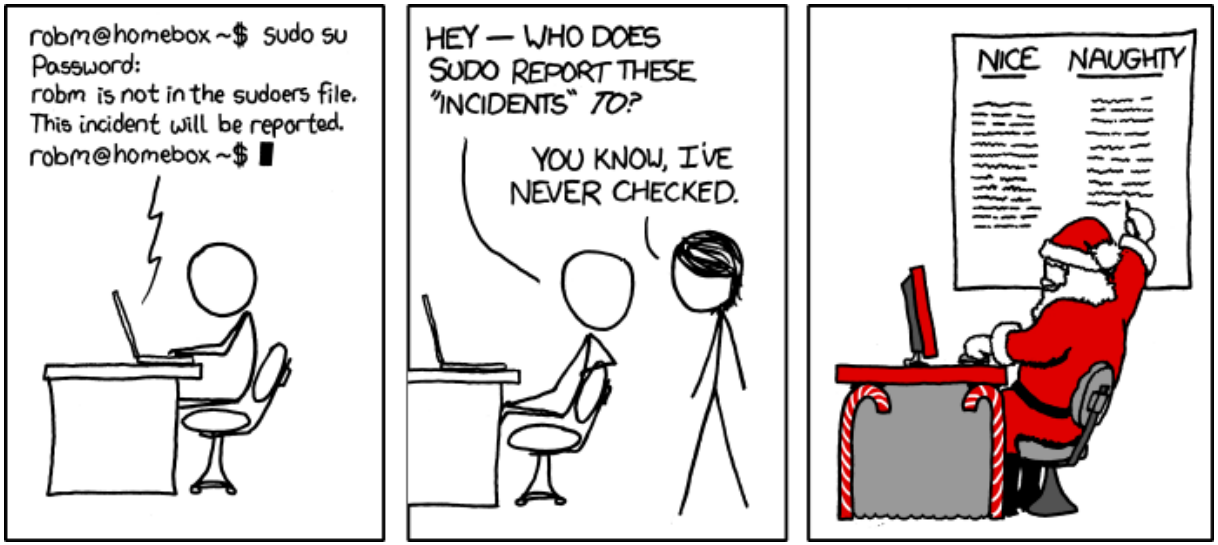
# Lecture 22: Audit

CS 138 Spring 2026

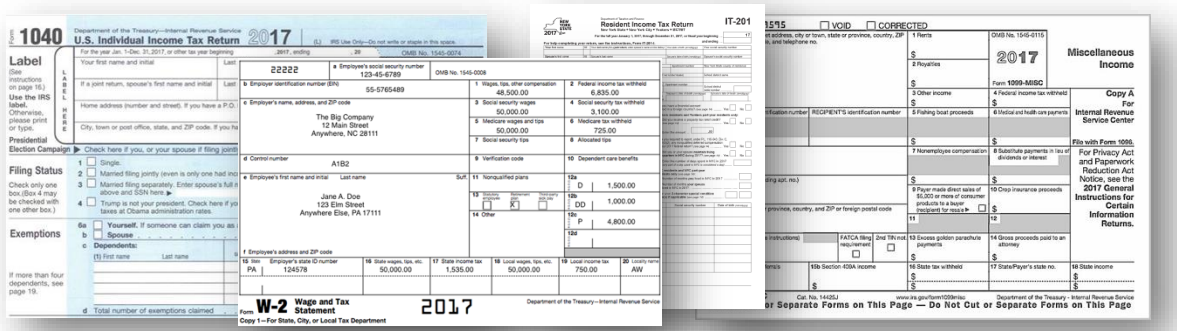


# Classes of Countermeasures

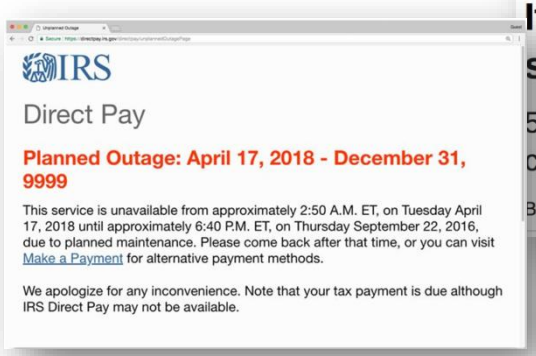
- **Authentication:** mechanisms that bind principals to actions
- **Authorization:** mechanisms that govern whether actions are permitted
- **Audit:** mechanisms that record and review actions

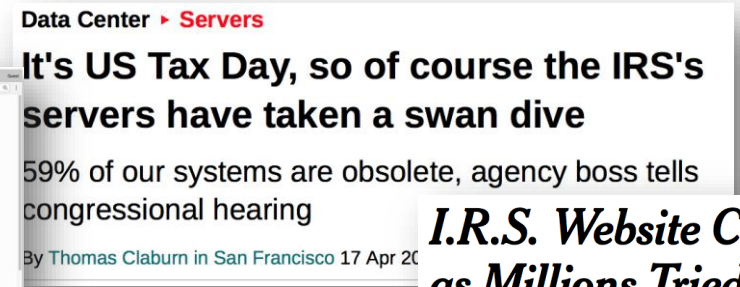
# Uses of audit

- **Deterrence through accountability:** deter misbehavior




- **Detection and recovery:** determine what happened and how to recover





- **Problem monitoring:** real-time intelligence

# Audit tasks

- **Recording:**
  - what to log
  - what not to log
  - how to log
    - locally
    - remotely
  - how to protect the log
- **Reviewing:**
  - automated analysis
  - manual exploration

# WHAT TO LOG

# What to log?

**Example:** US State Department pilot program (1980s)

- Requirements:
  - log every transaction related to protected electronic documents
  - system administrator reviews log daily to search for malicious behavior
- Experiment:
  - test system for 5 users, 10 minutes
  - audit log was a stack of paper
  - real system would have been 1000s of users working 24/7
- Lessons learned:
  - logging and review of everything by a human is impractical
  - need to reduce information logged: log reduction
  - need automated review

# States vs. events

- **States:** data, *what the system is*
  - backup, or more
  - survive power failures, crashes, attacks
  - what state? memory, disk, network, ...
  - consistent snapshot of distributed system is hard

- **Events:** actions, *how the system came to be*
  - login, access to protected resource, elevation and attenuation of privileges, ...
  - our focus
  - which events?

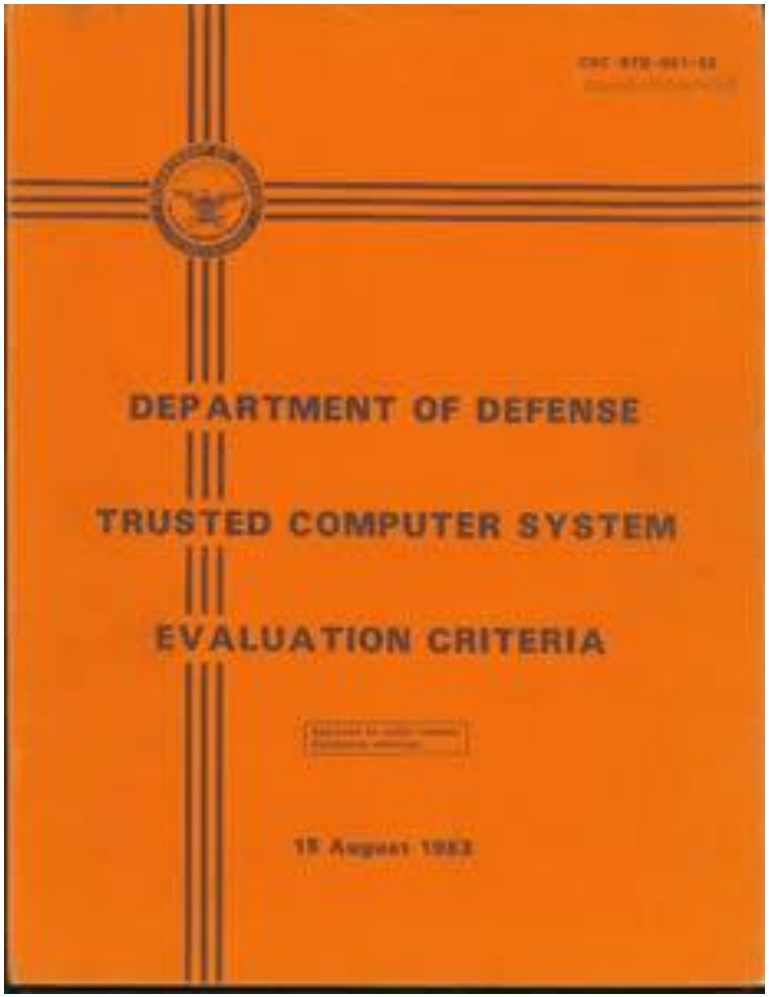
# Recall: Security requirements

- **Functional requirement:** something system should do
  - e.g., allow people to cash checks
- **Security goal:** something system should/shouldn't do
  - e.g., prevent loss of revenue through bad checks
- **Security requirement:** constraint on functional requirement to achieve goal
  - e.g., check must be drawn on bank where being cashed, or person cashing must be customer at that bank and deposit in their account

# Events to log

- **Any event that involves a security requirement**
  - Fact that requirement was checked
  - Whether it was met or not
  - The information that led to that decision
- Typically involves the gold standard...
  - whether a **principal was authenticated**, or
  - whether an **action was authorized**

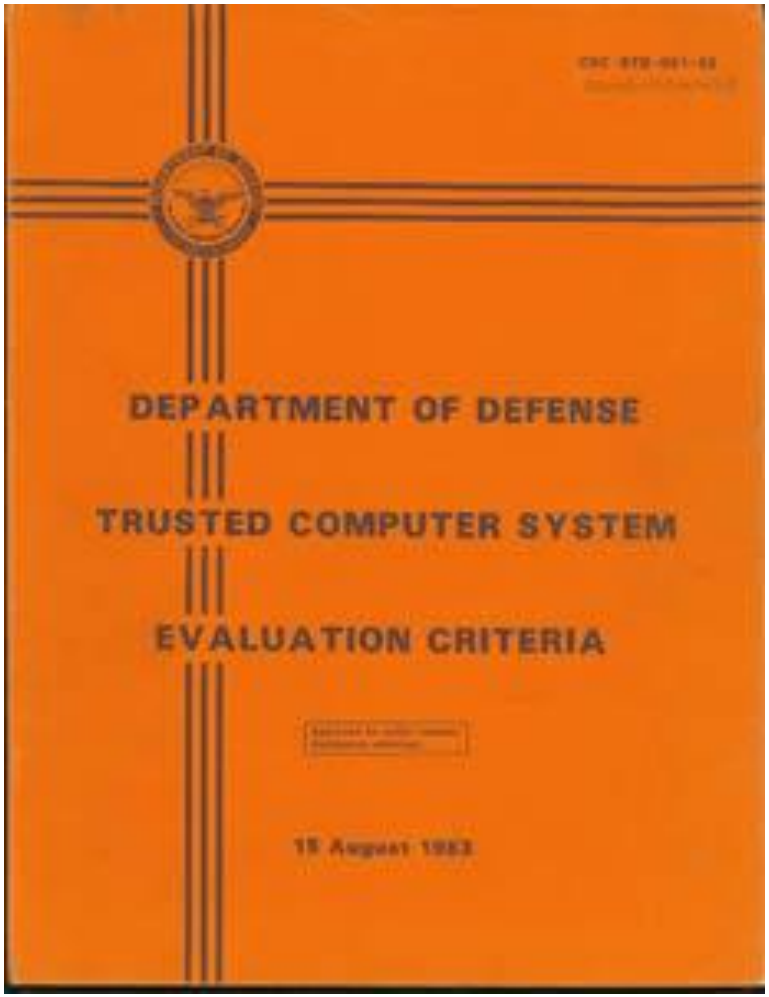
# Orange Book logging




For minimal C2 level certification:

- **Events** to log:
  - Use of identification and authentication mechanisms
  - Introduction of objects into a user's address space (e.g., file open, program initiation)
  - Deletion of objects
  - Actions taken by computer operators and system administrators and/or system security officers

# Orange Book logging

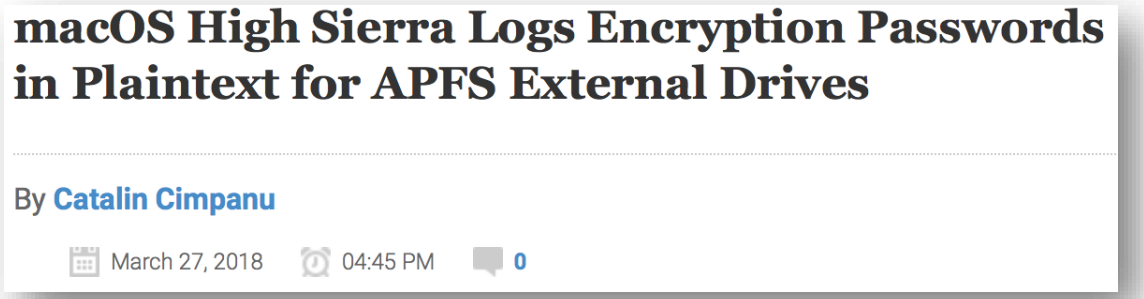



For minimal C2 level certification:

- **What** to log:
  - Date and time of the event
  - User
  - Type of event
  - Success or failure of the event
  - For identification/authentication events: origin of request
  - For events involving objects: name of the object

# What not to log

- Some information might be too sensitive for log files:
  - cryptographic keys, passwords
  - the details of company's shiny new product
  - the GPS coordinates of undercover secret agents




- Possibilities:
  - log it anyway, protect the log
  - sanitize log

# Sanitization

Protect confidential information in log

- by **deleting**
- by **modifying**
  - e.g., replace with user names with pseudonyms, keep separate protected map between names and pseudonyms

# Sanitization

- Before writing to log:
  - **Pro:** protects users from system administrators; maybe surveillance warranted only with probable cause
  - **Con:** have to decide in advance, as part of system design, what information to keep vs. discard
- After writing to log:
  - **Con:** confidentiality of log must be (more) protected
  - **Pro:** can decide afterwards what information to discard, perhaps even redact logs and send to 3rd party for analysis

# Exercise

- Imagine that you are designing the log for a system. What events would you log? What information would you include for each log entry?

- Possible systems:
  - Gradescope
  - Instagram
  - the 5C SSO
  - a banking app/site

# HOW TO LOG

# Say what you mean

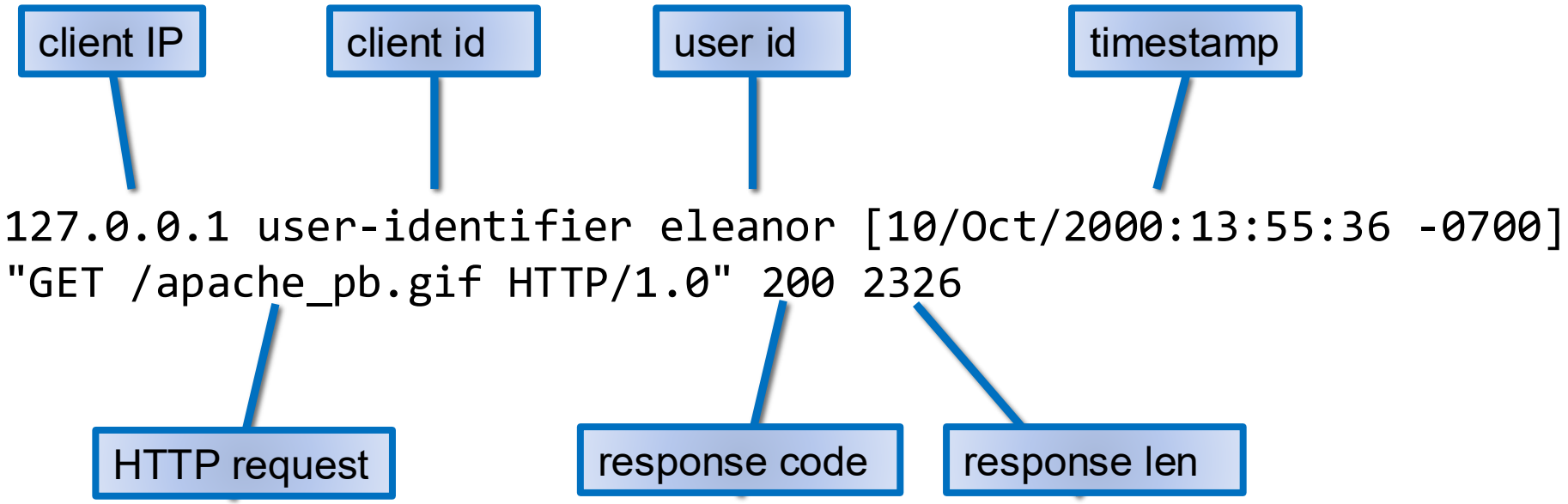
**Main principle:** Every log entry should say what it means

- Interpretation of log entry should depend only upon content of log entry
- Hence reviewer can recover meaning without needing to assume or supply any context

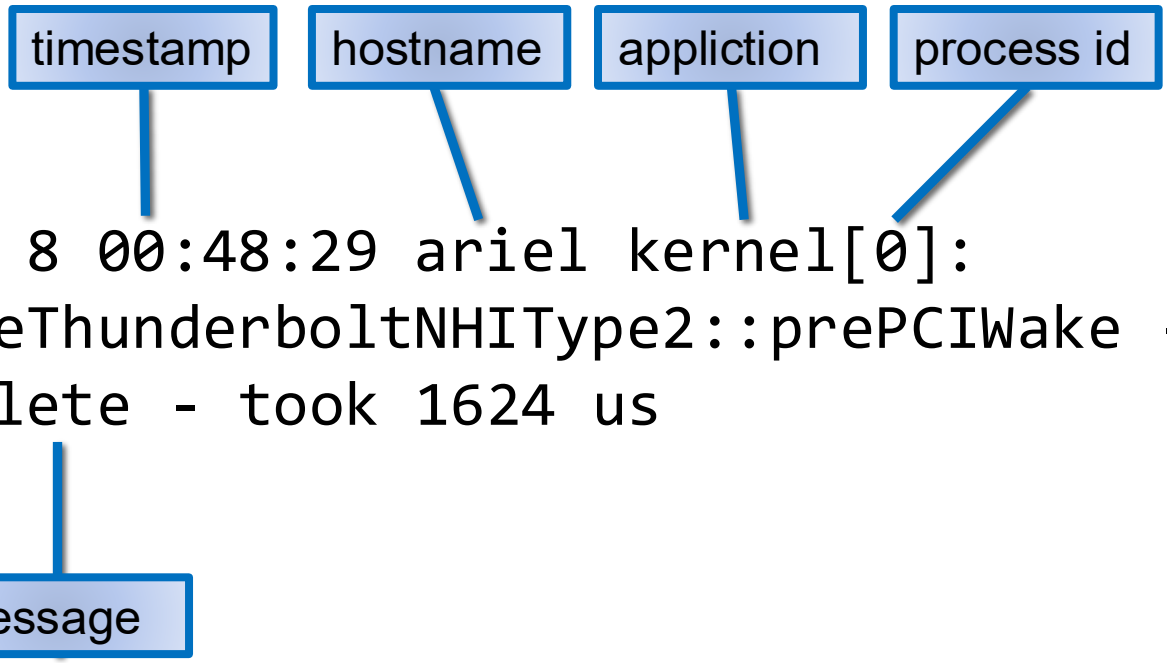
# Log file format

- Keeping log files in standard format enables...
  - Reuse of tools for log analysis
  - Correlation across logs from multiple applications
- Standard formats:
  - Common Log Format (used by web servers)
  - syslog (used by Unix)
    - originated with sendmail
    - became a *de facto* standard
    - then standardized by IETF: RFC 5424
    - examples: take a look in your local /var/log directory

# Common Log Format



# syslog example message



# Log space

What happens if log size grows too large?

- Halt system
- Overwrite previous entries
- Stop logging

# REVIEWING THE LOG

# Manual review

- Enable administrators to explore logs and look for {states, events}
- **Issues:**
  - Designers might not have anticipated the right {states, events} to record
  - Visualization, query, expressivity (HCI/DB issues)
  - Correlation amongst multiple logs

# Interfaces

- **Flat text**
- **Hypertext**
- **DBMS**
- **Visualization tools**

# Techniques

- Temporal replay: animate what happened when
- Slice: display minimal set of log events that affect a given object

# Automated review and response

- **Review:** detect suspicious behavior that looks like an attack, or detect violations of explicit policy
  - Custom-built systems
  - Classic AI techniques like training neural nets, expert systems, etc.
  - Modern applications of machine learning
- **Response:** report, take action

# Exercise

- Imagine that you have designing the log for a system. How should this log be used? What tools/visualizations/guidelines would you provide to make the best use of this log?

- Possible systems:
  - Gradescope
  - Instagram
  - the 5C SSO
  - a banking app/site

# Exercise

- Start designing an audit system for your project
  - what will you log?
  - how will you log it?
  - what will you do with it?

# Audit


robm@homebox ~$ sudo su
Password:
robm is not in the sudoers file.
This incident will be reported.
robm@homebox ~$
HEY — WHO DOES SUDO REPORT THESE "INCIDENTS" TO?
YOU KNOW, I'VE NEVER CHECKED.
NICE
NAUGHTY