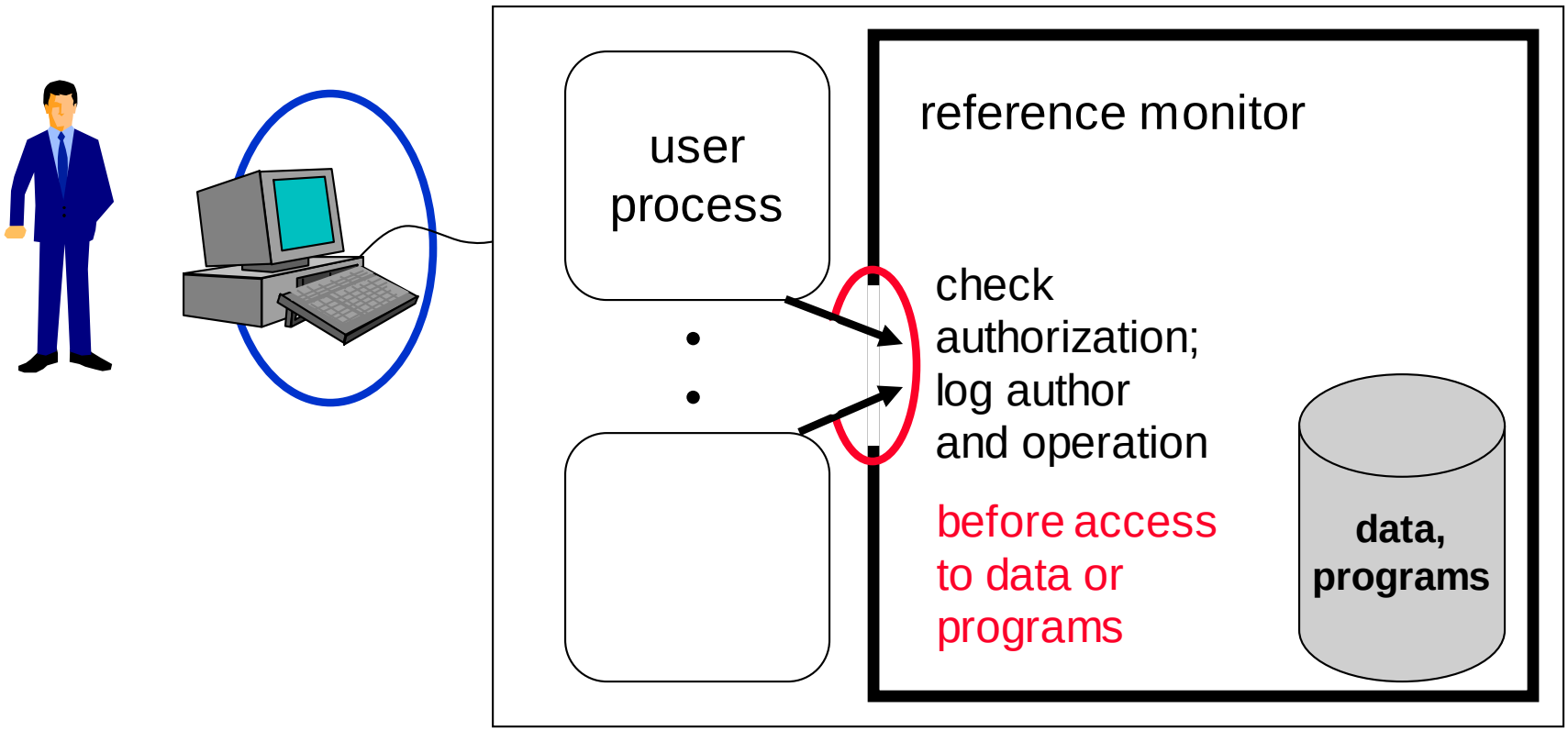
AUTHENTICATION

Slides are from Andreas Pfitzmann, Andreas Westfeld, Sebastian Clauß, Mike Bergmann and myself (Stefan Köpsell)

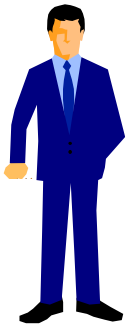
Lookahed: Why autentication: Admission and access control

Admission control communicate with authorized partners only



Access control subject can only exercise operations on objects if authorized.

Identification of human beings by IT-systems



What one *is*
- hand geometry
- finger print
- picture
- hand-written signature
- retina-pattern
- voice
- typing characteristics

has
- paper document
- metal key
- magnetic-strip card
- smart card (chip card)
- calculator

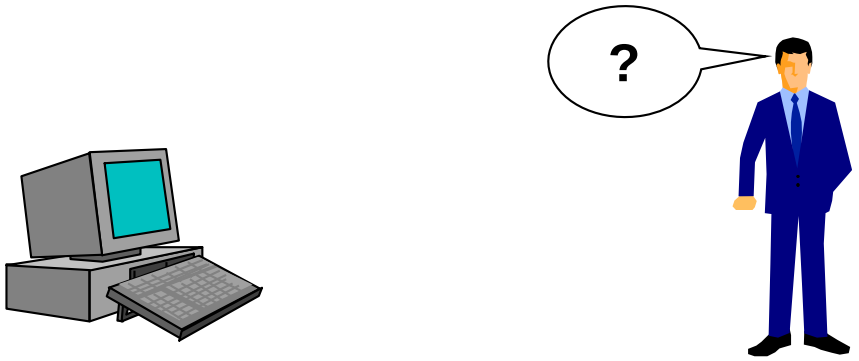
knows
- password, passphrase
- answers to questions
- calculation results for numbers

→ **eID-card**

New German eID Card

PIN protects access to chip

Identification of IT-systems by human beings

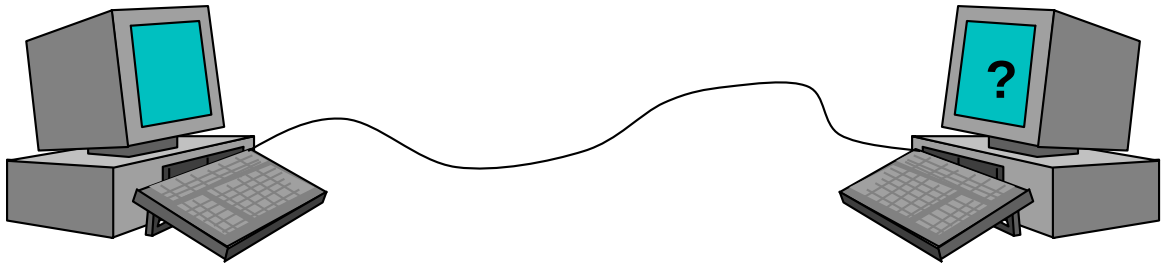


What it *is*
- casing
- seal, hologram
- pollution

knows
- password
- answers to questions
- calculation results for numbers

Where it *stands*

Identification of IT-systems by IT-systems



What it *knows*

- password
- answers to questions
- calculation results for numbers
- **cryptography**

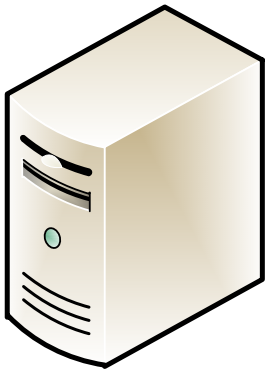
Wiring *from where*

Password based authentication

- Simple approach

User

(dog,bone)



Server

?

yes

grant access

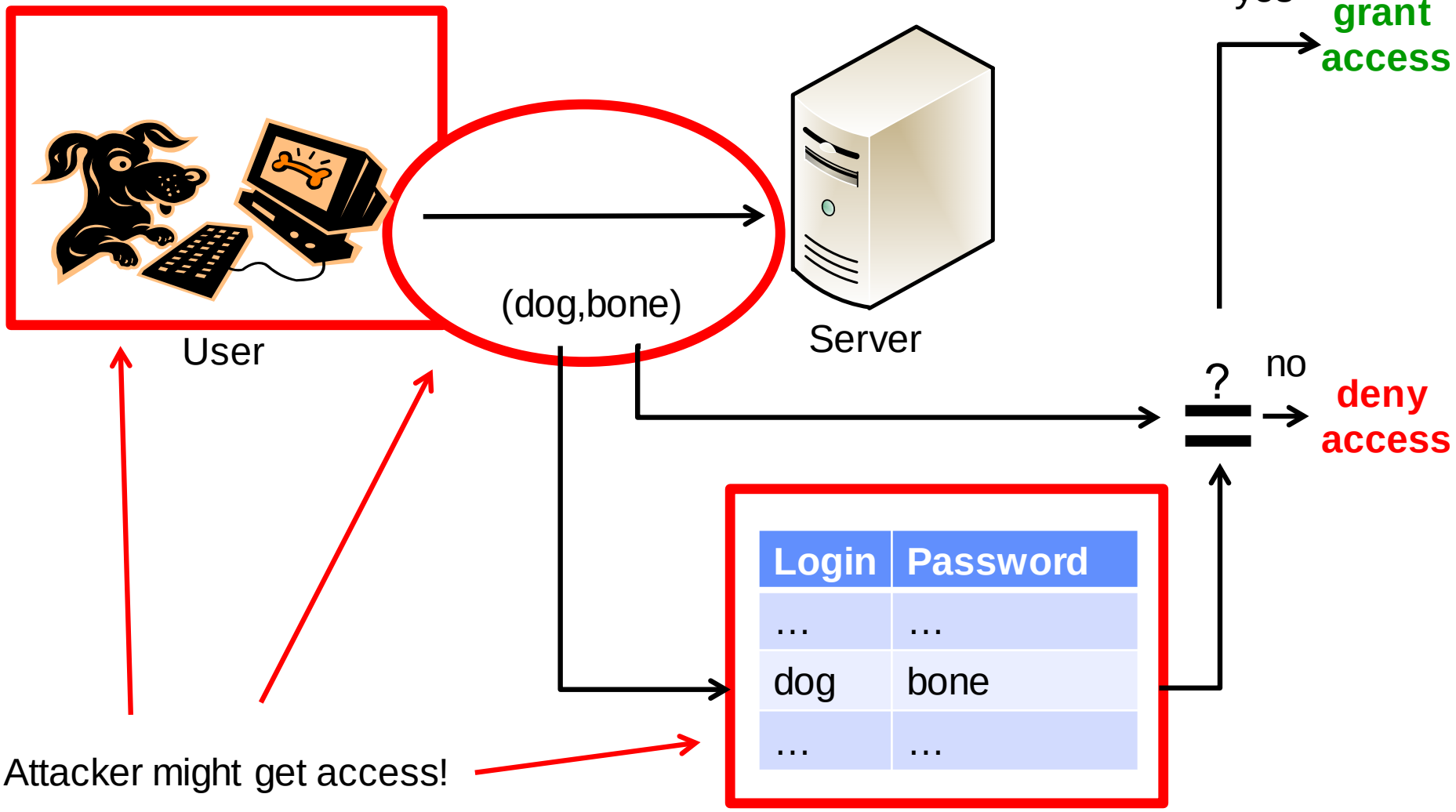
no

deny access

Login	Password
…	…
dog	bone
…	…

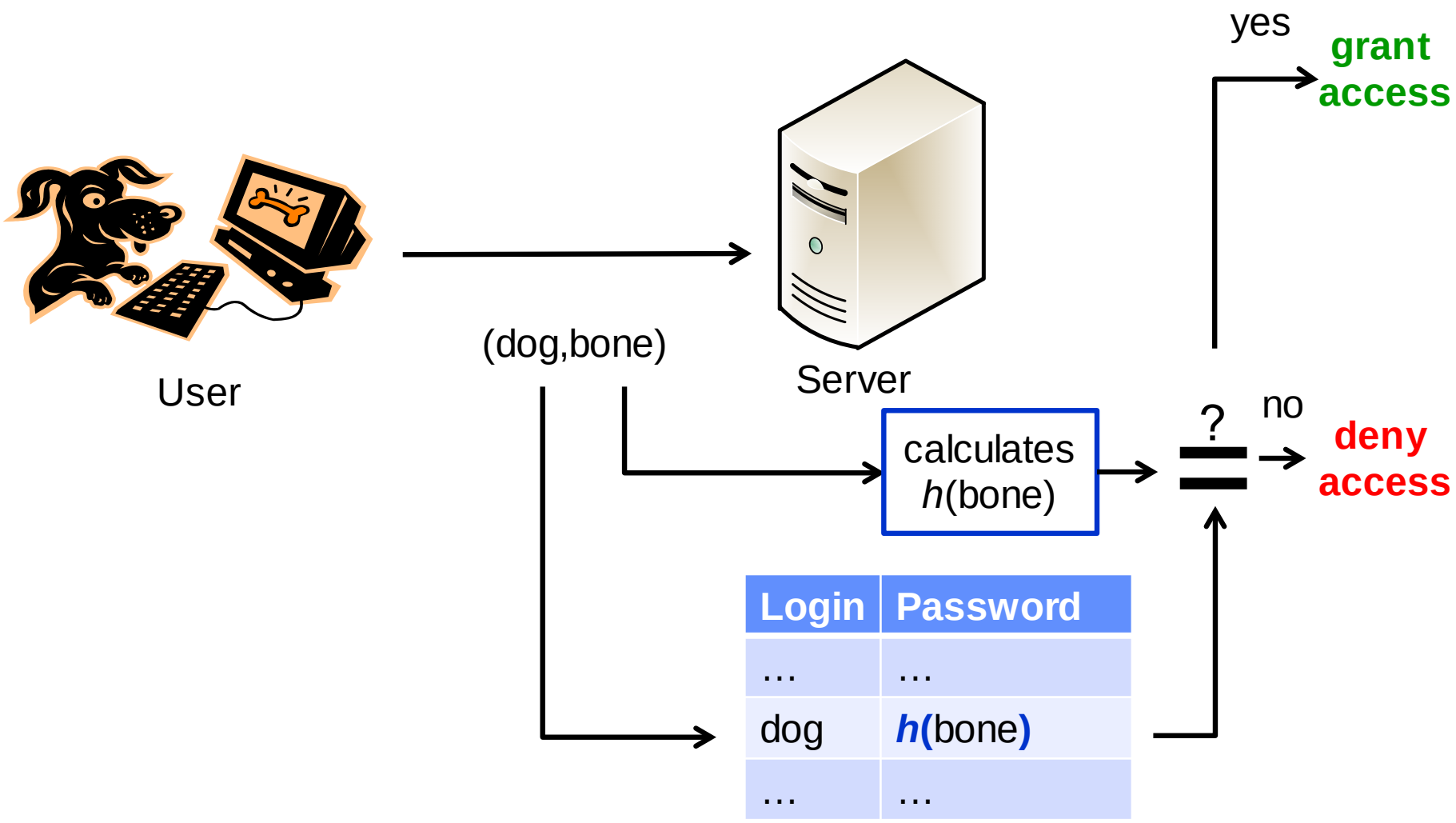
Password based authentication

- Simple approach – **security problems**



Password based authentication

- Enhanced approach using one way (hash) functions



One-way functions – cryptographic hash functions

- One-way function *f:*
  - calculating $f(x)=y$ is easy
  - calculating $f^{-1}(y)=x$ is hard
    - computation / storage
  - open question: Do one-way functions exist?
- Cryptographic hash function *h*
  - might have different properties depending on the use case
  - collision resistance:
    - it is hard to find $h(y)=h(x)$ with $y \neq x$
    - note: *h* is usually not *collision free*, because $|h(x)| \ll |x|$
  - preimage resistance / one-way function / secrecy
    - given $h(x)$ it is hard to find x
  - second-preimage resistance / weak collision resistance / binding
    - given $h(x)$ it is hard to find $h(y)=h(x)$ with $y \neq x$
  - Note:
    - *h* is not necessarily a “random extractor”
    - only one of “secrecy” and “binding” can be information theoretic secure

Examples for cryptographic hash functions

- MD5
  - Message-Digest Algorithm
  - developed by Ronald Rivest (April 1992)
  - produces 128 bit hash values
  - can process arbitrary long inputs
  - **today MD5 is broken!**
- SHA-1
  - Secure Hash Standard
  - published 1993 as FIPS PUB 180 by US NIST
  - produces 160 bit hash values
  - **today SHA-1 is insecure!**
- SHA-2
  - set of hash functions, with hash values of 224, 256, 384, 512 bit
  - published 2001 as FIPS PUB 180-2 by NIST
  - **SHA-2 hash functions are believed to be secure**
- SHA-3
  - result of the NIST Cryptographic Hash Algorithm Competition started November 2007
  - 3 selection rounds, 5 finalists
  - result 2nd October 2012: Keccak (pronounced [kɛtʃak], like “ketchak”)

MD5 Hash in the Wild

- United States Cyber Command (USCYBERCOM)
  - mission statement:= "*USCYBERCOM plans, coordinates, integrates, synchronizes and conducts activities to: direct the operations and defense of specified Department of Defense information networks and; prepare to, and when directed, conduct full spectrum military cyberspace operations in order to enable actions in all domains, ensure US/Allied freedom of action in cyberspace and deny the same to our adversaries.*"

MD5 Hash in the Wild

mission statement:= "*USCYBERCOM plans, coordinates, integrates, synchronizes and conducts activities to: direct the operations and defense of specified Department of Defense information networks and; prepare to, and when directed, conduct full spectrum military cyberspace operations in order to enable actions in all domains, ensure US/Allied freedom of action in cyber-space and deny the same to our adversaries.*"

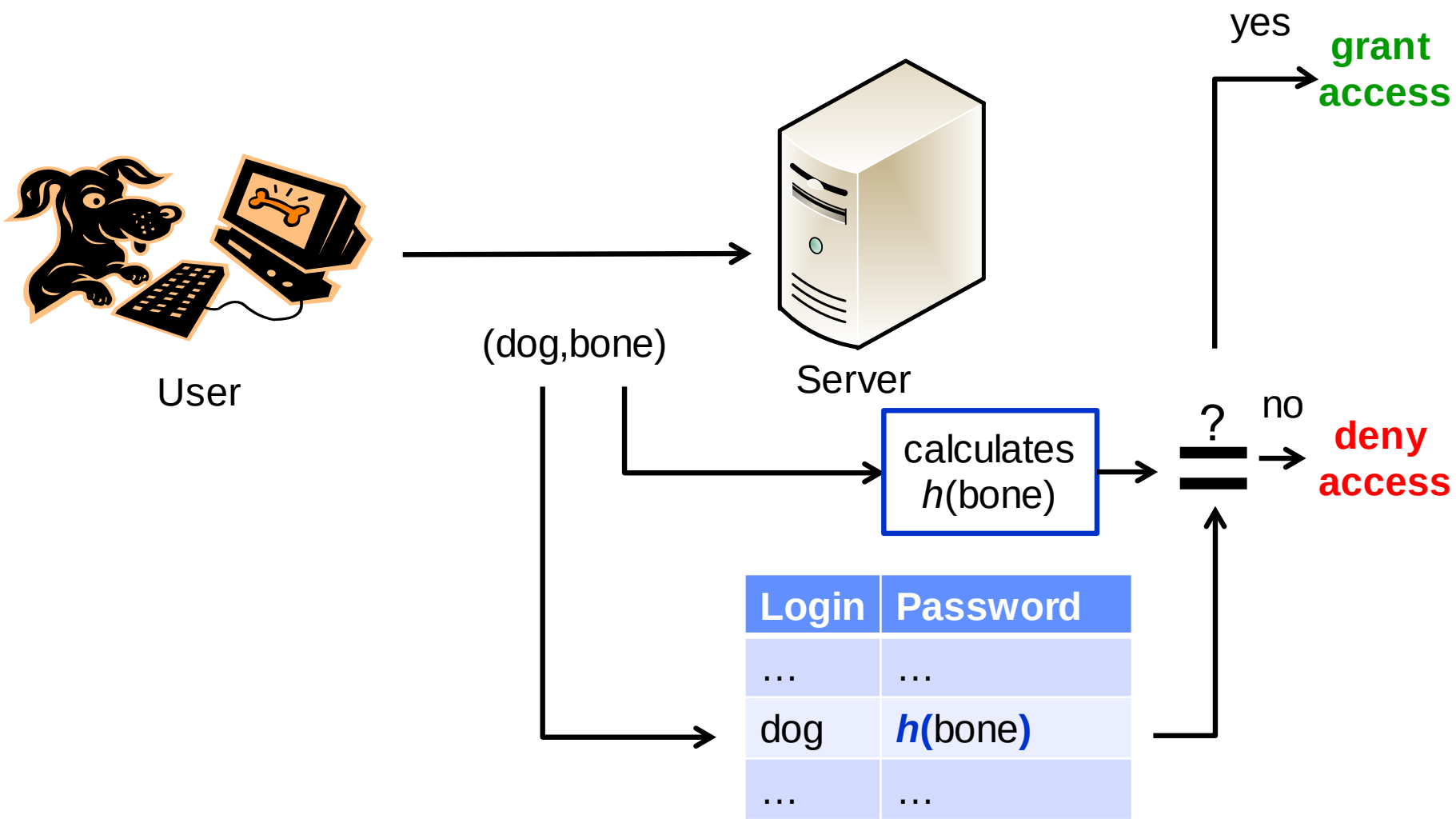
mission statement:= "*USCYBERCOM plans, coordinates, integrates, synchronizes and conducts activities to: direct the operations and defense of specified Department of Defense information networks and; prepare to, and when directed, conduct full spectrum military cyberspace operations in order to enable actions in all domains, ensure US/Allied freedom of action in cyber-space and deny the same to our adversaries.*"

MD5(mission statement)=
9ec4c12949a4f31474f299058ce2b22a

(Remember: MD5 is broken → find other interesting mission statements…)

Password based authentication

- Enhanced approach using one way (hash) functions

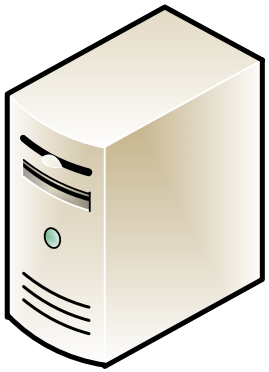


Password based authentication

- Enhanced approach using one way (hash) functions

User

(dog,bone)



Server

yes

grant access

?

no

deny access

calculates *h*(bone)

Login	Password
…	…
dog	*h*(bone)
…	…

Slightly reduced risk, if attacker gets access.

Remaining problems of password based authentication based one way functions

- Brute Force attack
  - function *h*() is public
  - value of *h*(*x*) is known to the attacker

  ➔ try all possible values for *x*

  Considerations:
    - usually >> 1 Mio. *h*(x)/s on ordinary hardware
    - assumption: password uses only small letters
    - password length = 8

Login	Password
…	…
dog	*h*(bone)
…	…

time needed: $\frac{26^8}{1\,000\,000 \cdot 60 \cdot 60} \approx 58\text{h}$

- first countermeasures (against remote attacks):
  - limit false attempts
- first password rules:
  - use a large alphabet (small and capitalised letters, numbers, specials)
  - use a long password

Remaining problems of password based authentication based one way functions

- first password rules:
  - use a large alphabet
    - (small, capitalised letters, numbers, specials)
    - time needed: $\frac{(26+26+10+30)^8}{1\,000\,000 \cdot 60 \cdot 60 \cdot 24 \cdot 365.25} \approx 162a$
  - use a long password

Login	Password
…	…
dog	*h*(bone)
…	…

- remaining possible attacks:
  - increase in computation power
    - distributed approach
    - GPU
    - Moore's law
  - pre-computation:
    - attacker creates lockup table
    - search time (example above):

      $\mathrm{ld}\left((26+26+10+30)^8\right) < 53$ comparisons

Remaining problems of password based authentication based one way functions

- remaining possible attack:
  - pre-computation:
    - attacker creates lockup table
    - search time (example above):
      $\mathrm{ld}\left((26+26+10+30)^8\right)<53$ comparisons
    - problem: storage space
    - Example:
      - size of MD5 hash: 128 bit
      - number of hashes: $(26+26+10+30)^8$=5 132 188 731 375 616
      - storage
        - space: 5 132 188 731 375 616 · 128 bit ≈ 75 000 TByte
        - price: ≈ 2 Mio €
    - solution: storage space / computation tradeoff
      - Martin E. Hellman: “A Cryptanalytic Time – Memory Trade-Off”

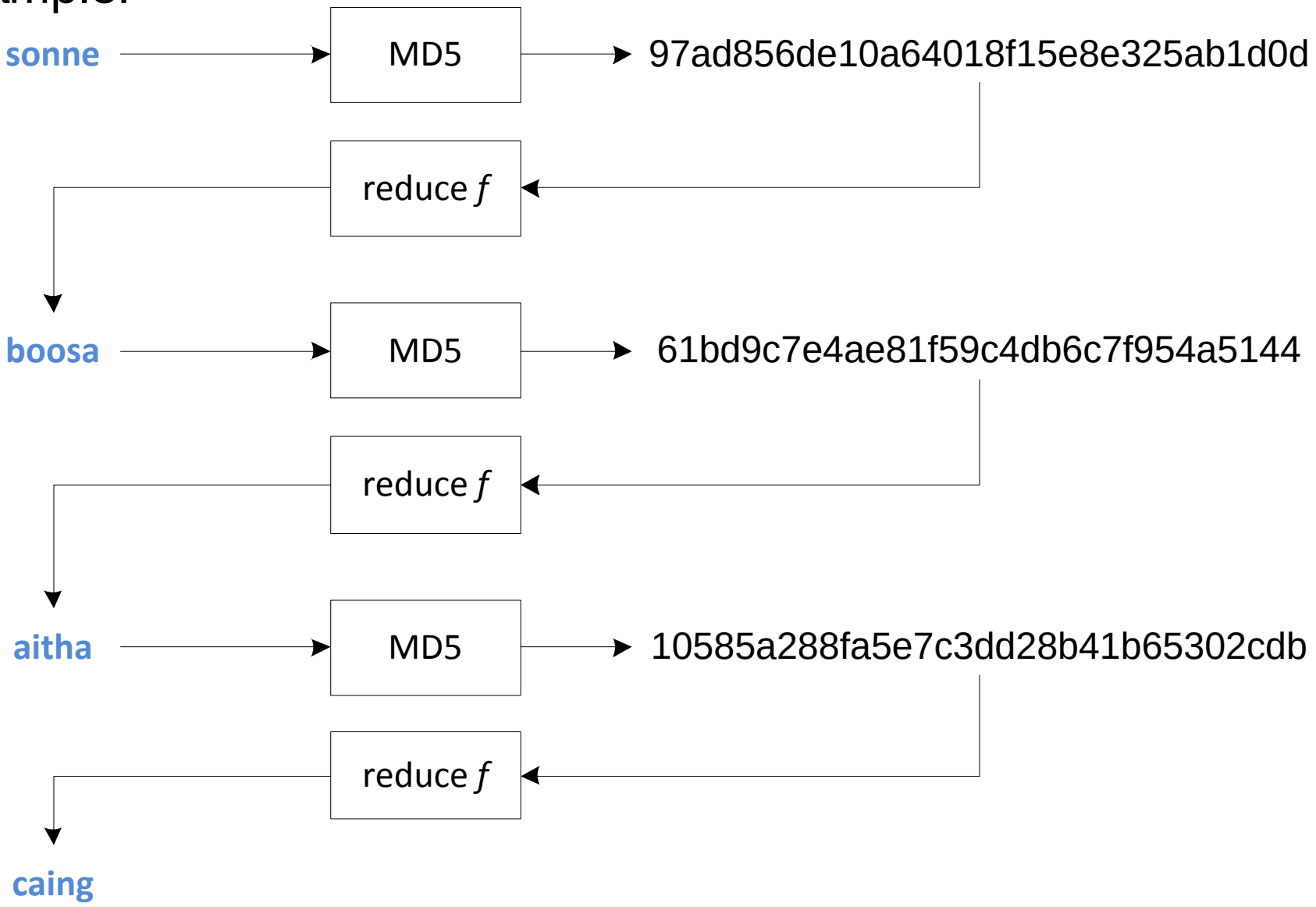
Login	Password
…	…
dog	*h*(bone)
…	…

Cryptanalytic Time – Memory Trade-Off

- main idea:
  - store only certain parts of the lookup table
  - regenerate the missing parts on demand
- requires “reduce” function *f*
  - *f*: H → P (H: set of hash values, P: set of passwords)
  - note: *f* is NOT the inverse of *h*
- general procedure:
  - calculate a chain of hash and reduce function calls
    - p → h() → f() → h() → f() → h () … → f() → p’
  - store first and last value in a table
    - sort by the last value
  - length of chain influences Time – Memory trade-off

Cryptanalytic Time – Memory Trade-Off

- Example:



Cryptanalytic Time – Memory Trade-Off

- 2nd example
  - breaking of PINs
  - *h*(x):= (x·7807) mod 16157
  - *f*(x) := x mod 9000 + 1000

- PIN-table:

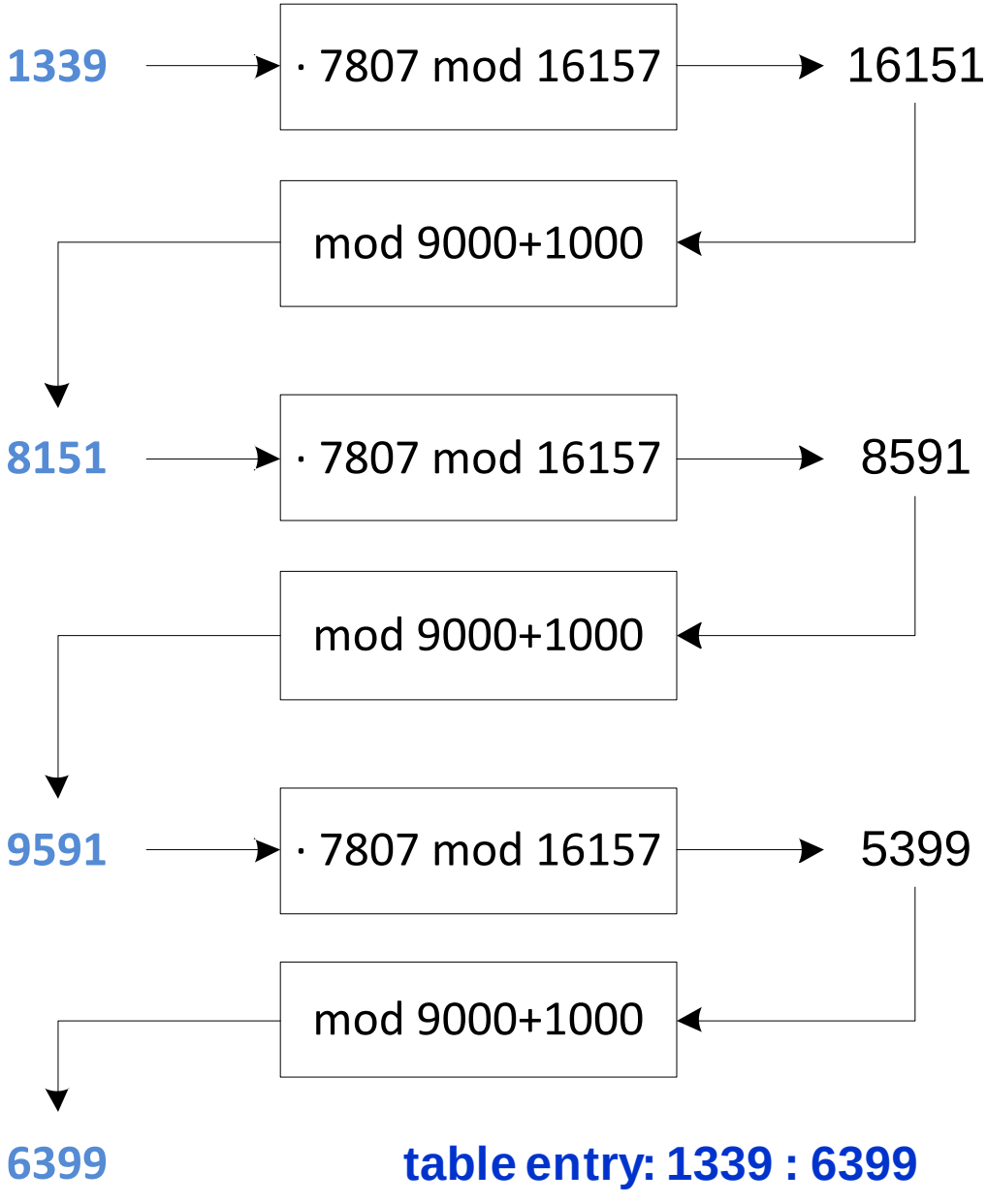
1309–9139–7018–**2139**

2439–9327–4447–**4493**

1084–4677–6676–**5207**

1339–8151–9591–**6399**

3128–8069–6697–**7584**



Cryptanalytic Time – Memory Trade-Off

- PIN-table:

1309–9139–7018–**2139**

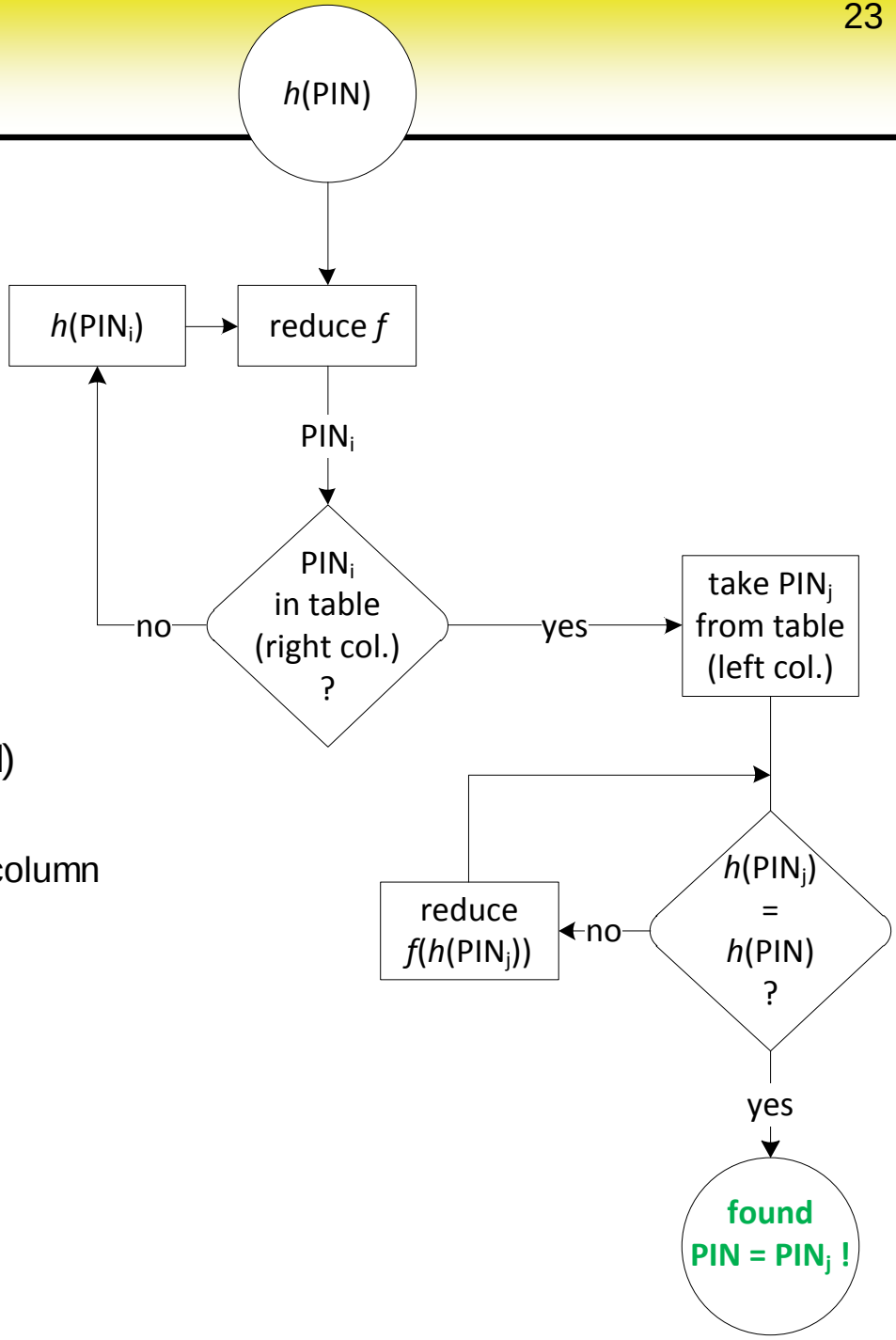
2439–9327–4447–**4493**

1084–4677–6676–**5207**

1339–8151–9591–**6399**

3128–8069–6697–**7584**

- Breaking a PIN:
  - Goal: find PIN for given hash value *h*(PIN)
  - Algorithm:
    - 1. hash / reduce until value is in the right column
    - 2. take left column value
    - 2. hash / reduce until PIN is found



Remaining problems of password based authentication based one way functions

- remaining possible attack:
  - pre-computation:
    - attacker creates lockup table
    - search time (example above):
      $\mathrm{ld}\left((26+26+10+30)^8\right)<53$ comparisons
    - problem: storage space
    - Example:
      - size of MD5 hash: 128 bit
      - number of hashes: $(26+26+10+30)^8$=5 132 188 731 375 616
      - storage
        - space: 5 132 188 731 375 616 · 128 bit ≈ 75 000 TByte
        - price: ≈ 2 Mio €
    - solution: storage space / computation tradeoff
      - Martin E. Hellman: “A Cryptanalytic Time – Memory Trade-Off”
    - today: use the power of the Internet / Cloud
      - http://www.freerainbowtables.com/

Login	Password
…	…
dog	*h*(bone)
…	…

Remaining problems of password based authentication based one way functions

- remaining possible attack:
  - pre-computation
- countermeasure:
  - salt!
  - *h*(x) → *h*(**salt**,x)
  - salt:
    - long (e.g. 128 bit) random value
    - some part is unique for the system (i.e. 104 bit)
    - some part is randomly chosen by the system for each entry in the password table (i.e. 24 bit)
      - NOT stored at the system
    - verification: iterate over all possible salt values

Login	Password
…	…
dog	*h*(bone)
…	…

➔ pre-computation has to be done *for each possible salt*

Remaining problems of password based authentication based one way functions

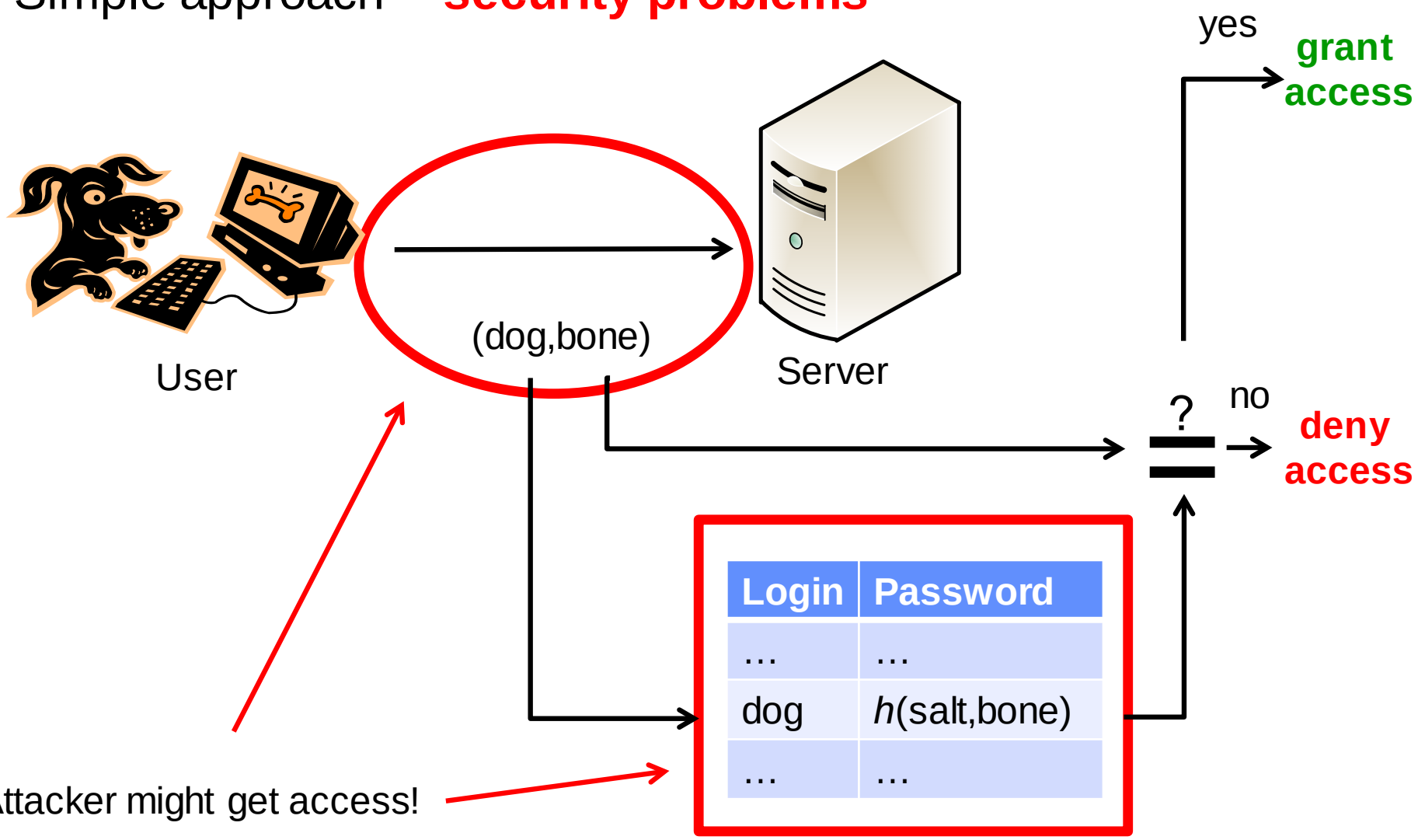
- remaining possible attack:
  - **dictionary attack**
  - problem: people do not chose passwords **randomly**
  - often names, words or predictable numbers are used
  - http://www.whatsmypass.com/the-top-500-worst-passwords-of-all-time
  - attacker uses dictionaries for brute force attack
  - prominent program: *John the Ripper*
    - supports dictionary attacks and password patterns

Login	Password
…	…
dog	*h*(salt,bone)
…	…

- possible solutions:
  - enforce password rules
    - consider usability
  - pre-check passwords (e.g. using John)
  - train people to “generate” good passwords
    - Example: sentence → password
    - “This is the password I use for Google mail” → “TitpIu4Gm”

Password based authentication

- Simple approach – **security problems**



Password based authentication

- Simple approach – **security problems**

User

(dog,bone)

Server

yes

grant access

?

no

deny access

Login	Password
…	…
dog	*h*(salt,bone)
…	…

Might be the attacker!

The Server as Attacker

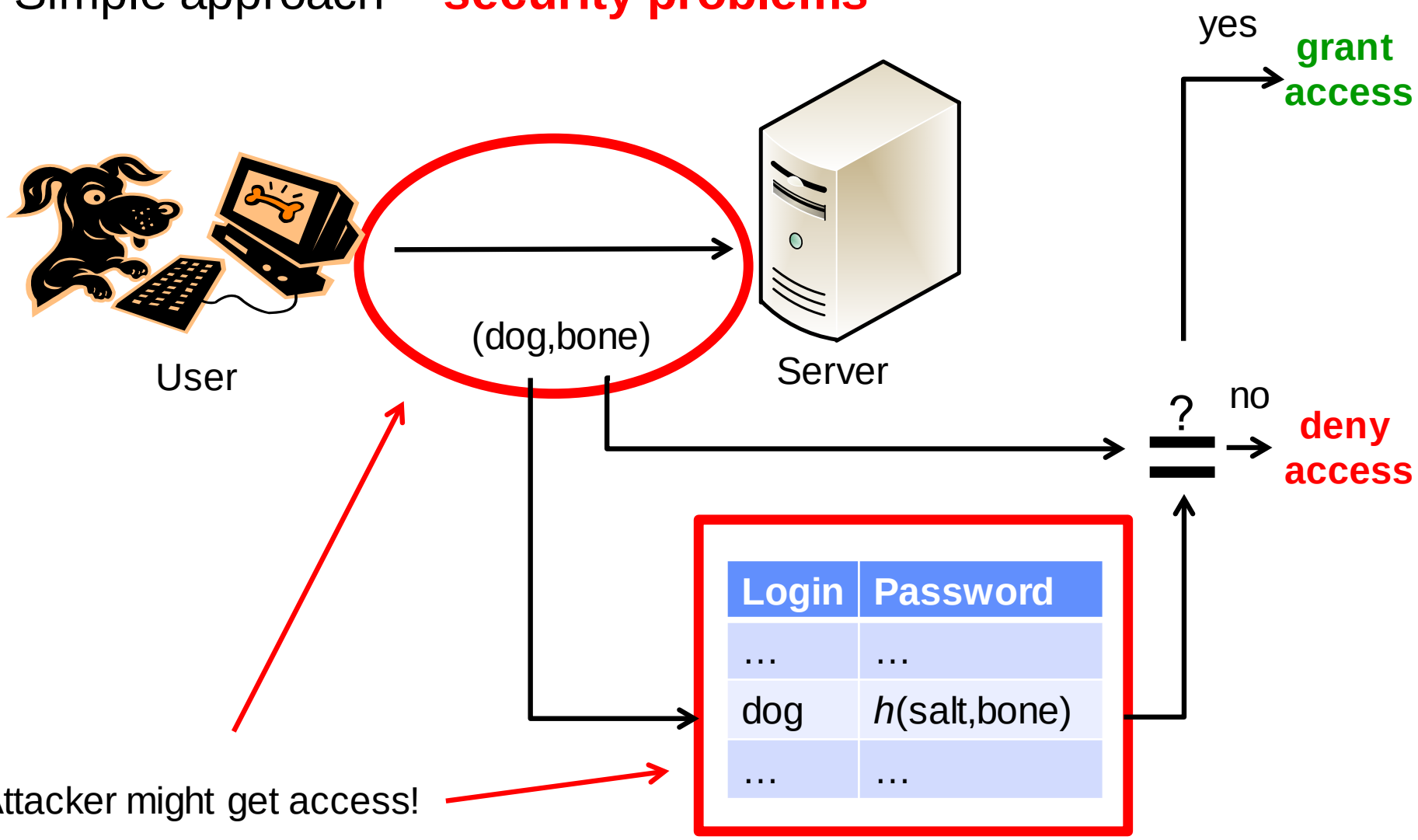
- … a new Web 2.0 service
- … for people which like city journeys
- … find cool cities and places like shops, restaurants, hotels etc.
- ... information from globe-trotters for globe-trotters
- … they can share their knowledge after secure login
- So that's wrong?

➔ It collects (username,password) and tries to login into other popular services like Gmail, Twitter, eBay, Amazon etc.

password rule: never "reuse" passwords!

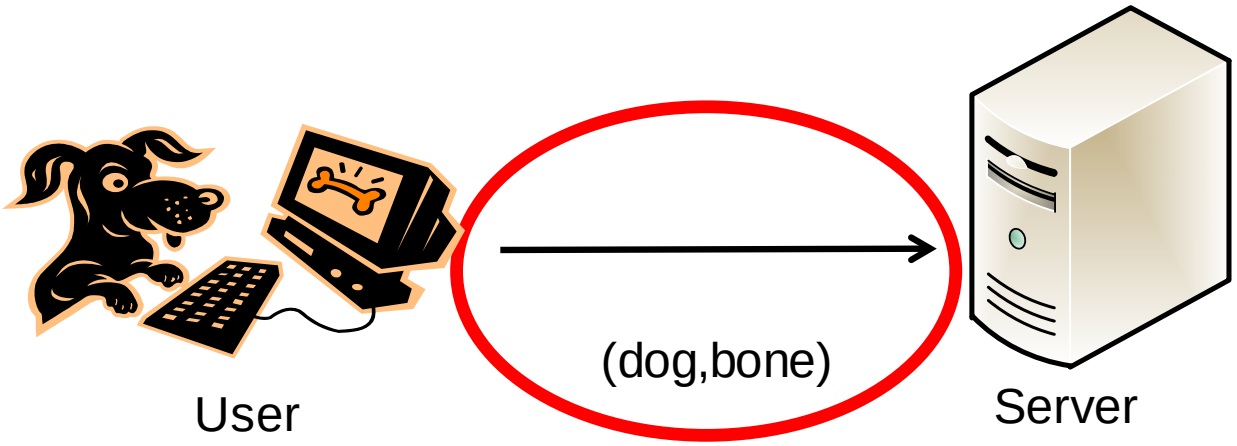
Password based authentication

- Simple approach – **security problems**



Password based authentication

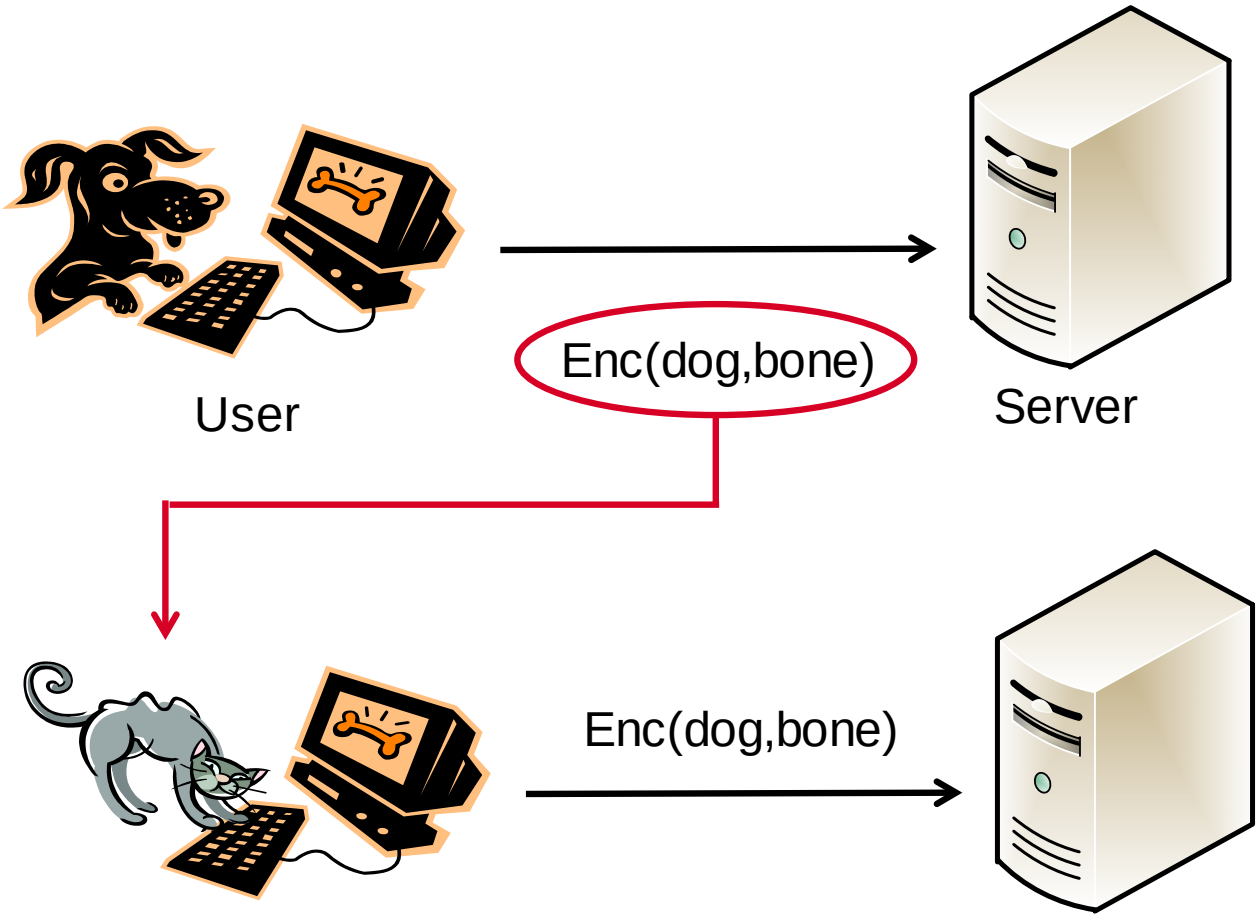
- **security problems**



- possible solution:
  - encrypt communication
- remaining problems:
  - not always possible
  - replay attack

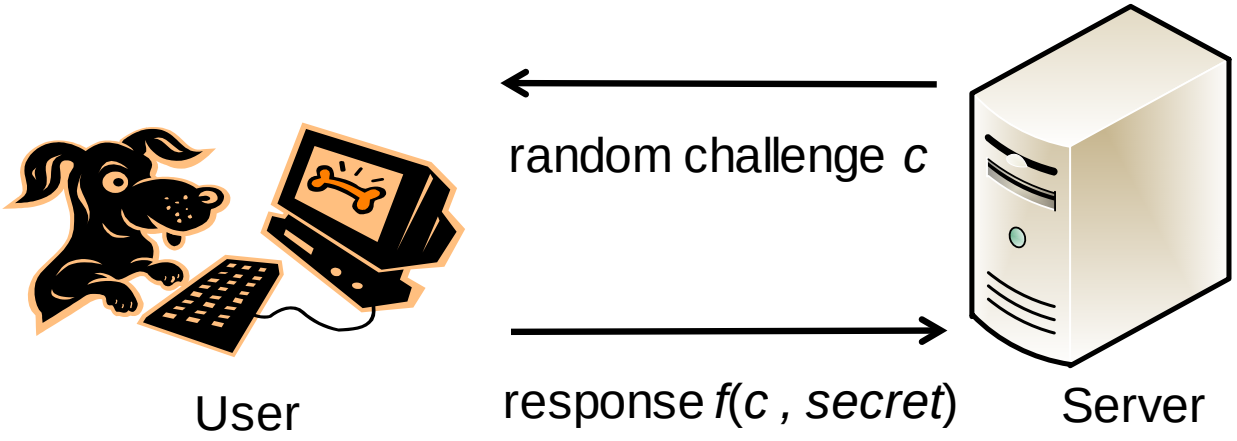
Password based authentication

- **security problem – replay attack**



Password based authentication

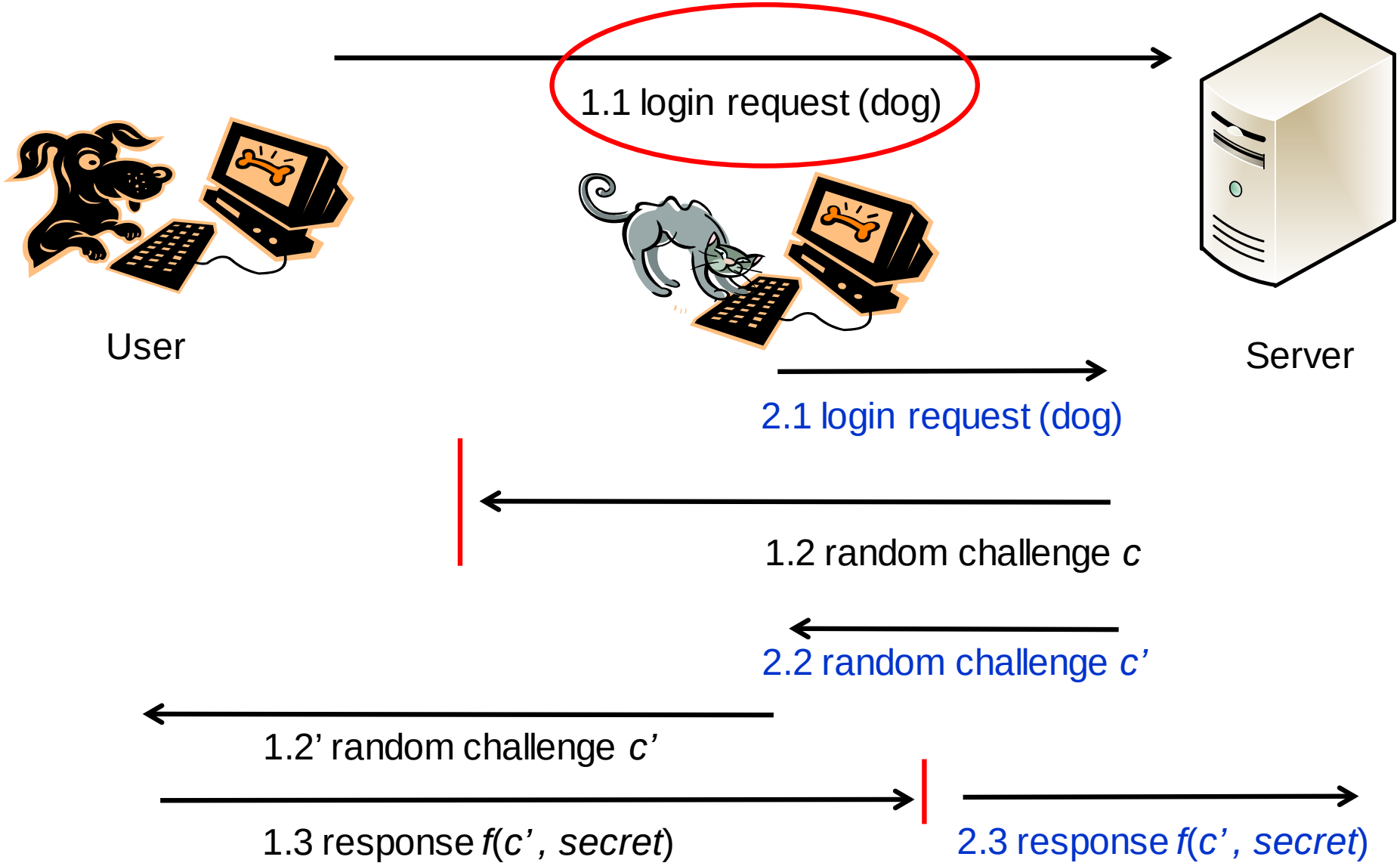
- **security problem – replay attack**
- **possible solution: challenge-response protocol**



  - tries to ensure *freshness*
- remaining problems:
  - Man-in-the-middle attacks
  - parallel protocol runs

Password based authentication

- **security problem – MITM / parallel protocol runs**



Password based authentication

- **security problem – MITM / parallel protocol runs**
- **possible solutions:**
  - disallow parallel login protocol runs for the same user
  - make protocol runs distinguishable

Password based authentication

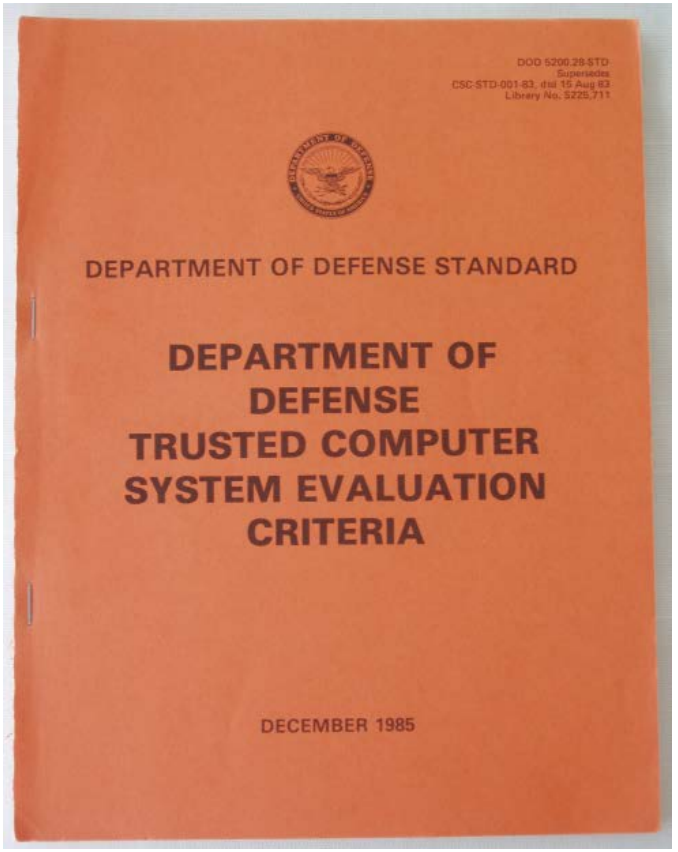
- **security problem – MITM / parallel protocol runs**
- **possible solutions:**
  - disallow parallel login protocol runs for the same user
  - make protocol runs distinguishable
- remaining security problems:
  - …

  (ok I will stop here – if you are interested in many more problems / solutions I recommend: Colin Boyd, Anish Mathuria: “Protocols for Authentication and Key Establishment”, Springer, 2003.)

Password based authentication

- **(non protocol related) security problems:**
  - phising, i.e. faked UI for entering secret information
  - today: mostly Internet based attacks
  - but: local attacks possible as well
    - faked login / lock screen
    - solution: “trusted path” / Secure Attention Key

      3.2.2.1.1 The TCB [Trusted Computing Base] shall support a trusted communication path between itself and user for initial login and authentication. Communications via this path shall be initiated exclusively by a user.
      [Department of Defense: “Trusted Computer System Evaluation Criteria”, CSC-STD-001-83, 15. August 1983 – called “Orange Book”]
    - well known implementations:
      - Windows: Ctrl+Alt+Del
      - Linux: Ctrl+Alt+Pause
        - could be freely chosen in principle



[http://en.wikipedia.org/wiki/File:Orange-book-small.PNG]