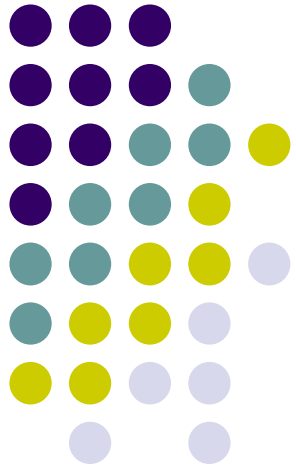
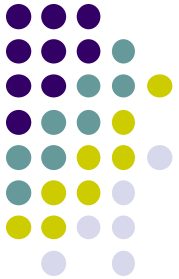
Operating Systems

Security

Professor Sina Meraji



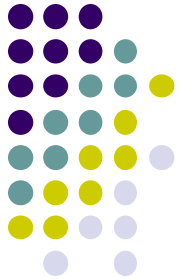
Security



- Types of security threats
- Protection basics
- Intrusion
- Malicious software

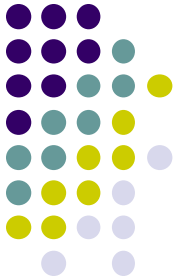
- Some principles of computer security

Security



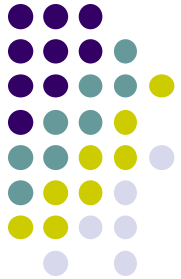
- Computer Security
  - Techniques for computing in the presence of adversaries
    - Four requirements of security
      - Confidentiality: preventing unauthorized release of info
      - Integrity: preventing unauthorized modification of info
      - Availability: ensuring access to legitimate users
      - Authenticity: verifying the identify of a user
    - Protection is about providing all on a single machine
      - Usually considered the responsibility of the OS
- Cryptography
  - Techniques for communicating in the presence of adversaries

Types of Threats



- Security requirements are intended to thwart four corresponding types of threats
- Interception, or eavesdropping - attacker gains knowledge they should not have access to
  - Loss of confidentiality
  - Reading or copying files that attacker should not have access to
  - Intercepting network packets
    - IP packets include destination field in header. Well-behaved network cards ignore packets that are not intended for them, but network cards can be placed in “promiscuous mode” where all packets are accepted

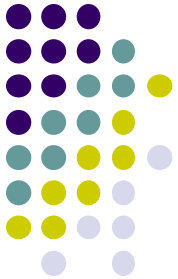
More threats…



- Modification - attacker alters existing files, programs, network packets, etc.
  - Attack on integrity
  - Destruction of data is a special case

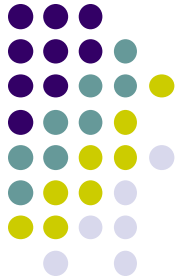
- Theft of service – attacker installs daemon
  - Attack on availability
- Fabrication - attacker creates counterfeit objects (files, messages, etc.) which appear to come from a trusted source
  - Attack on authenticity

Vulnerabilities in the System



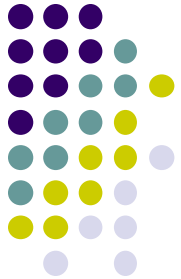
- Physical access
  - Unauthorized physical access makes it a lot easier to gain unauthorized digital access
- Humans
  - Who should you trust? How much?
- Operating system
  - Flaws in the system allows security protocols to be circumvented
- Networks most sceptible
  - Data travels over unsecured communication lines, across multiple administrative domains

Intrusion Detection



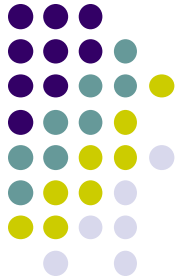
- Generally want to know when a system has been compromised
- Two main approaches
  - Signature-based detection – examine system activity or network traffic for patterns that match known attacks
  - Anomaly detection – identify patterns of "normal" behavior over time and detect deviations from those patterns
- Systems need records of user activity to help detect and recover from intrusions
  - These records have to be protected from tampering by the intruder

Malicious Software (Malware)



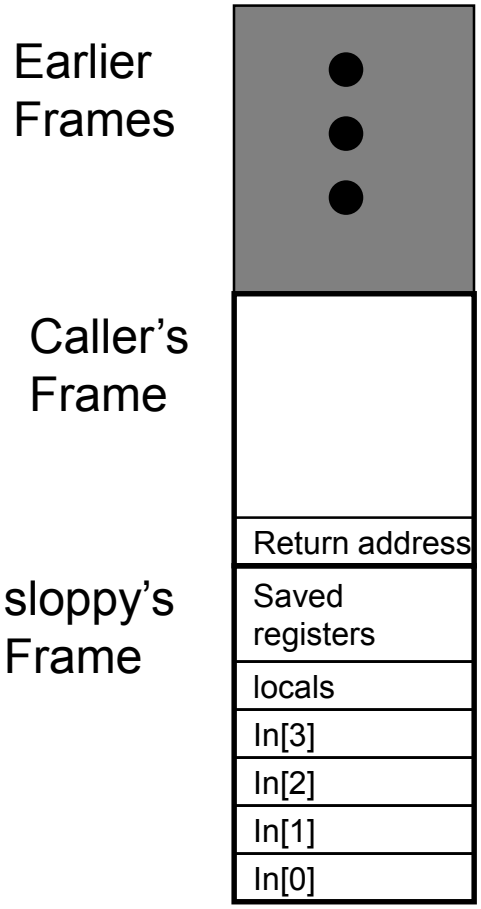
- Trap doors - program contains secret entry point that allows attacker to bypass security
- Logic bombs - destructive code in legitimate program triggered by some event unintentional
- Trojan horses - apparently useful program that tricks users into running it. (give full control to a stranger)
  - May contain a logic bomb, or be a vehicle for spreading a virus
- Viruses cannot run by itself, relies on the execution of other programs
  - A program that can “infect” other programs by copying itself into them
- Worms
  - Program that spreads via network connections, do not need to attach to another program unlike viruses

Stack & Buffer Overflow Attacks



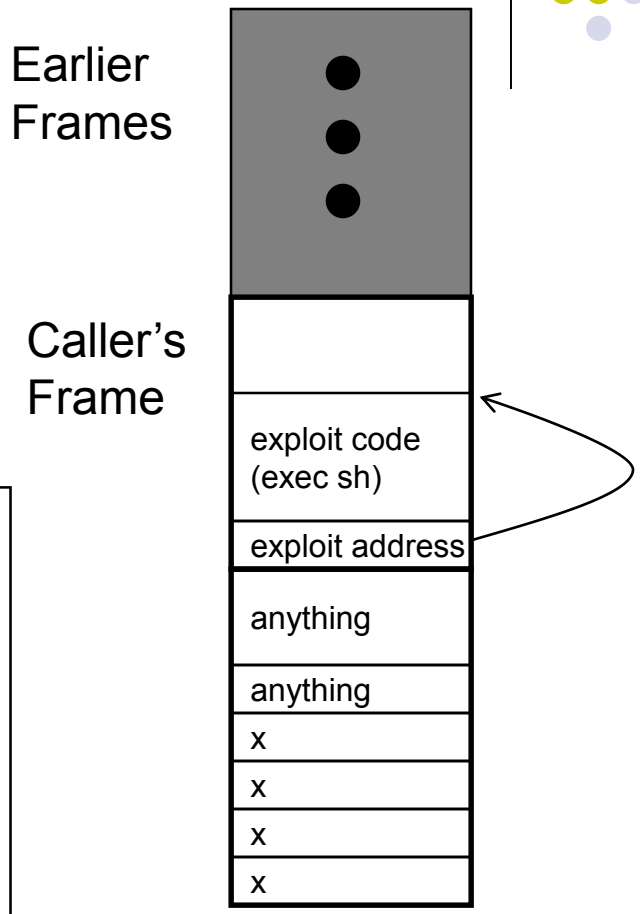
- Most common means of gaining unauthorized access to a system
- How it works:
  - Overflow some stack-allocated input buffer past the space allocated for the input
  - Overwrite the return address with the address of exploit code
  - Overwrite next space in stack with the exploit code itself
- ??? Let's see what this looks like…

Stack Attack

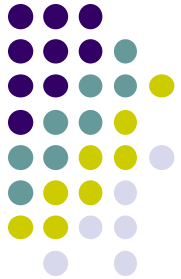


Increasing Addresses

```
sloppy() {
   int i;
   char In[4];
   printf("Continue
          (Y/N)?\n");
   gets(In);
   if (In[0] == 'N')
      return;
   /* more code... */
}
```

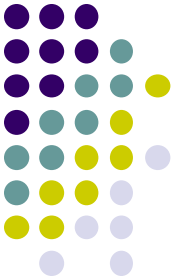


Trusted Computing Base (TCB)



- Think carefully about what you trust with your data
  - If you type your password on a keyboard, you're trusting
    - The keyboard manufacturer
    - Your computer manufacturer
    - Your OS
    - The password library
    - The application that is checking the password
  - TCB = set of components (hardware, software, people) that you trust your secrets with
- Public Web kiosks should not be in your TCB

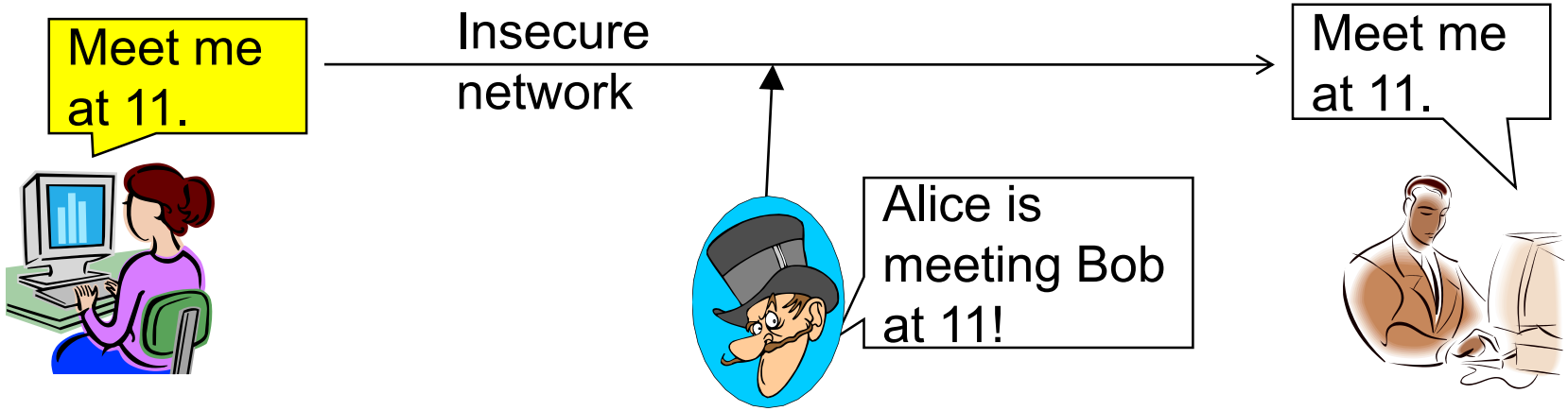
Network Communications



- Protecting “assets” on a single computer system with multiple users is hard, but dealing with networks is even harder
  - Multiple administrative domains
  - Users of network may not have common interests
- Two main categories of network attacks
  - Passive - eavesdropping or monitoring communications
  - Active - modification or tampering with the communication stream

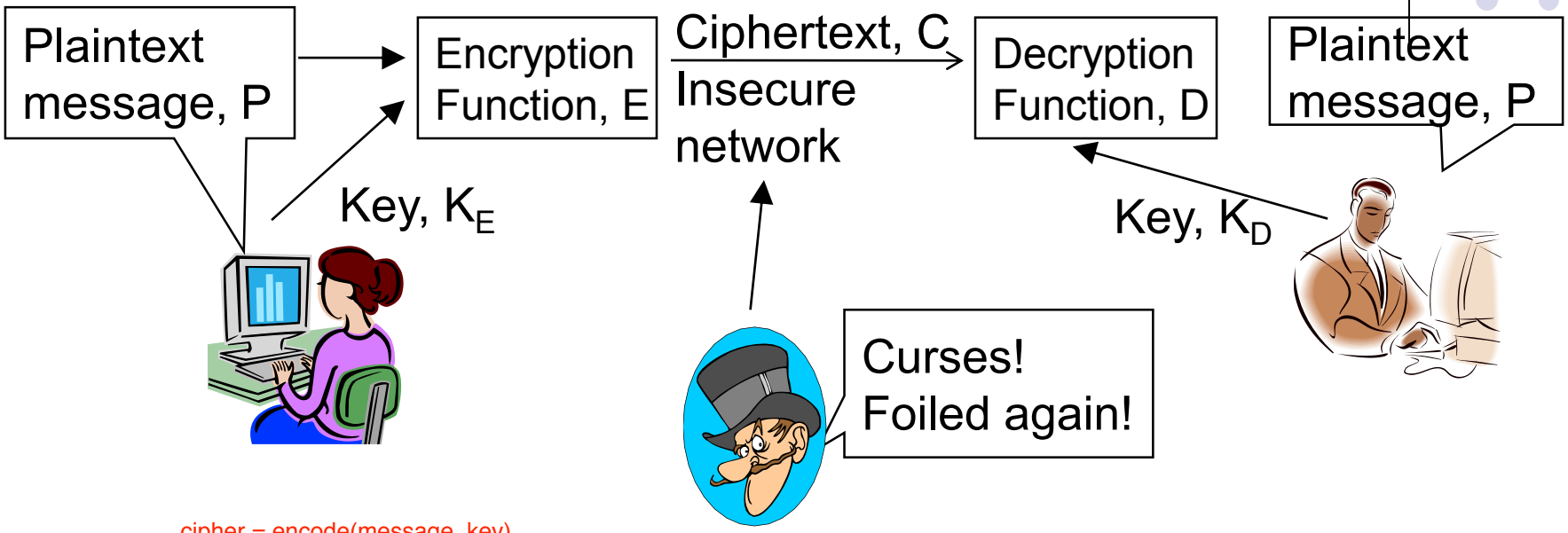
Passive Network Attacks

- *Eavesdropping* - read network packets that were not intended for attacker
  - Does not interfere with delivery of message to intended recipient



  - Defense: obfuscate message contents so attacker gains no information from intercepting the message (i.e., encrypt the message)

Encryption basics

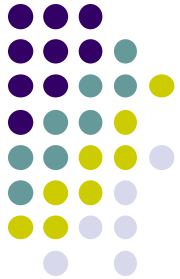


cipher = encode(message, key)

- $C = E(P, K_E)$
- $P = D(C, K_D) = D(E(P, K_E), K_D)$

message = decode(cipher, key)

Encryption Keys



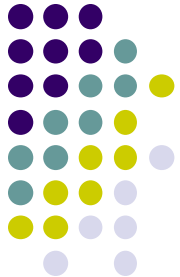
- Keys used for encryption and decryption can be the same (symmetric, or secret-key) or different (public-key)
- Secret-key cryptography
  - ✓ Fast encryption/decryption algorithms are known
  - ✘ Distributing the shared secret key is a problem
- Public-key cryptography
  - Pair of keys, one to encrypt, one to decrypt
  - ✓ Encryption key can be published, decryption key is kept secret; knowledge of one key does not reveal other
  - ✘ Encryption/decryption algorithms ~1000X slower

More Passive Attacks

analyze the system

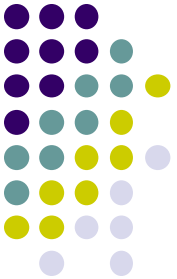
- Traffic Analysis - infer information based on sender/recipient of messages, size of messages, frequency of communication, etc.
  - Encryption doesn't help here
  - Defense: obfuscate communication pattern (i.e., make all messages the same length, all communications have same number of messages, etc.)

Active Network Attacks



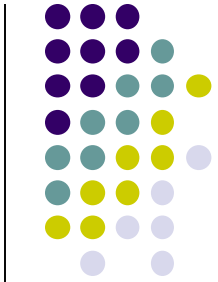
- Replay - capture a legitimate message and re-send it later to produce unauthorized effect
  - e.g. unauthorized file accesses
  - Defence: “nonce” - messages include an item so they can’t be reused
- Modification of messages - alter contents of message to change effect
  - Include *message digest* to detect tampering
- Masquerade - attacker pretends to be another entity or user
  - Often combined with another attack like replay

Message Digests



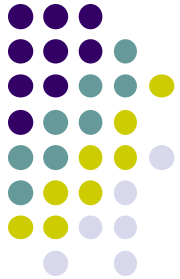
- Given some input data of arbitrary length, compute a fixed-length output
  - Relies on one-way function
    - Given input x, computing digest y = f(x) is “easy”
    - Given y, finding x is computationally infeasible
    - So attacker cannot construct a new input that will have the same digest as the original
- To detect tampering, compare digest computed on received message against digest of sent message
  - Need to know original digest
- Sender can encrypt or sign digest only
  - Or distribute digest separately from message

Digital Signatures



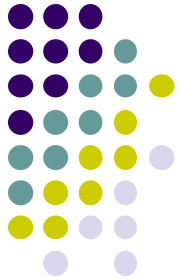
- Provide a way to verify the origin and integrity of a message (i.e. authenticity)
- Sender includes something with the message (or modifies the message in some way) that could only be generated by the sender
- Public-key cryptography helps here
  - Signatures – encrypt message with sender's private key; anyone can decrypt using sender's public key, thereby verifying that sender created the message
  - Typically, only a message digest is signed to reduce overhead

One more active attack



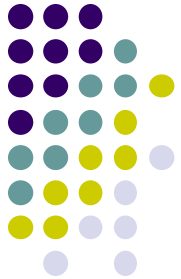
- Denial of service - system asset becomes unavailable to legitimate users
  - Attack on availability
  - Common network attacks - overload server with bogus requests so there are no resources for legitimate ones (e.g. TCP SYN flooding)
  - Extra evil - distributed DOS - compromise a set of computers and use them as “zombies” in a coordinated attack on the victim
- Active attacks are harder to defend against
  - Need measures to detect and recover from them

Security Design Principles



- Security is much, much more than just crypto
  - If there is a fundamental flaw in the design of the system, then all of the crypto in the world won't help you
  - It is usually easier to find a bug in an implementation than circumvent a crypto systems
- Unfortunately, systems design is still as much an art as it is a science
  - But, decades of building systems the wrong way have helped us gain some learned wisdom

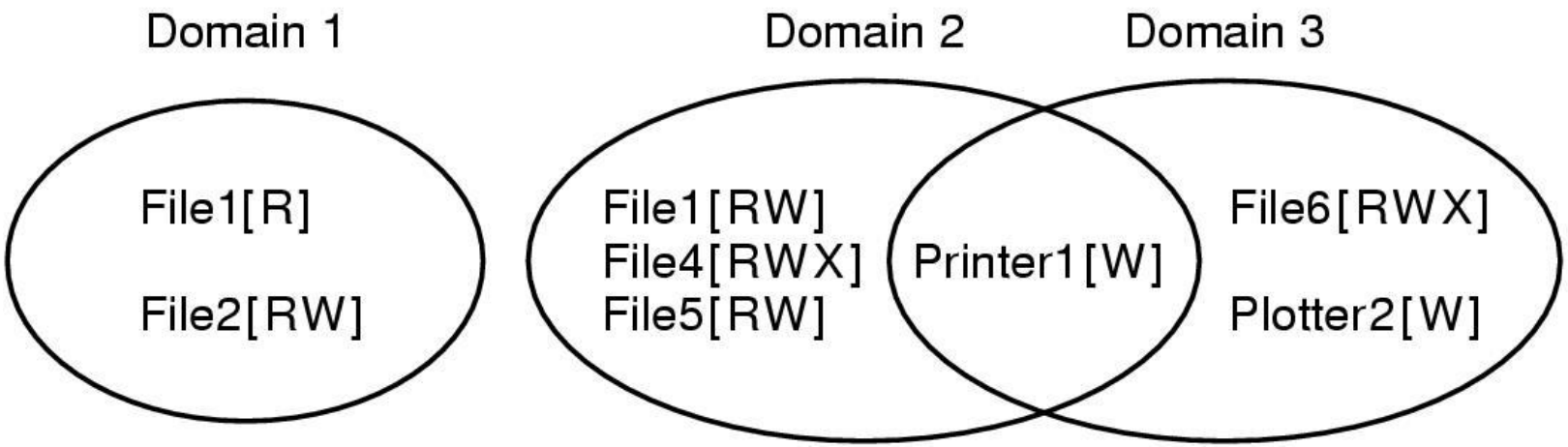
Principle of Least Privilege



- Figure out exactly which capabilities a program needs to run, and grant it only those
  - Not always easy, but one algorithm: start with granting none, run and see where it breaks, add new privileges, repeat

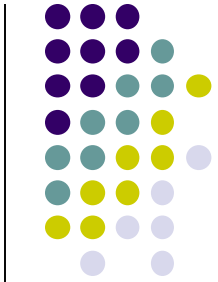
- Domains
  - A set of objects and rights
  - Each pair
    - An Object
    - Set of rights

Protection Domains



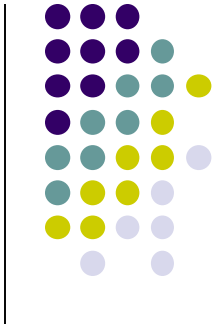
Three protection domains.

Principle of Least-Common Mechanism



- Example UNIX Process
  - User part
  - Kernel part
- System Call switches the domain
  - The rights will be changed
- How system tracks which object belongs to which domain?

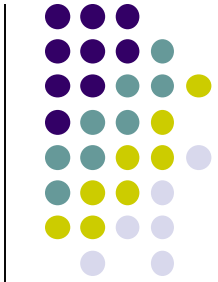
Protection Domains



	Object							
	File1	File2	File3	File4	File5	File6	Printer1	Plotter2
Domain 1	Read	Read Write						
2			Read	Read Write Execute	Read Write		Write	
3						Read Write Execute	Write	Write

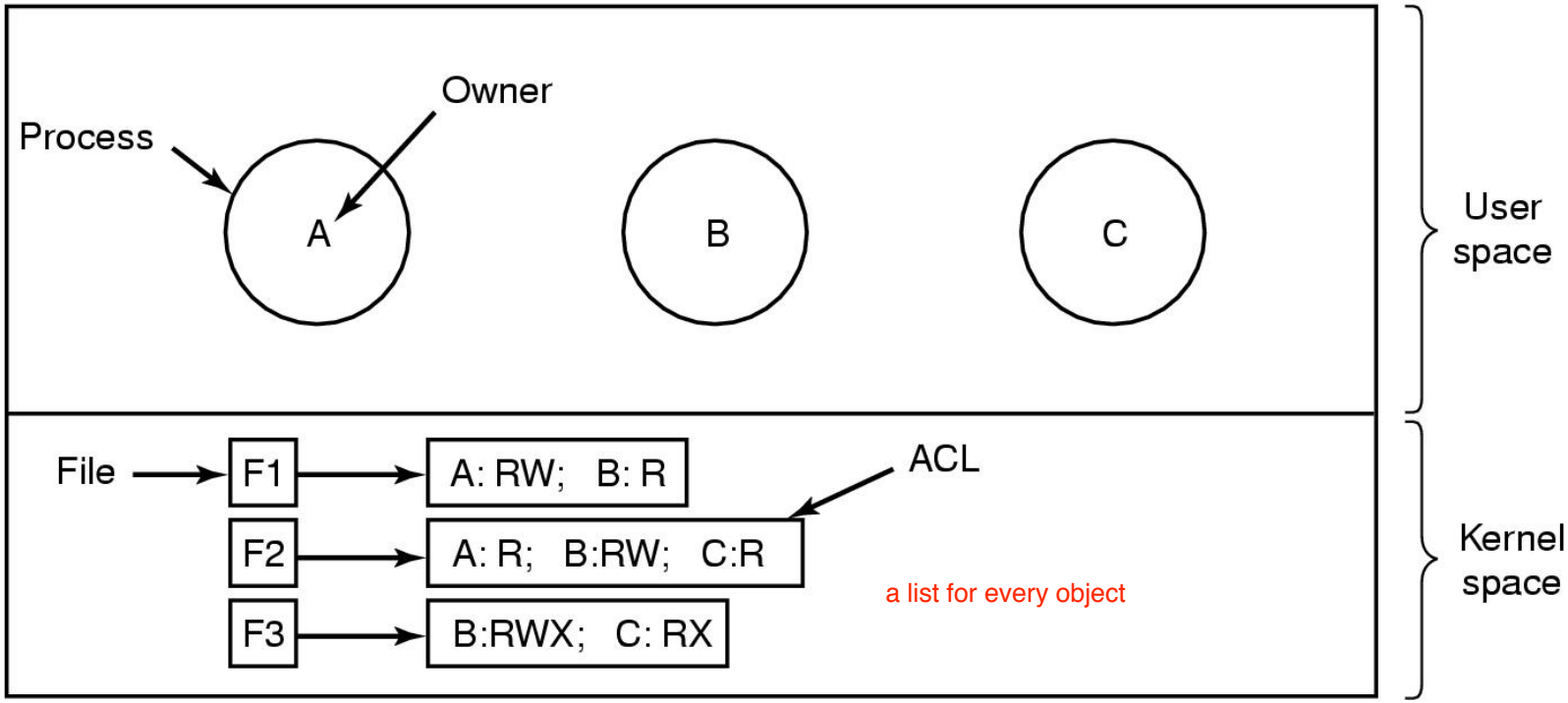
A protection matrix.

Protection Domains



- Problem with the protection matrix?!
  - It's too costly too store such a sparse matrix
- Access Control List
  - Associate with each object an ordered list
  - The list contains all the domains that may access the object

Access Control Lists

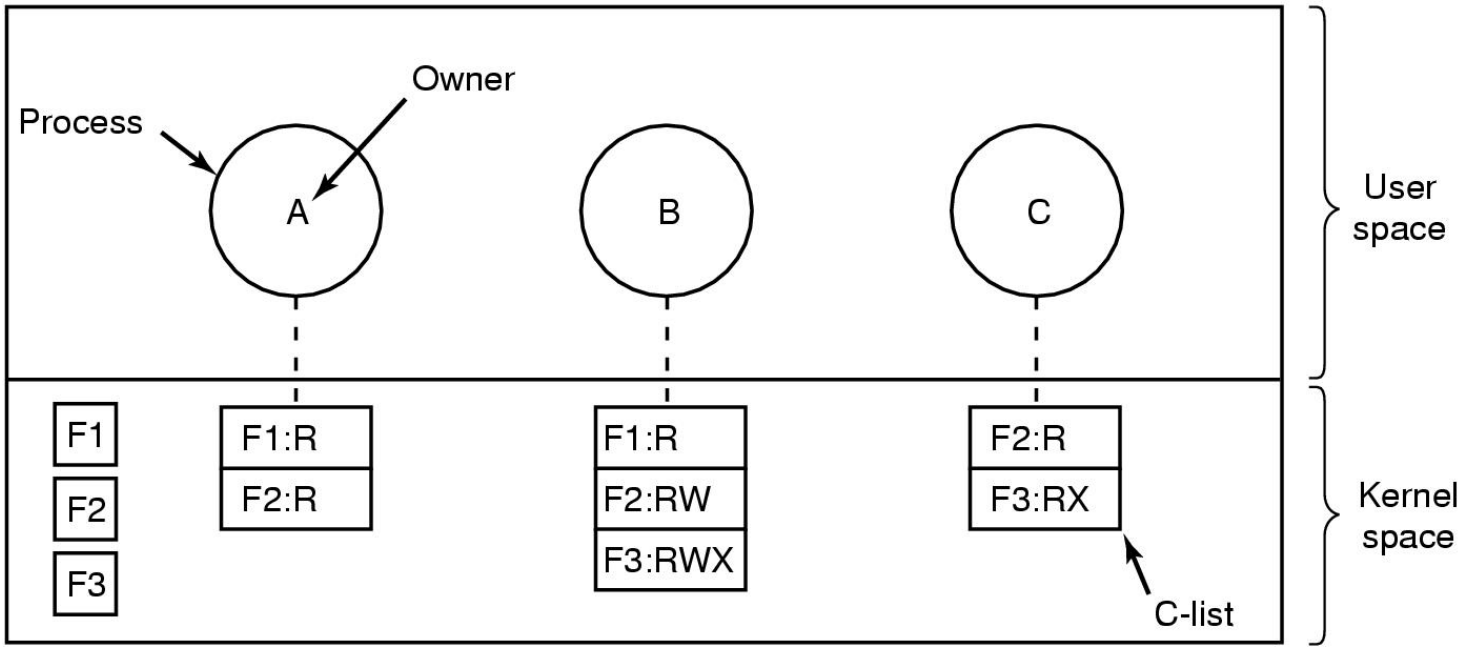


File	Access control list
Password	tana, sysadm: RW
Pigeon_data	bill, pigfan: RW; tana, pigfan: RW; ...

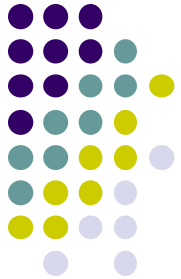
Use of access control lists to manage file access.

Protection Domains

- Capabilities
  - With each process is a list of objects that may be accessed

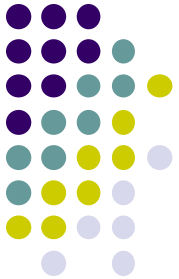


Outlook For The Future



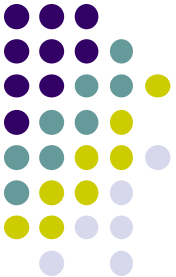
- Doesn't look bright…
  - More and more complex systems are being deployed
    - More and more lives are being trusted to them
- Bruce Schneier: 3 waves of security attacks
  - 1st wave: physical attacks on wires and hardware
    - Physical security to defend against this
  - 2nd wave: syntactic attacks on crypto protocols and systems
    - E.g., buffer overflows, DDoS attacks
  - 3rd wave: semantic attacks: humans and computers trust information that they shouldn't
    - E.g., falsified press announcements
      - Emulex corp stock hoax: CEO "resigns", 61% stock drop

Case Study: SSL



- Recall crypto basics
  - Messages sent over insecure networks are *encoded* using a secret *key*
  - Only parties with the correct key can decode the message
  - Two main types of keys
    - Symmetric, or private key, cryptography
      - Same key is used to encode and decode
      - Encode/decode operations are computationally fast
    - Asymmetric, public key, cryptography
      - 2 keys are needed, one is kept secret, the other is public
      - Messages encoded with one key can be decoded with the other
      - Enables both privacy and authentication
      - Encode/decode operations are computationally expensive

Secure Sockets Layer



- SSL == Secure Sockets Layer
  - Used for secure communications
  - Typically seen in web services (https://)
  - Can be used for any service above TCP/IP layer
- Uses both public key and private key cryptography
- Communication begins with a handshake protocol between client and server to establish identity and set up session keys used to encrypt remainder of the transmissions
  - Requires the existence of a trusted Certification Authority (CA)
    - CA issues certificate to verify server's identity (i.e., its name, IP address, and public key), signed with CA's private key
    - Client can verify certificate (and thus server identity) using CA's public key
    - Web applications ship with a (long) list of CA's pre-installed

SSL Handshake Protocol (1)

crypto info, a list of algo supported by client

ClientHello(version, random_c, session id, ciphers)

ServerHello(version, random_s, session id, ciphers)

ServerCertificate(certificate list)

client verifies server certificate with CA

server public key in certificate
a certificate is generated by certificate authority (CA)

ServerHelloDone()

Client verifies server identity. Generates random *pre-master secret*. Selects cipher.

only server can decrypt random number with private key

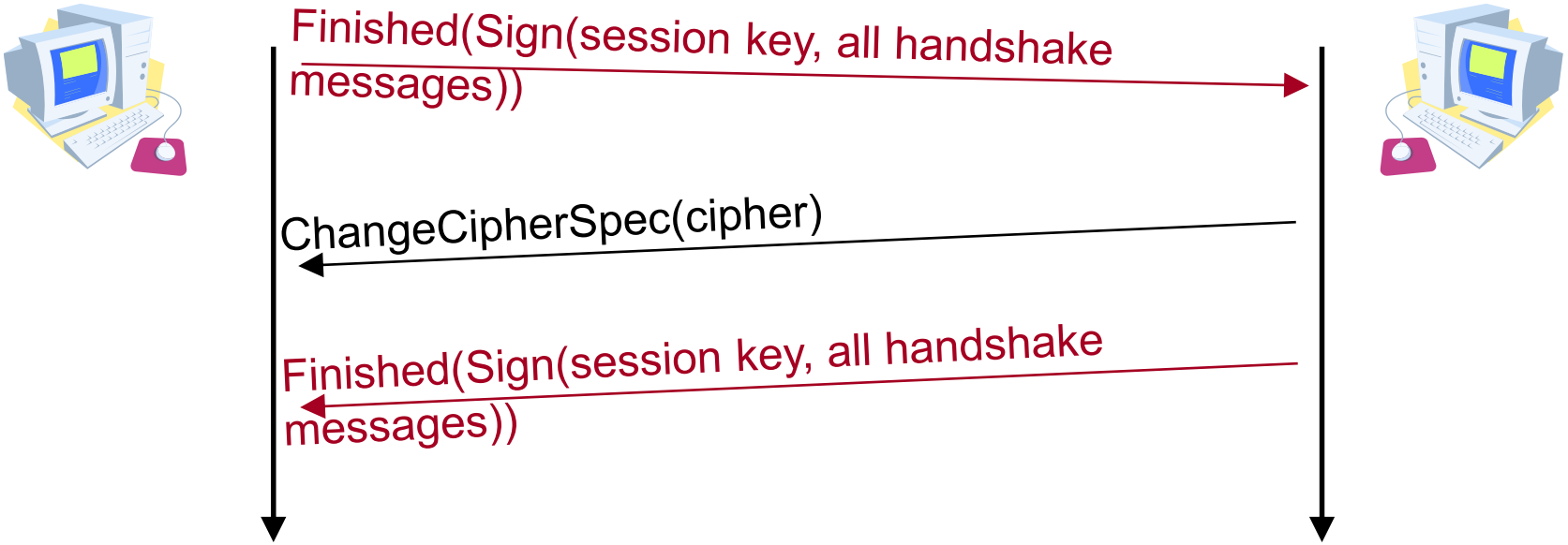
ClientKeyExchange(E(server public key, pre-master secret))

ChangeCipherSpec(cipher)

Now client & server each generate shared secret session keys using selected cipher and (random_c, random_s, pre-master secret)

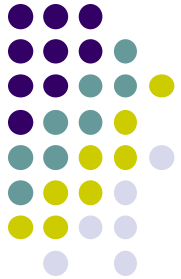
encryption -> need public key
decryption -> need both private + public key

SSL Handshake Protocol (2)



All future communication is encrypted using session key
("Finished" messages are encrypted and signed)

If OSs were Beer…



DOS Beer

- Requires you to use your own can opener, and requires you to read the directions carefully before opening the can. Originally only came in an 8-oz. can, but now comes in a 16-oz. can. However, the can is divided into 8 compartments of 2 oz. each, which have to be accessed separately. Soon to be discontinued, although a lot of people are going to keep drinking it after it's no longer available.

MAC Beer

- At first, came only a 16-oz. can, but now comes in a 32-oz. can. Considered by many to be a "light" beer. All the cans look identical. When you take one from the fridge, it opens itself. The ingredients list is not on the can. If you call to ask about the ingredients, you are told that "you don't need to know." A notice on the side reminds you to drag your empties to the trashcan.

If OSs were Beer…

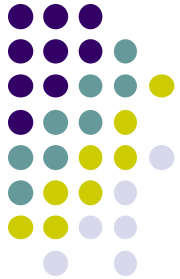
Windows 2000 Beer

- The manufacturer of the Windows line of beers says this will be "the" beer, if they can just finish playing with the ingredients. This beer will have many ingredients of Windows 95/98 and NT beers. Many drinkers in the future will be forced to drink this when they get thirsty since they won't be able to find Windows 95 or 98 or NT beer on the shelves. According to manufacturer it combines the greatest taste ever with almost no calories. Only one problem, the cans explode without warning and take out half the refrigerator with them.

Unix Beer

- Comes in several different brands, in cans ranging from 8 oz. to 64 oz. Drinkers of Unix Beer display fierce brand loyalty, even though they claim that all the different brands taste almost identical. Sometimes the pop-tops break off when you try to open them, so you have to have your own can opener around for those occasions, in which case you either need a complete set of instructions or a friend who has been drinking Unix Beer for several years.

If OSs were Beer…



Linux Beer

- LINUX beer tastes just like Unix beer. Like Unix beer, Linux beer is intended for expert beer drinkers only. It originally had no pop tops or cans because you had to brew it yourself. First you would get a recipe and some yeast from a Unix guru. Then go plow a field, plant your barley and hops. After harvest you would take your Kernels and put them into a barrel full of water, then you just add your yeast close the lid, and let your beer compile. After all this you have what experts claim to be one of the Worlds Best Beers. Linux beers do not normally explode but many brewers have been known to produce such beers. Linux beer is now available from some Micro Brewerys in handy pop top versions for easy drinking by beginner Unix or Linux beer drinkers. Keep your can openers handy.