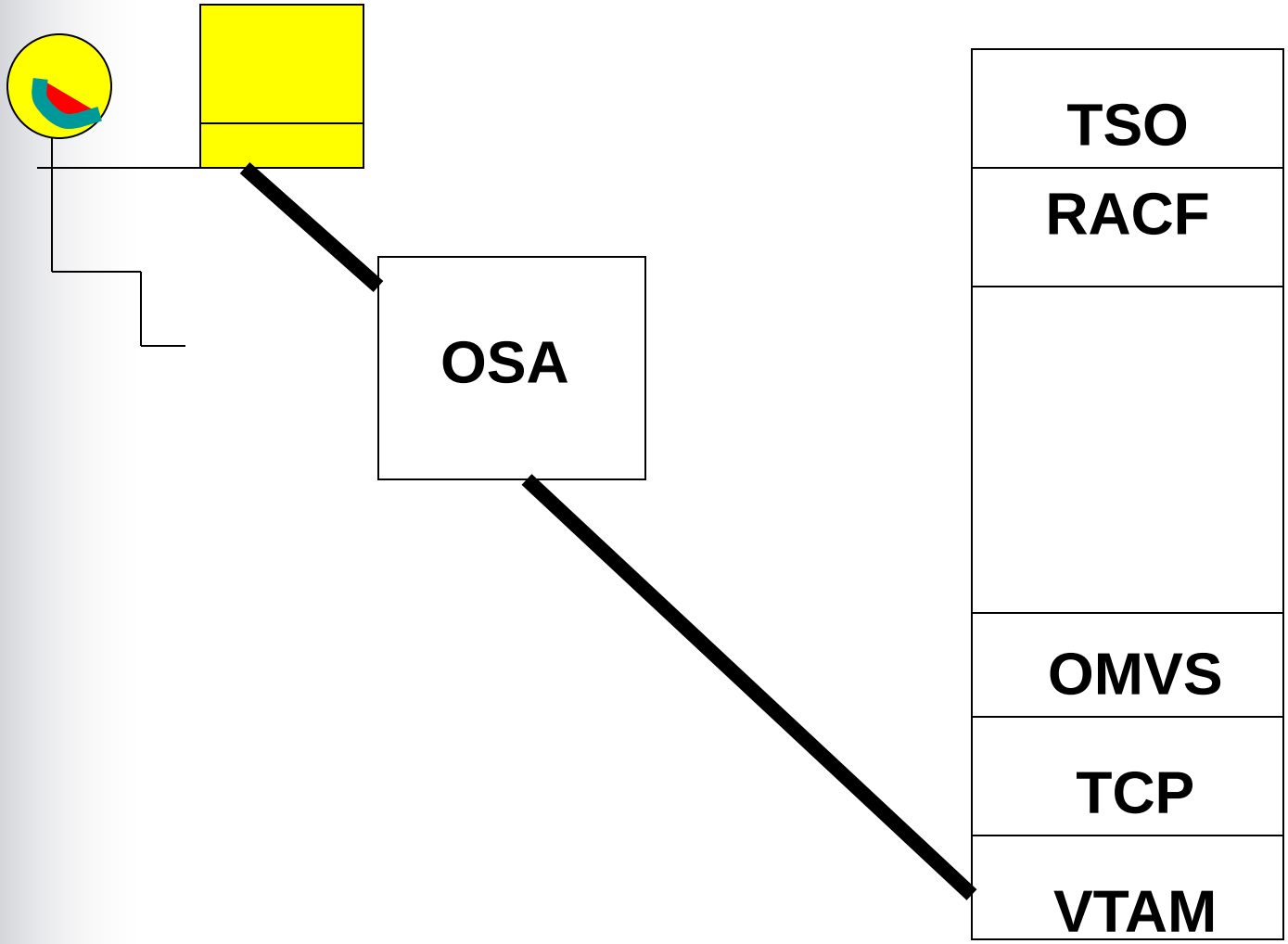
THE USP

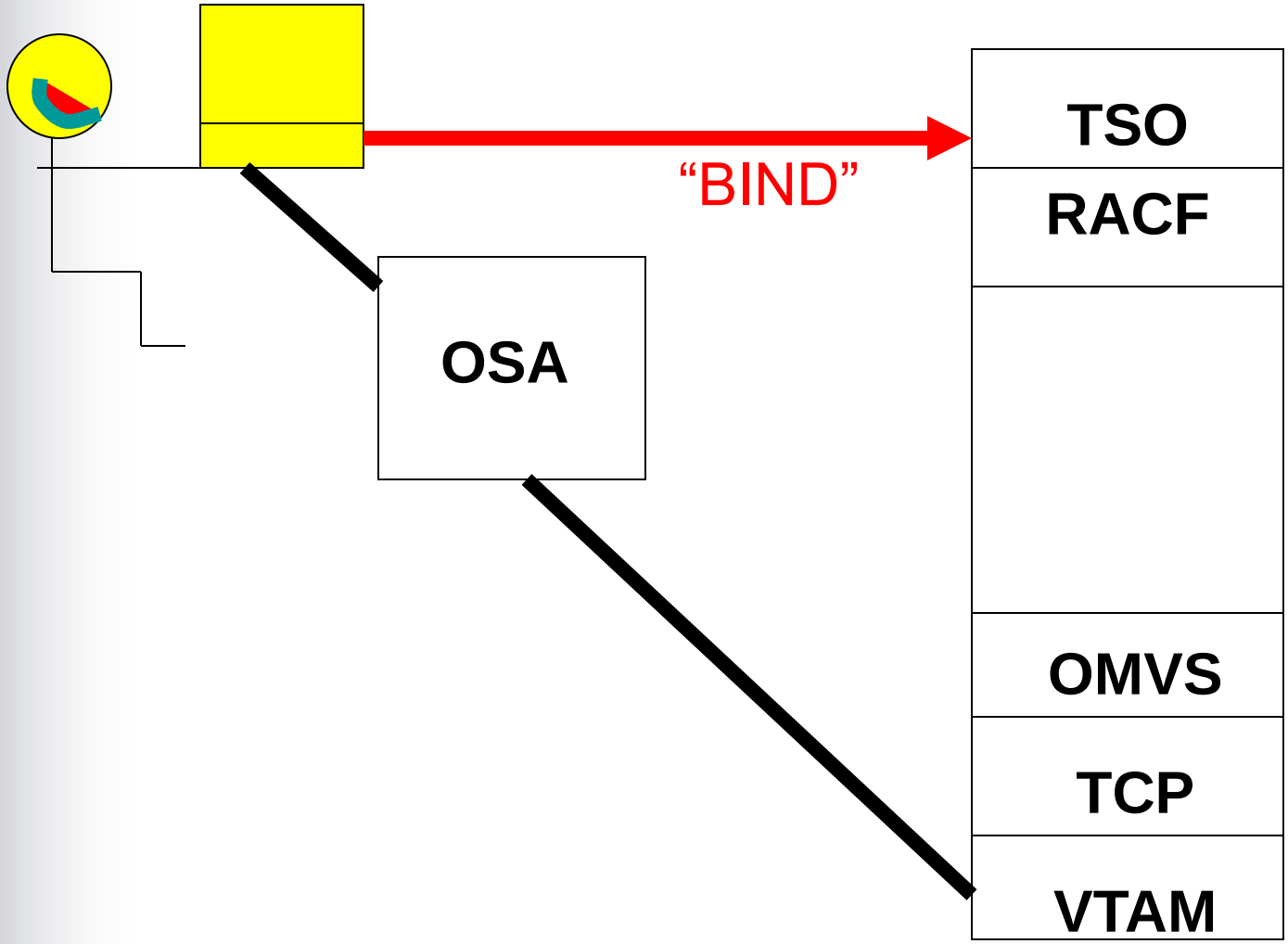
UID	GIDs	Other Stuff
17	532, 612, 27	

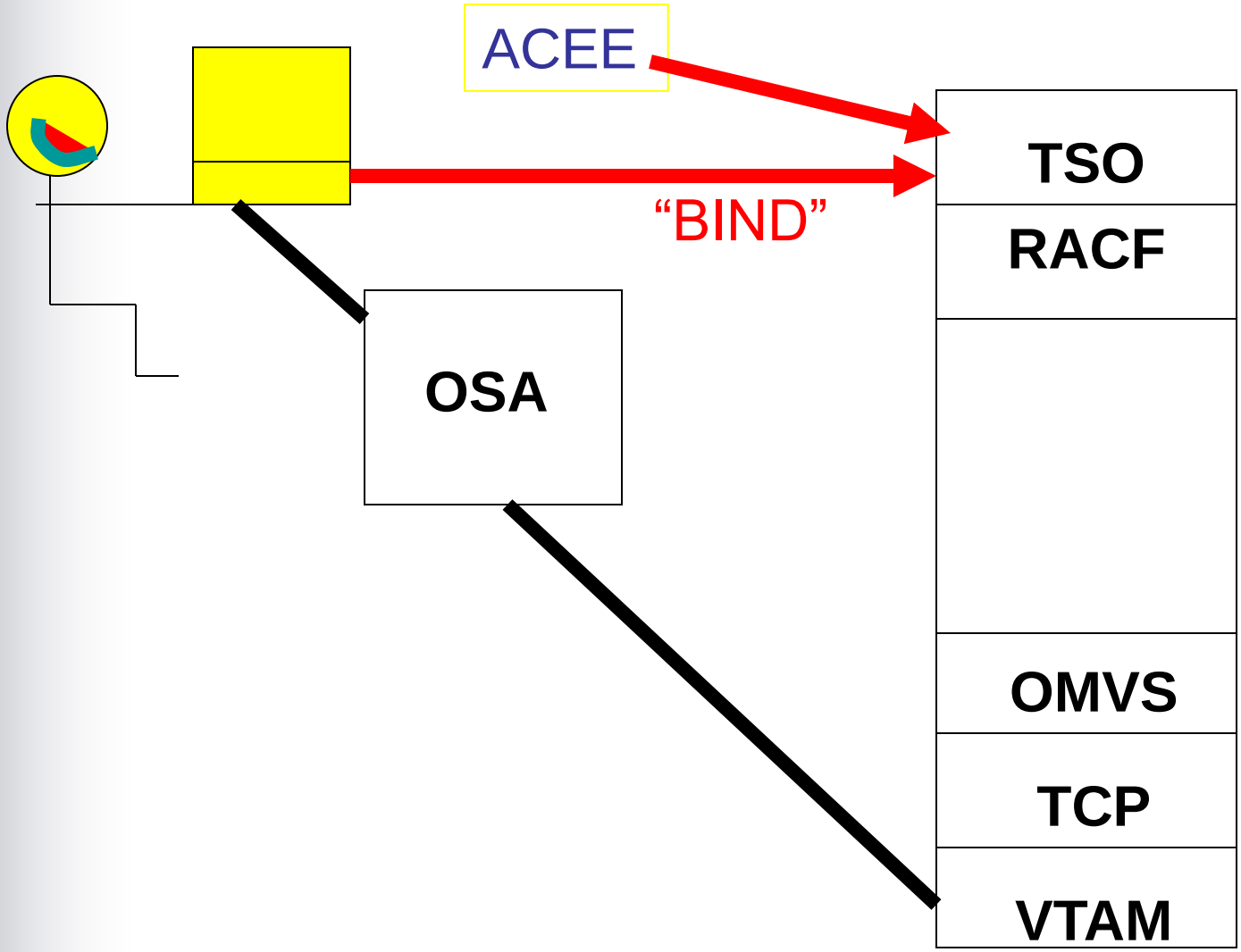
How the USS File System Works on MVS

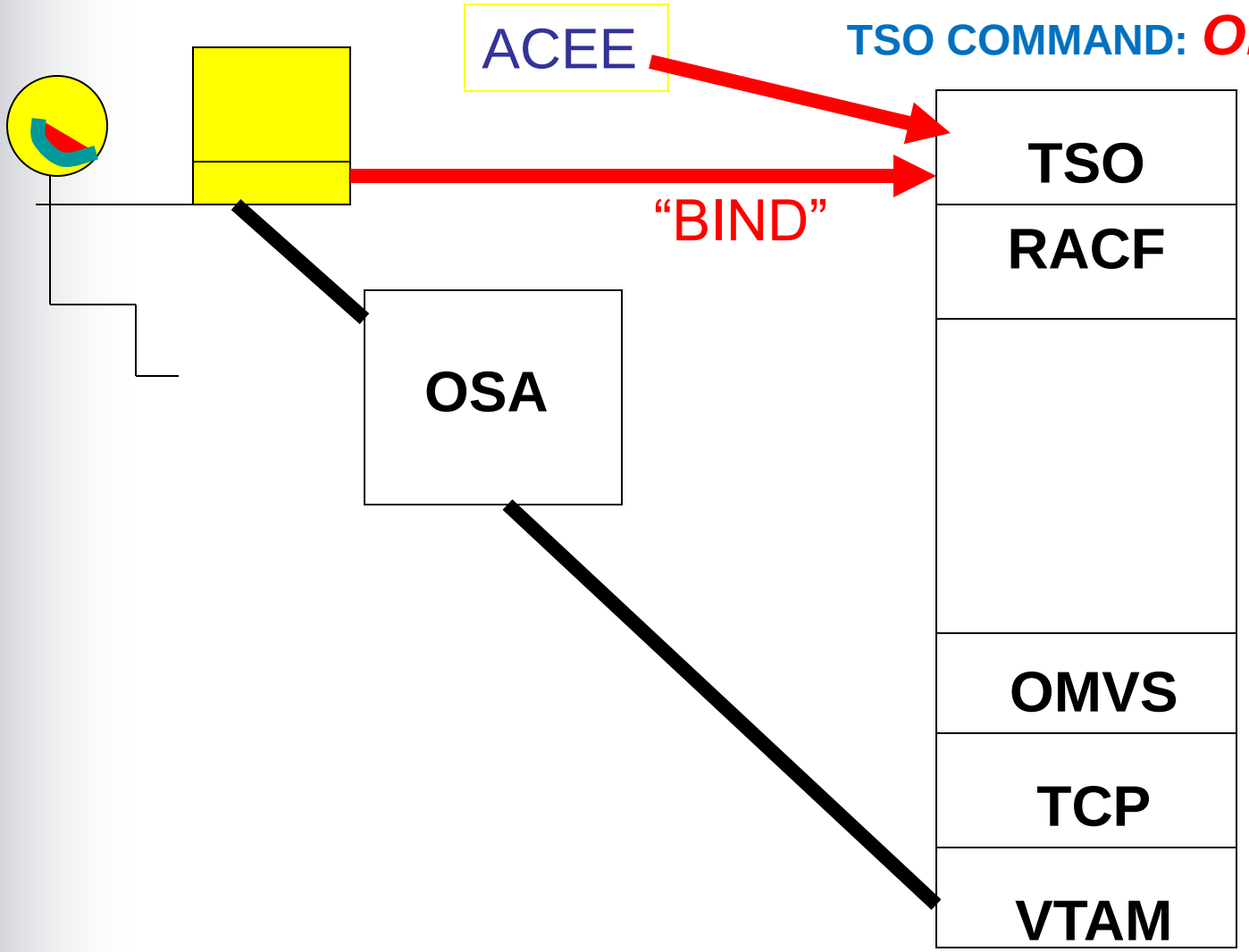
- **USS uses a large, VSAM file called the HFS or Hierarchical File System**

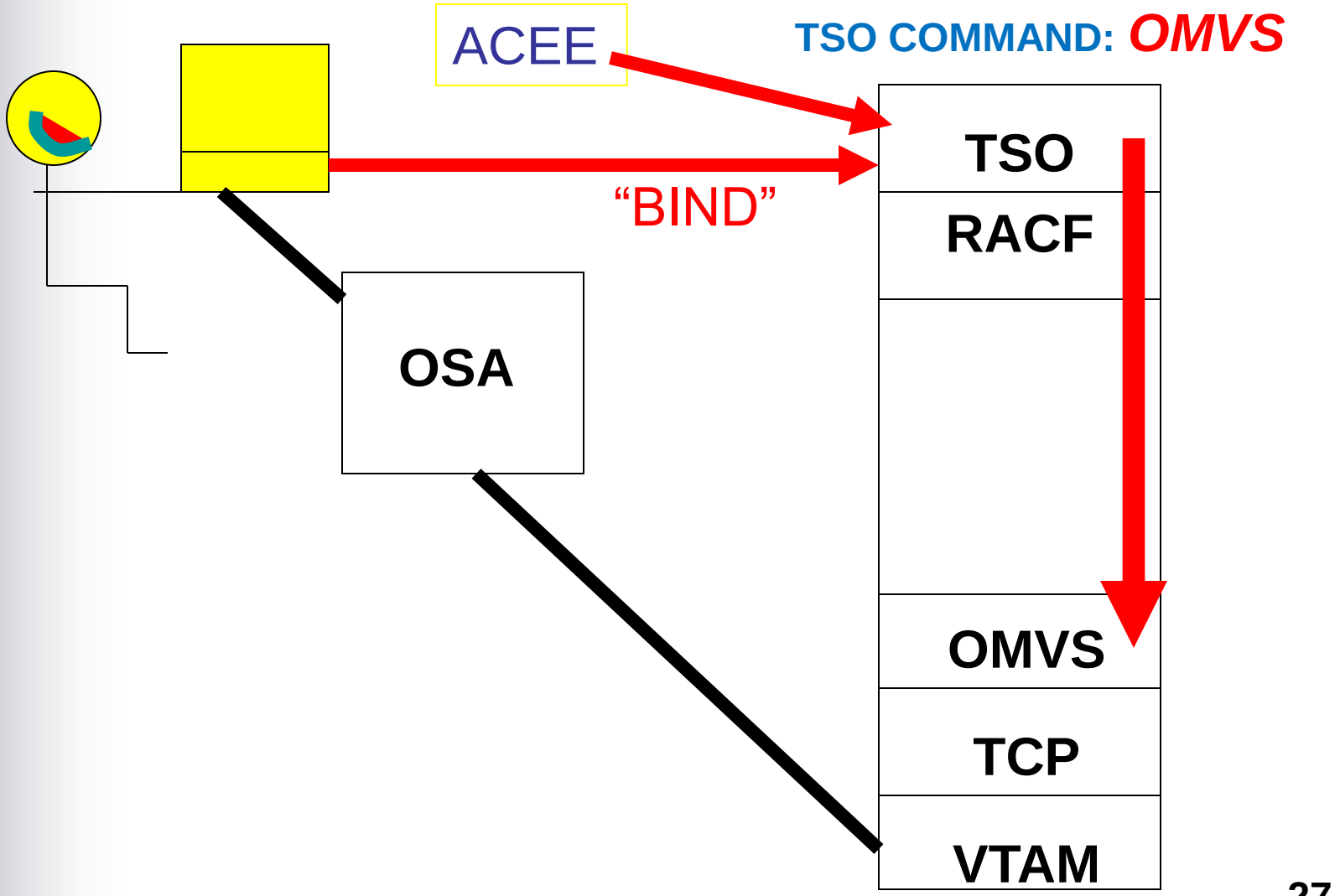
- **From MVS' point of view, it is a PDS/E file, which needs to be RACF-protected like any other disk file.**

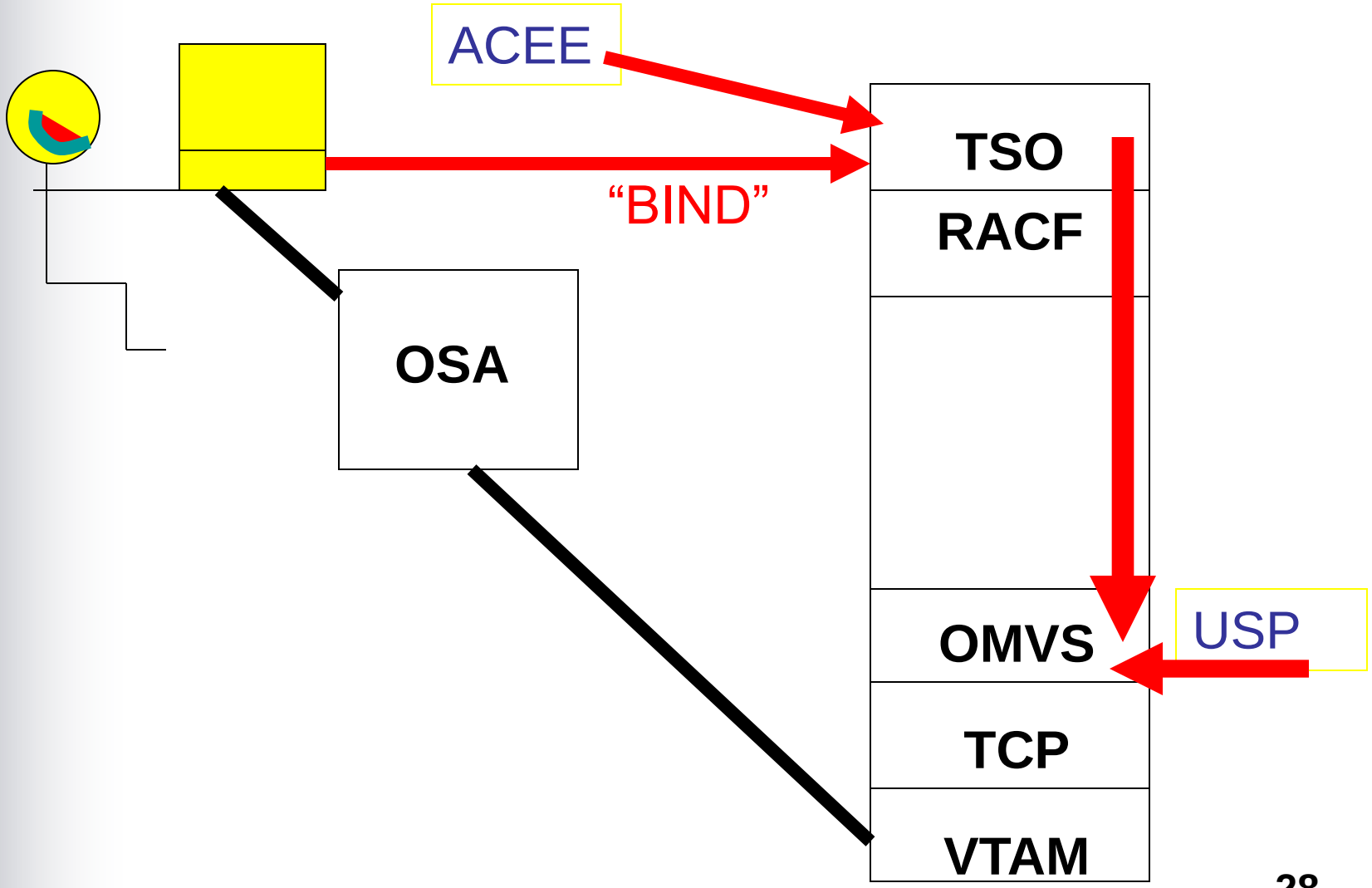


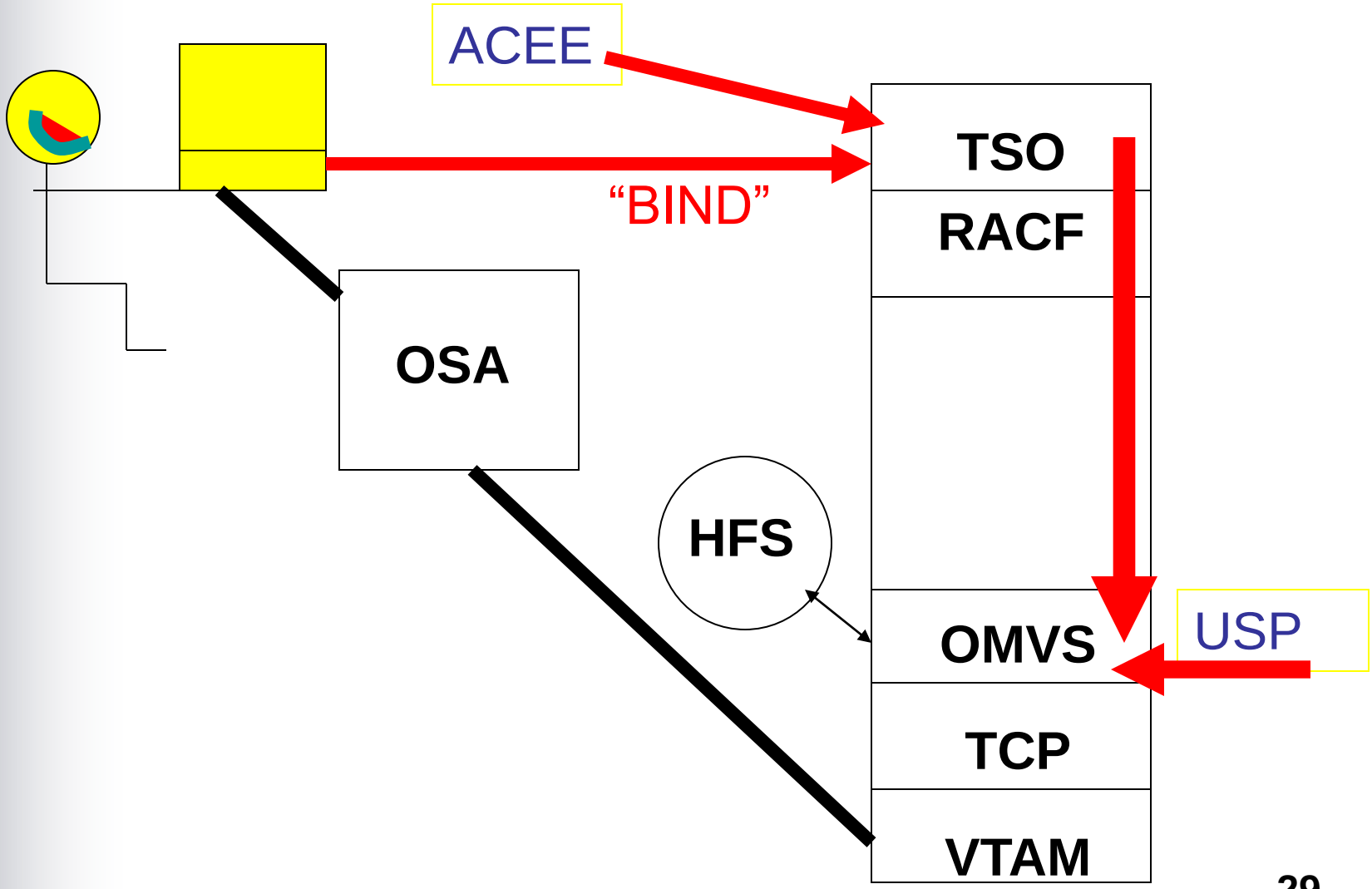


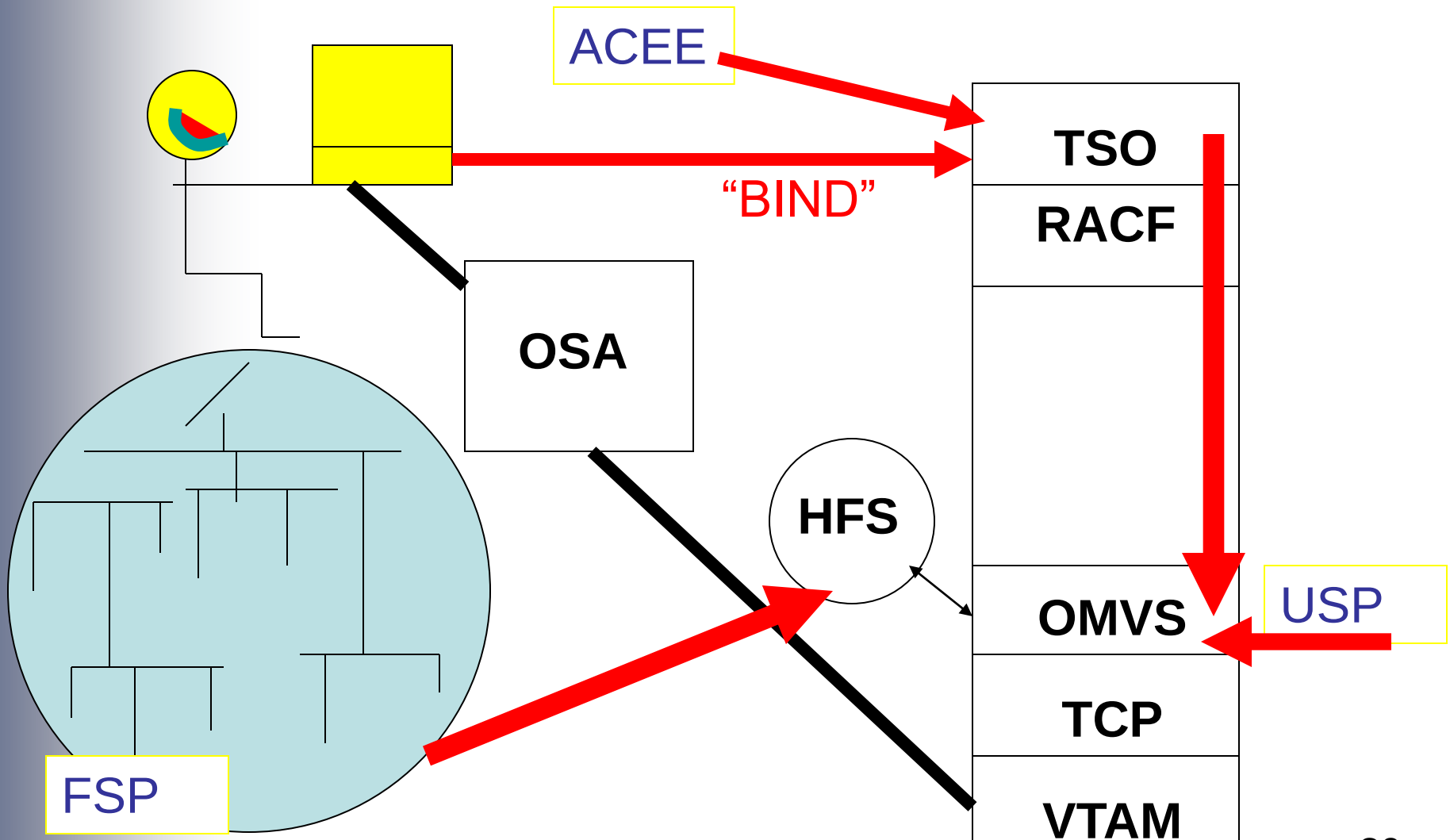








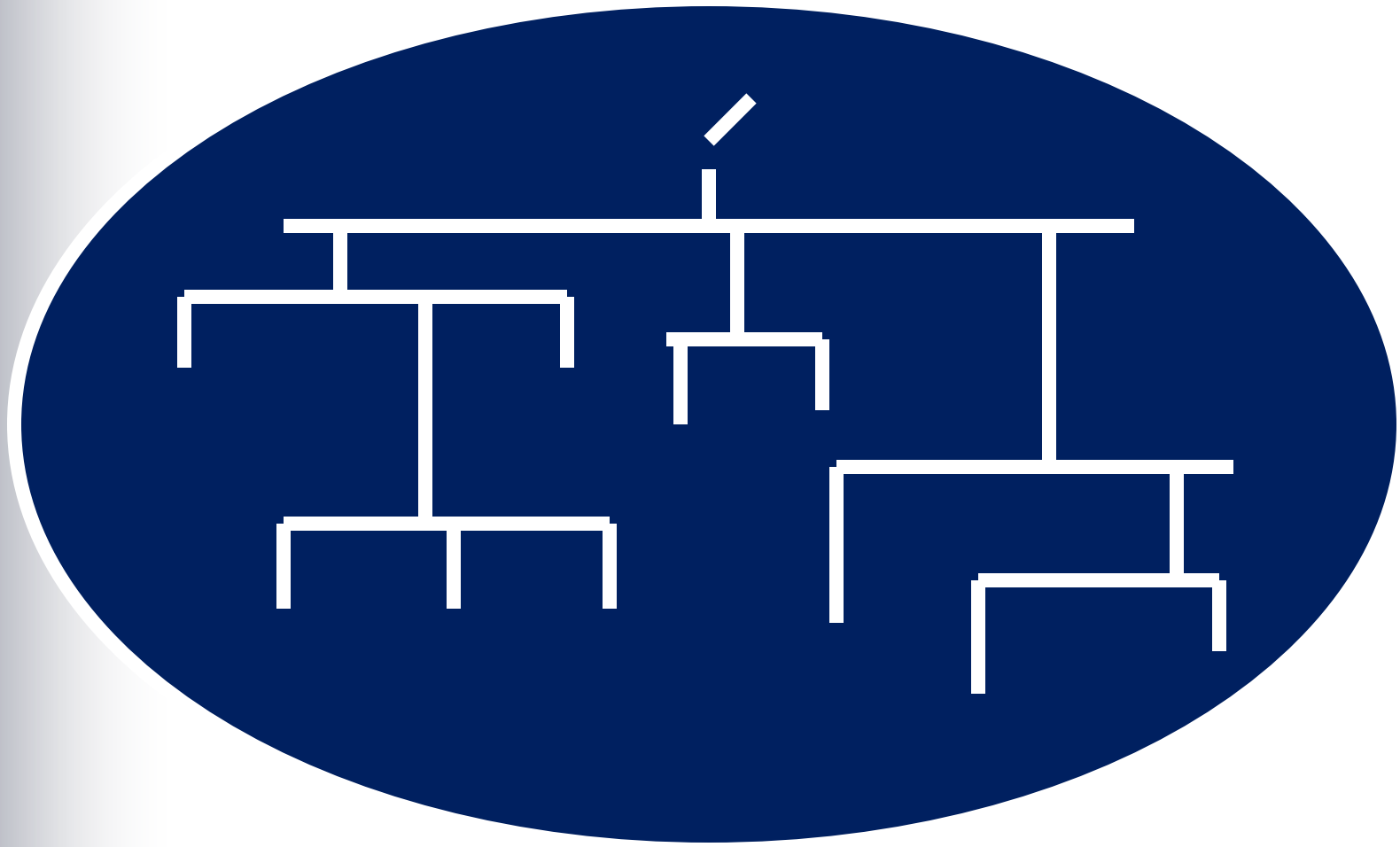




USS builds his file system within the HFS by

- **Reading and writing UNIX files in it, all organized into a directory structure.**
- **This directory structure is similar to the directory structure on your personal computer with MS/DOS or Windows. Each directory is actually a special type of file.**

HFS (Hierarchical File System)



When Your System Programmer Installs USS:

- She allocates a large, PDS/E MVS dataset, called the HFS (or Hierarchial File System) dataset. USS uses the HFS dataset to contain all the UNIX files.
- UNIX files in the HFS are organized into directories. The directories are organized into a tree, with parent and child directories, just like on your personal computer.
- Each file and directory has an FSP (File Security Packet) which controls who can access it.

Then How Does This FSP Work To Control File Access?

- **The FSP contains a string of information which is compared to the user's UID and the GIDs of all the groups she is connected to. The FSP includes the UID of the owner of the file, the GID of the group of the owner of the file, and 3 sets of 3 bits each. (There's other info in the FSP, but let's digest just a bit at a time.)**

The 3 sets of bits are for:

- the owning UID,
- the owning GID, and
- the rest of the world, respectively.

- Each set of 3 bits includes one bit for read access, a second bit for write access, and a third bit for execute access. These bits are often referred to as R, W, and X, respectively.

Owner		Owner 3 Bits			Group 3 Bits			Rest of World 3 Bits			Other Stuff
UID	GID	R	W	X	R	W	X	R	W	X	...
16	27	1	1	1	1	0	0	0	0	0	...

- **(If the file were a directory instead of a regular file, the bits would have slightly different meanings.) Each of the three sets of three bits can be expressed as an octal number, in our example: 740.**

THE USP

UID	GIDs	Other Stuff
17	532, 612, 27	

THE FSP

Owner		Owner 3 Bits			Group 3 Bits			Rest of World 3 Bits			Other Stuff
UID	GID	R	W	X	R	W	X	R	W	X	...
16	27	1	1	1	1	0	0	0	0	0	...

THE USP

UID	GIDs	Other Stuff
17	532, 612, 27	

THE FSP

Owner		Owner 3 Bits			Group 3 Bits			Rest of World 3 Bits			Other Stuff
UID	GID	R	W	X	R	W	X	R	W	X	...
16	27	1	1	1	1	0	0	0	0	0	...

THE USP

UID	GIDs	Other Stuff
17	532, 612, 27	

THE FSP

Owner		Owner 3 Bits			Group 3 Bits			Rest of World 3 Bits			Other Stuff
UID	GID	R	W	X	R	W	X	R	W	X	...
16	27	1	1	1	1	0	0	0	0	0	...

To Display the FSP,

- **You use the UNIX command ls -l. The left-most part of each output line from the ls command consists of 10 characters representing these permission bits. The leftmost character is d if the file is actually a directory, otherwise the leftmost character is a -. The remaining nine characters represent the three sets of three bits, each displayed as one of these characters: r, w, x, or -.**

For our Example,

- **The ten characters would be:**

-RWXR-----

- **which could be viewed as:**

- RWX R-- ---

- **You Should Assume that OMVS (that is USS) Users Can Get to the Rest of MVS, and That MVS Users Can Get to USS If You Don't Control Them.**

If You Come Into USS, Not Through TSO, But From the Outside,

- Say through TCP/IP, then USS lets you do nothing until you prove who you are to RACF's satisfaction.

- There are two exceptions where you don't have to supply a valid USS UID and GID from the security software, but these are both controlled by SAF (One is a FACILITY class rule to specify a default userid, and the other is an anonymous userid used with FTP under TCP/IP. The anonymous userid however MUST be defined to RACF with a valid UID and GID, or it won't work.)

If You Come Into USS, Not Through TSO, But From the Outside,

- Please note that IBM is moving us away from the default userid to a unique userid, controlled by a FACILITY class rule named **BPX.UNIQUE.USER**
- If this rule exists, then RACF assigns a unique userid and gid as needed to RACF userids accessing USS.
- The APPLDATA field of this rule can specify a model OMVS segment.

In Addition to USPs and FSPs, USS Uses:

- **ACLs (Access Control Lists)**
- **Callable Services in MVS and RACF**
- **the OMVSAPPL APPL Rule in RACF**
- **RACF Rules in the FACILITY Class to Control What Users Can Do**
- **RACF Rules in the UNIXPRIV Class to Control What Users Can Do (For Example, Subsets of SUPERUSER)**

USS Break-In Approaches

- **Assume Someone's Identity**
  - **setuid**
  - **Callable Services**
  - **Default Userid/Group**
  - **the su Command**
  - **Slipping In Commands**
- **File and Resource Attack Points**
  - **Mountpoints**
  - **Directory Tree Fencing Not Used / Weak Control Over Files**
  - **Programs**

setuid

- **Each Program is a File. Its FSP Has Two Bit Flags That Can Be Set:**
  - **setuid says that while this program runs, the user assumes the UID that owns the file (from the FSP)**
  - **setgid says the same thing for the owning GID**
- **Can I Somehow Insert a setuid Program Owned by 0 (SuperUser)?**

The setuid and setgid Bits Are Turned Off Whenever:

- A New File is Created
- The Owning UID or GID is Changed
- The File is Modified

Callable Services

- IBM Added Several Callable Services to MVS and RACF so USS Can Talk to Them
- One of These Callable Services is a Request to Assume the USS Identity (that is the USP) of Some Other User
  - seteuid()
- Another is a Request to Assume the RACF Identity of Some Other User
  - pthread_security_np()

These Callable Services Are Controlled by:

- **RACF FACILITY Class Rules Named BPX.DAEMON and BPX.SERVER Respectively**

- **If the Rules Exist, then You Need READ Permission to Them to Use the Callable Service. If They Don't Exist, Then Any Superuser Can Use these Callable Services**

If These Rules Exist, Then

- **Extra Program Control is Required with RACF, and Sometimes Additional Checking in the RACF SURROGAT Resource Class**

- **If They Don't Exist, Then Any Superuser Can Become Any RACF User Whose Userid Has an OMVS Segment. Even the RACF Admin!**

Default Userid/Group

- **USS Has Three Standard Ways to Define Anonymous Users:**
  - **FTP Anonymous User (Described Below)**
  - **The RACF Anonymous User Defined with a FACILITY Class Rule Named BPX.DEFAULT.USER whose APPLDATA Field Contains the Default RACF Userid and Group for Default UID and GID**
  - **BPX.UNIQUE.USER (described above)**

The su Command

- **The Command su (switch user) is a Request to Assume Some Other User's Identity. You Must Provide That User's Password.**

- **The Exception is su with No Operands, a Request for SuperUser Privilege. This Requires READ Permission to the RACF FACILITY Class Rule Named BPX.SUPERUSER.**

Slipping In Commands

- **UNIX has several files of commands that get executed whenever the system is started up, whenever someone logs on, or whenever other events occur**

- **If I can update one of these files, then I just have to wait for the event to occur to have my commands executed under someone else's authority.**

Examples of These Command Files at USS Startup Include:

- **STARTUP_EXEC is a REXX Exec That Executes at USS Startup**
- **If STARTUP_EXEC is NOT Specified, then /etc/init OR /etc/sbin/init is Executed**
- **/etc/init Often Specifies that /etc/rc is to be Executed**

Examples of Command Files That Execute at Logon Include:

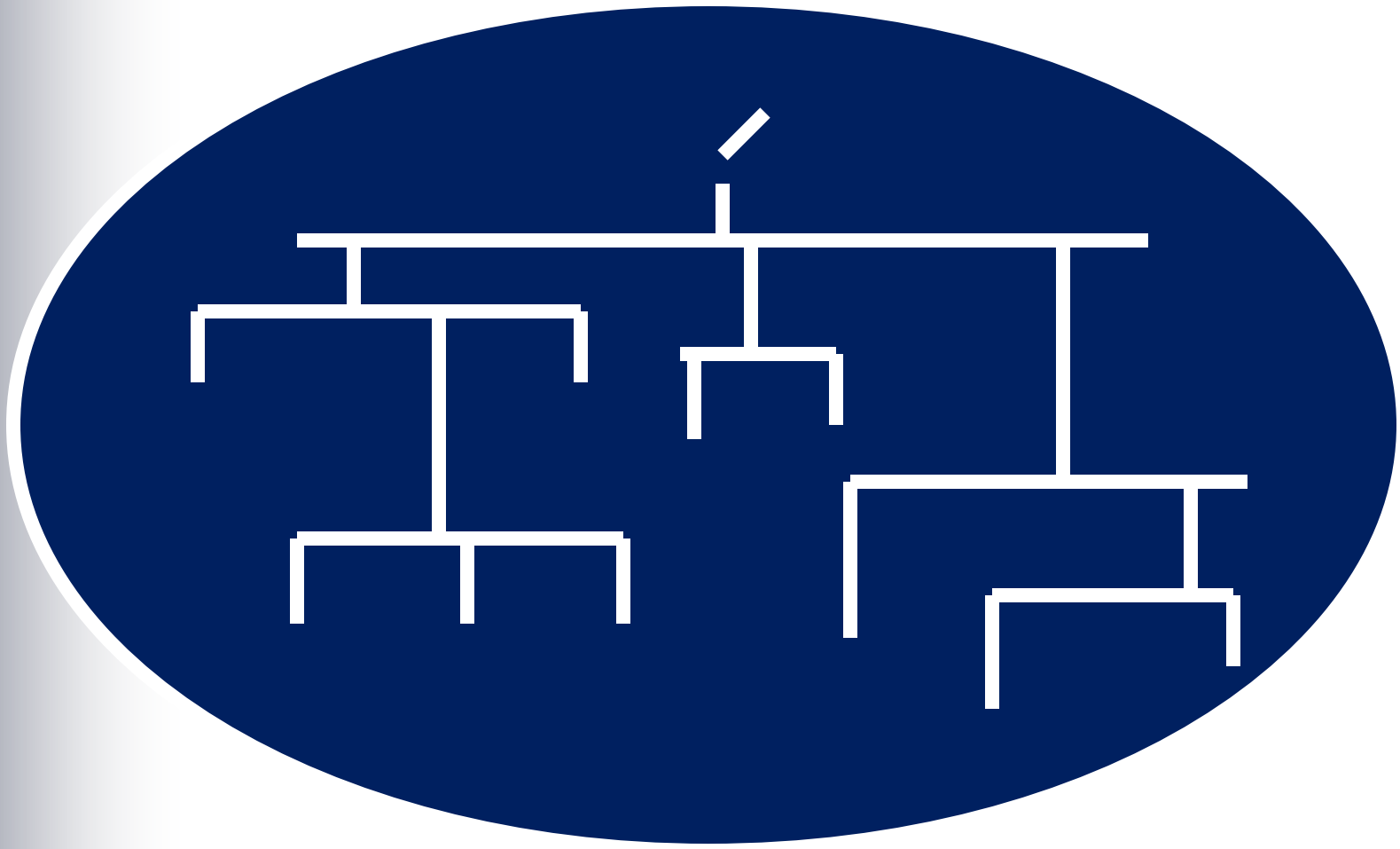
- **/etc/profile**
- **$HOME/.profile ($HOME is a variable equal to your home directory from your RACF User Record, OMVS Segment)**
- **Whatever File is Named in the ENV Variable (Type the env command to learn it)**

Mountpoints

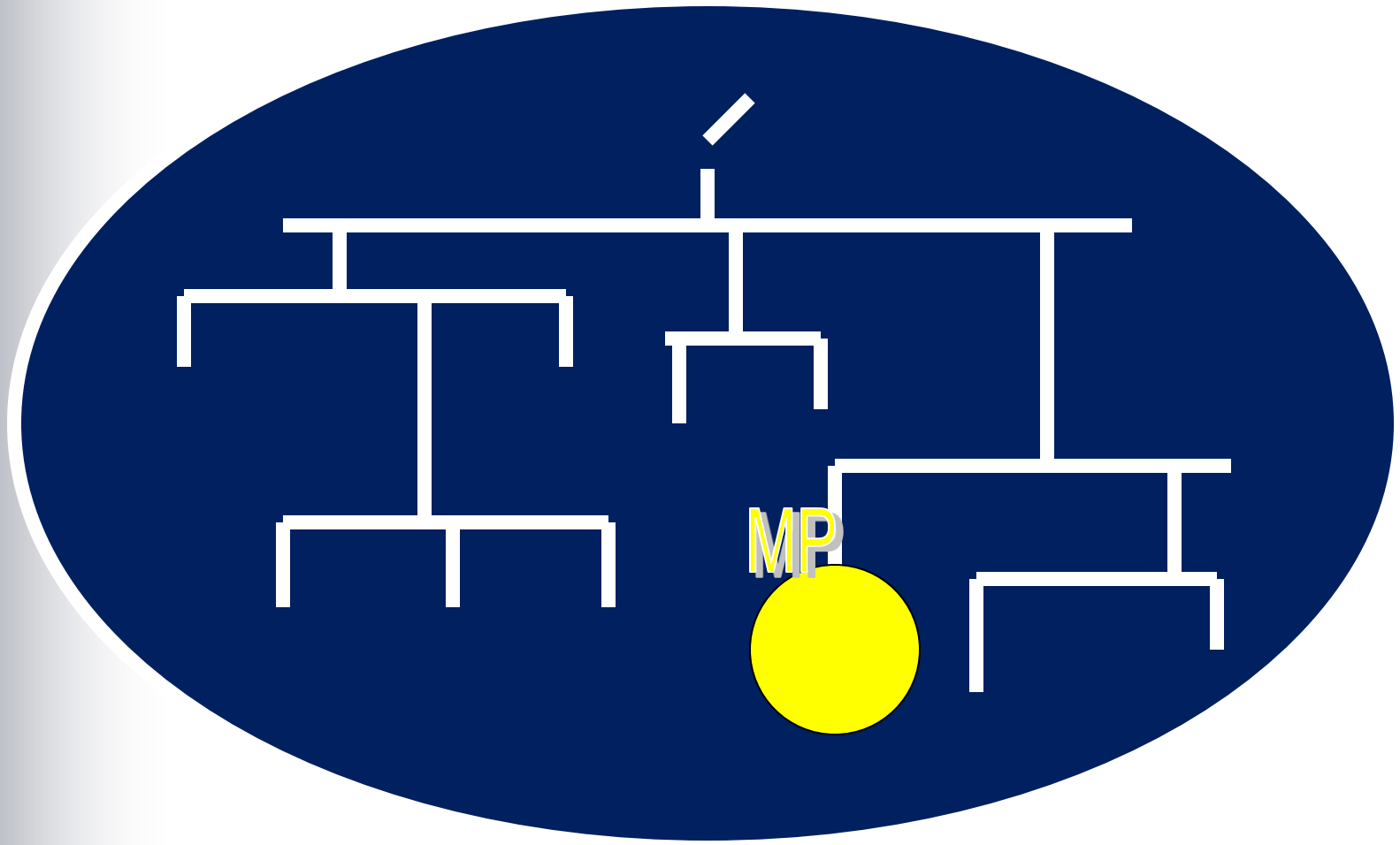
- **Suppose You Wanted to Take the Top of One Directory Tree and Plug It Into Some Part of Another Directory Tree.**

- **The Place It Gets Plugged In Is Called a Mount Point.**

HFS (Hierarchical File System)



HFS (Hierarchical File System)



Security Options at MountPoints Can Specify That All Files Below the MP:

- Are READ-ONLY OR
- Have No Security OR
- Have Standard File Security (FSPs)

AND

- Have the SETUID and SETGID Functions Disabled or Not

Directory Tree Fencing Not Used

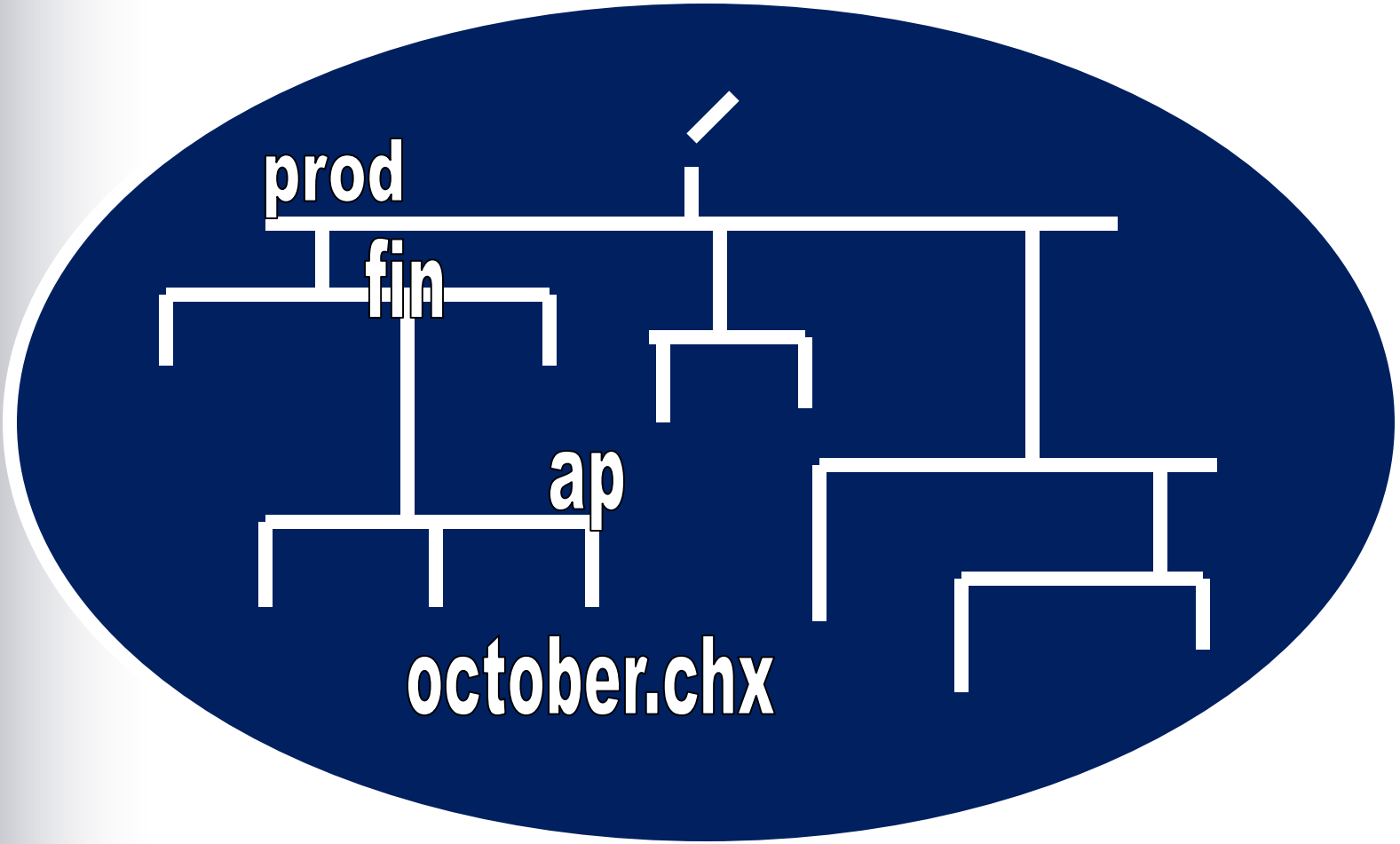
- **As With Other UNIXes, You Cannot Access a File Unless You Have EXECUTE Permission to Every Directory in Its Path**

- **Example:**

  /prod/fin/ap/october.chx

- **By Default, a New File Inherits Its Owning GID From Its Parent Directory**

HFS (Hierarchical File System)



Weak Control Over Files

- **Without a Formal Method for Controlling File Access, Inappropriate Rules are Sure to Creep In**

- **On a System of Any Size, This Will Usually Require Use of Directory Permissions to Fence Off Parts of the Directory Tree and/or Use of ACLs (Access Control Lists)**

Program Control

- When an MVS Program Executes, the Address Space is Tightly Controlled by Contents Supervision
- Programs with Privileges Lose Those Privileges When a non-Trusted Program Enters the Address Space
- UNIX Doesn’t Have That Control

Two Approaches

- **Use the Sticky Bit (a bit in the FSP) to Indicate That a USS Program's Execution is to be Replaced by Execution of a Corresponding MVS Program (Requires PROGRAM Protection in RACF) OR**

- **Use Extended Attribute Bits in the FSP or Sanction Lists to Indicate that a Program is Protected or APF-Authorized**

Examples of USS Attacks

- **SETUID and Mountpoints**
- **Callable Service**
- **Directory Tree**

Attack Method Example 1A

- **SETUID**
  - **Make a Program Owned by Root (UID Zero) and Get Its SETUID Bit Set; Then Execute It**
  - **This Can Be Difficult Since UNIX Forces the Owning UID for a File to Be The User Who Creates It**

Unless You Are SuperUser Or Have Permission from Some UNIXPRIV Rules:

- **You Can't Change the Owning UID of a File**
- **You Can Only Change the Owning GID if You Own the File AND You Are a Member of the Group**

Attack Method Example 1A (cont'd)

- **Mountpoints**
  - **Trick Someone Into Mounting Your File System Without Disabling SETUID Processing or Modify the Automount File (*/etc/auto.master*) to Add My File System with Open Security**
  - **Include in the File System a Program Owned by Root with the SETUID Bit Set**
  - **Execute the Program**

Attack Method Example 1B

- **Callable Service**
  - **Find a System That Doesn't Have BPX.DAEMON and BPX.SERVER Defined**
  - **Write a Program in C That Executes a Callable Service to Assume Someone's UNIX or RACF Identity**

Attack Method Example 1C

- **Directory Tree**
  - **Find a System That Gives Everyone EXECUTE Permission to Every Directory**
  - **When You Write a Program to Be Put Into Production, Include Statements in the Program to Permit You to Write to the Files The Program Creates**

PROTECTION AGAINST USS ATTACKS

- OMVSAPPL Rule in APPL Resource Class
- FACILITY Class and UNIXPRIV Rules
- Directory Tree Structure
- PROGRAM Class or Extended Attributes or Access Control Over Sanction List
- Commands df –v or D OMVS,F

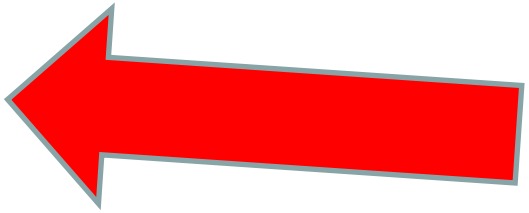
PROTECTION AGAINST USS ATTACKS

- **Maintain Inheritance Rules for Owning GID of Directories and Owning UID of Files**

- **Don’t Have OMVS Segments for Userids with RACF Privileges**

Three Break-In Paths

1) USS

2) TCP/IP 

3) TCP/IP Daemons

2) TCP/IP Break-Ins

- How TCP/IP Security Works
- Break-In Approaches
- Protections

How TCP/IP Security Works

- **TCP/IP Messages are Routed Over the Internet Using IP addresses (four numbers separated by dots, like 198.162.1.10) Each IP address corresponds roughly to one computer, and is like a phone number for routing.**

- **The TCP Part of the Message Contains a Port Number Which Corresponds to an Application on that Computer**

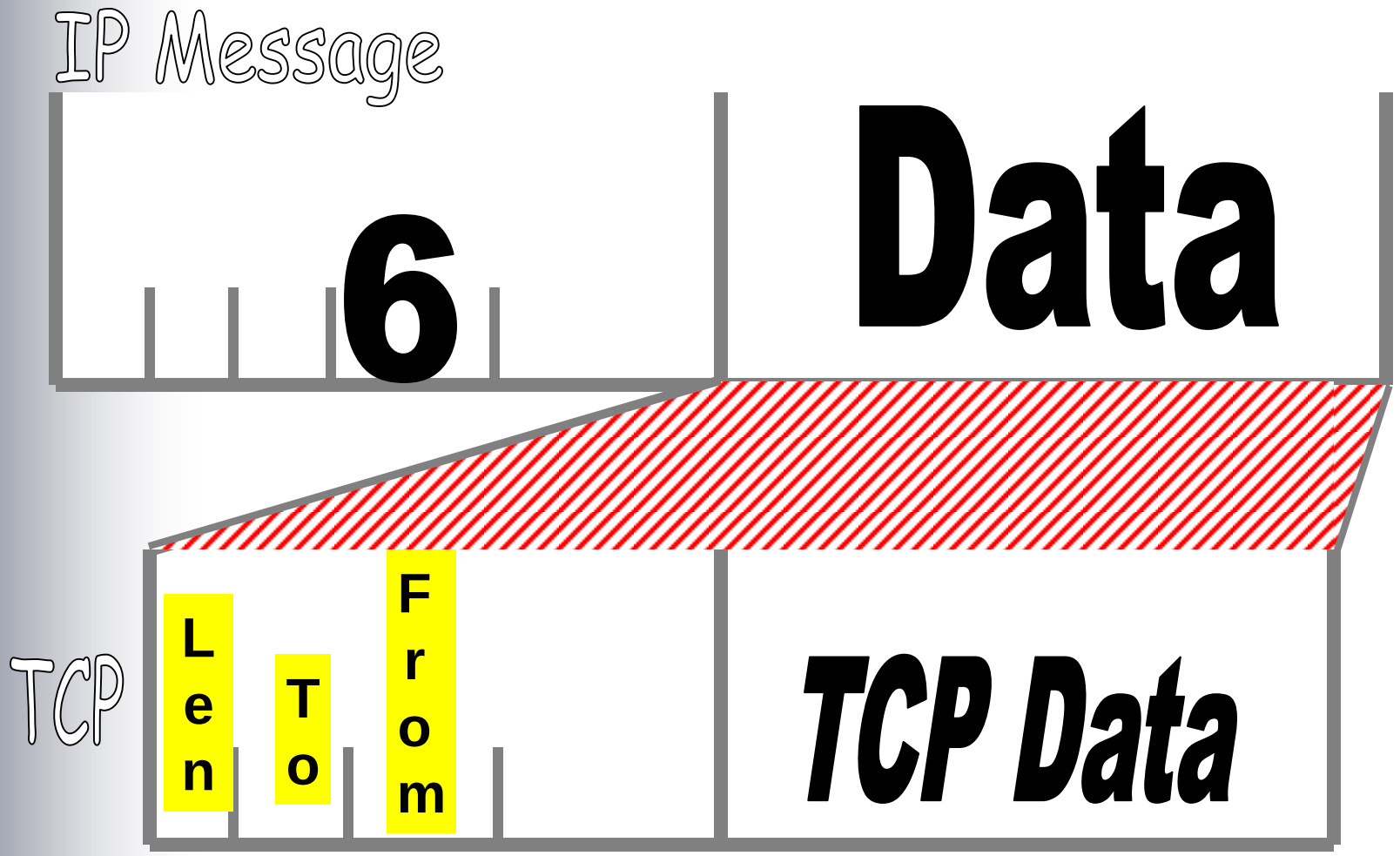
The Port Number Identifies the Application, Such as Email or File Transfer

- The TCP Software Uses the Port Number to Decide Which Server or Daemon to Hand the Message To. For Example, Port 25 is Usually Reserved for the Email Server. Daemons and Servers are Always Started Tasks.

- TCP Keeps a Control File, Often Named PROFILE.TCPIP, Which Lists the Ports and the Daemon for Each.

IP Message Has Length, To/From Addresses, Protocol, Data

Simplified TCP/IP Message Layout (Data Part of IP is TCP Packet)



TCP/IP Break-In Approaches

- Port Scanning
- Denial of Service
- Hijack Session
- View Passwords on Un-Encrypted Links
- Open a Port
- Trick Someone Into Executing a Program for You (Not Addressed in this Presentation)

Port Scanning

- Use whois to Learn the IP Address of the Target
- PING Every Port at That Address to See Who Responds
- For Software with Known Weaknesses, Abuse Them (Google: sendmail vulnerabilities or see CERT)
- Why This Doesn't Work so Well with z/OS

Denial of Service (DOS)

- **Overwhelm the Machine or Buffer Overflow**
- **(Why It Doesn't Work So Well with z/OS)**
  - **z/OS Machines Have More Power**
  - **IBM Cleaned Up the Code**
  - **MVS Doesn't Use a Stack to Save Registers**

Hijack a Session

- **Get a Job as Sysadmin at an Internet Site**
- **Modify the System to Intercept Someone's Conversation or Session with Someone Else**
- **The Flag to Make Every Message in the Conversation Come Through Your Computer**

View UnEncrypted Passwords

- You'd Think We'd Encrypt Automatically
  - Sniffers on LANs
  - System SSL

Open a Port

- **If You Work for a Company, and Want to Leave a Back Door, Write a Program that Opens a Port, and Waits for Someone to Send it a Message Telling it What to Do**

- **Install the Program to Execute Automatically Every Day**

PROTECTION AGAINST TCP/IP ATTACKS

- Protected Userids
- Firewalls
- Explicit Control of Ports and Applids with TCP/IP control file
- No modems allowed on PCs connected to a LAN (These bypass the firewall protection.)
- Encryption to ID user (SSL and Kerberos)

Additional Protection With TCP/IP

- System SSL/TLS (AT-TLS and native)

- Be aware that when implementing SSL/TLS there are different methods of operation. When operating in AT-TLS, application transparent TLS, you will need the policy agent (PAGENT), as you are letting the PAGENT control how SSL initiates. When operating natively, the SSL/TLS configuration is in the control file of that given application.

Additional Protection With TCP/IP

- **NETSTAT Command to Learn Open Ports**
- **Firewalls (Named After the Real Firewalls in a Building, Separating Areas into Cells to Stop the Spread of Fire)**
- **The SERVAUTH Resource Class (More info available in article at website: www.stuhenderson.com)**

Additional Protection With TCP/IP

- **Intrusion Detection Software**
  - **Built-In for Free, Part of Policy Agent**
  - **Detects Patterns of Incoming Messages Which Identify Likely Attacks**

Additional Protection With TCP/IP

- **IPSEC is suite of protocols available when IP filtering is enabled, it works on the transport layer and encrypts the packets individually, which means it is possible for an application to run its own security algorithms on top of an IPSEC connection – this is not possible with SSL/TLS**

Three Break-In Paths

1) USS

2) TCP/IP

3) TCP/IP Daemons

3) Daemon Break-Ins

- How Daemon Security Works
- Break-In Approaches
- Protections

Each Daemon Has Its Own Way of:

- **Q1 Answering "Who Is This User?"**
  - **Always Comes Down to RACF Userid (Often with a USS UID as well)**
- **Q2 Answering "Can He Do X?"**

These Answers Are Based on Each Daemon's Control File

Methods of Attack for Daemons

- FTP
- Telnet
- Websphere
- Other

FTP (File Transport Protocol)

- Anonymous Logon
- JES and SDSF
- DB2
- MVS Datasets As Well As USS Files

FTP - Anonymous Logon

- **The FTP Control File (Often Named FTP.DATA) Specifies Whether Anonymous Logon is Allowed, and If So How**

- **It Can Require a Userid and Password or Not, Restrict What Directories Can Be Accessed or Not, and Restrict What Userids Can Be Used or Not**

FTP - JES and SDSF

- **FTP Can Connect to JES. It Appears Like an SDSF Connection**

- **Can Submit Batch Jobs and Operator Commands, as Well as Access Printouts**

FTP - DB2

- **FTP Can Also Talk to DB2**

- **This Requires Coordination of DB2 Security with FTP**

FTP - MVS Datasets as Well as USS Files

- **Control Files Like VTAMLST and /etc/inetd.conf and FTP.DATA and PROFILE.TCPIP**

- **Often Have a Default Access That Lets Anyone Read Them**

Telnet (Remote Logon)

- **Bind to an Applid– TN3270 (an enhanced version of telnet) Lets You Bind to a VTAM Applid (Application Identifier), Depending Upon the TCP/IP Control File Settings**

- **Logon Without SSL -- If You Use Telnet to Logon to TSO Without Providing Automatic Encryption, Your Passwords Are at Risk**

Websphere

- **Holes in httpd.conf (Q1 and Q2)**
- **Program Execution**
- **WAS (and DB2, CICS, MQSeries, and JAVA)**
- **(Lose Identity of User, Who Understands All the Various Security Technology?)**

Holes In httpd.conf

- **This Control File Specifies for Websphere How Users Are Identified (Q1) and What They Can Do (Q2)**

- **It Starts by Assigning a Default Userid, and Then Pattern Matching Against the Name of the File You Clicked on**

http://mytarget.com/dir1/dir1a/thisfile.html

If You Can Browse the Control File,

- **You May Find Holes in the Logic Which Permit You to Access Files or Execute Programs Improperly**

WebSphere: Program Execution

- **If the Pattern Match is to the Name of A Program, Then The Program Can Be Executed**

- **If the Program is Written in JAVA or Perl or Some Other Language, Most RACF Administrators Will Not Be Able to Read It.**

WebSphere: WAS (WebSphere Application Server)

- If the Patterns Match, Websphere Can Hand the Request to WAS, Who May Pass It to:
  - CICS Or
  - JAVA Or
  - MQSeries Or
  - DB2 (SQL Injection)

- Few People Know Enough to Evaluate the Security

SQL Injection is:

- **The Malicious Entering of SQL That Completely Changes the Intent and Effect of the Code.**
- **The Best Protections Against This May Be Peer Review of DB2 Code and SECLABELS in DB2**

Example of SQL Injection:

- **The Program Asks the User to Enter a Customer Number with Code Like This:**

  SELECT * FROM custable WHERE custno = '$INPUT[typedno]';

- **Instead of Just a Customer Number, User Enters This Line (Note Quotes):**

  123' OR 'XYZ' = 'XYZ

- **Resulting in:**

  SELECT * FROM custable WHERE custno = '123' OR 'XYZ' = 'XYZ';

- **Which is True for Every Row**

Protection Against Daemon Attacks

- SERVAUTH Resource Class in RACF
- System SSL
- Control Files for Daemons
- Restricted Userids
- Protected Userids
- Change Control Over Programs, Control Files, JCL
- Firewalls (Policy Agent)
- FACILITY Class Rules in RACF

Protection Against TCP/IP Daemon Attacks

- **TERMINAL and APPL Protection with RACF for Dial-in Ports and for FTP IP addresses**
- **Make Every Applid Call RACF to Check Userid and Password or Digital Certificate**
- **Exits to Control FTP Processing**

Three Break-In Paths

1) USS

2) TCP/IP

3) TCP/IP Daemons

5) SUMMARY AND CALL TO ACTION

- **It's Tough to Secure a Platform When You Need to Know All of:**
  - **MVS and RACF/ACF2/TSS,**

    PLUS
  - **USS, TCP/IP, and All the Daemons.**

- **But You Can Also See That IBM Gives Us One of the Most Secure Architectures Available.**

SUMMARY AND CALL TO ACTION (cont'd)

- **You Can Also See Ways that You Can Have Holes in That Architecture.**
- **Expanding the Approach Outlined Here Will Show You Where You Need to Tighten Security, Starting with: MVS, RACF/ACF2/TopSecret, USS, TCP/IP, Formal Change Control, and then Each Daemon.**
- **Do You Know Who Can Update the USS and TCP/IP Control Files?**

For Further Information:

- See article on SERVAUTH and back issues of the RACF User News and Mainframe Audit News at www.stuhenderson.com
- Computer Associates Has Cookbooks on Their Website www.cai.com for Both ACF2 and TopSecret to Address All the Functions We've Illustrated Here with RACF
- Kevin Mitnick's "The Art of Deception"

For Further Information:

- IBM Manual “Security Server RACF Callable Services” SA22-7691
- “The Cuckoo’s Egg” by Cliff Stoll
- Stu at stu@stuhenderson.com

For Further Information:

- See articles and back issues of the RACF User News and Mainframe Audit News at www.stuhenderson.com
- IBM manual “z/OS Communications Server: IP Configuration Reference”, SC31-8776
- IBM manual “z/OS Communications Server: IP Configuration Guide”, SC31-8775
- Computer Associates Cookbooks for ACF2 and TopSecret

End of Presentation

Thanks for Your Kind Attention.