CS 556 – Computer Security
Fall 2012

Dr. Indrajit Ray
Email: indrajit@cs.colostate.edu

Department of Computer Science
Colorado State University
Fort Collins, CO 80523, USA
PASSWORD BASED AUTHENTICATION

PASSWORD AUTHENTICATION MODELS

UNIX PASSWORD AUTHENTICATION

PASSWORD MANAGEMENT

PASSWORD BASED AUTHENTICATION
Passwords

- Commonly used method
- For each user, system stores in a password file
  $$< \text{User name}, F(\text{password}) >,$$
  where $F$ is some transformation
  ✦ $F(\text{password})$ is easy to compute
  ✦ From $F(\text{password})$, password is difficult (ideally) to compute
- When a user enters the password, system computes $F(\text{password})$; A match provides proof of identity
Simple Password Scheme

CLAIMANT TERMINAL

p'  \rightarrow  f

id

VERIFIER

id  q

Message

compare

Authenticaton OK or not

q'  \quad id
Choice of Passwords

- Suppose passwords can be from 1 to 8 characters in length
  - Lower case English alphabets used
- Possible choices for passwords $= 26^1 + 26^2 + \ldots + 26^8 = 1.5 \times 10^{12}$
- At the rate of 1 password per millisecond, it will take about 150 years to test all passwords

We don’t need to try all possible passwords – only many probable ones 😞
Probable Passwords

- In a Bell Labs study (Morris & Thompson 1979), 3,289 passwords were examined
  - 15 single ASCII characters
  - 72 two ASCII characters
  - 464 three ASCII characters
  - 477 four alphanumeric character
  - 706 five letters (all lower or all upper case)
  - 605 six letters all lower case
  - 492 weak passwords (dictionary words spelled backwards, first names, last names etc)

- Summary: 2831 passwords (86%) were weak, that is they were too easily predictable, or were too short
Dictionary Attacks on Passwords

- The statistics haven’t changed much in later studies
- To improve upon the expected probability of success of an exhaustive search, an attacker may search the space of all possible passwords in order of decreasing probability
- Note that these attacks work only with strong passwords
Dictionary Attacks on Passwords

- The statistics haven’t changed much in later studies.
- To improve upon the expected probability of success of an exhaustive search, an attacker may search the space of all possible passwords in order of decreasing probability.
Dictionary Attack on Simple Password Scheme

- Attacker constructs a table of values of $q$ to values of $p$ ensuring, especially, that the table contains the most likely expected values for $p$
- Passively monitor large numbers of authentication attempts
  - Can obtain with high degree of probability some passwords for some user
Dictionary Attack (Case 1)

- Create a dictionary of common words and names and their simple transformation
  - For example: indrajit → jndaitr
- Use these to guess the password
Dictionary Attack (Case 2 – known $\mathcal{F}$)
PASSWORD AUTHENTICATION MODELS
Improved Password Scheme (#2)

CLAIMANT TERMINAL

p'

f

id

Message

q'  id

VERIFIER

id  q

compare

Authentication OK or not
Now we have stored password $q = \mathcal{F}(p, id)$

It is still possible to prepare the table of q values for password values but now only for one particular id.

Suffers from verifier compromise

- attacker gets $<id, q>$ pair and attacker generating an authentication message request on the wire.
Improved Scheme #3

CLAIMANT TERMINAL

p' id

Message

p' id

VERIFIER

f q'

compare

Authentication OK or not

Dr. Indrajit Ray, Computer Science Department
UNIX PASSWORD AUTHENTICATION
Scheme #3 ≡ UNIX Password

- Designed by Bob Morris and Ken Thomson to encrypt passwords in Unix
- Uses an extended DES algorithm for the password function $F$
- User password and salt is used as encryption key to encrypt
- Process is repeated 25 times
  - $i^{th}$ encrypted block used as the plaintext for the $(i + 1)^{th}$ round
Unix Crypt Algorithm

user password

truncate to 8 ASCII characters pad with 0s if necessary

Modified DES

56 bit key

data

next input Ii 2le i

Output Oi

12 bit user salt initially taken from the system clock

repack 76 bits into eleven 7-bit characters

encrypted password
Password Salt

- Salt is used to make dictionary attack a bit more difficult.
- Salt is a 12 bit number between 0 and 4095.
- It is derived from the system clock and the process identifier.
- Rather than computing $f(password)$, compute $f(password + salt)$; both salt and $f(password + salt)$ is stored in the table.
- With salt the same password can result in 4096 different values stored in the table.
Attack on Scheme #3

- Password on the clear in the network
  - Vulnerable to eavesdropping
    - Serious concern on any network
  - Also susceptible to replay attacks
    - Attacker eavesdrops on the communication channel and intercepts a legitimate authentication exchange
    - Attacker later replays the authentication exchange
    - Serious on interconnected networks
Improved Password Scheme #4

**Claimant Terminal**

- `id`
- `p'`
- `f`
- `q'`
- `id`

**Verifier**

- `id`
- `r`
- `r'`
- `g`
- `compare`

Authentication OK or not
PASSWORD MANAGEMENT
Policy and Procedure

- Educate users to make better choices
  - Does not work if user population is large or novice
- Define rules for good password selection and ask users to follow them
  - Rules serve as guidelines for attackers
- Ask or force users to change their passwords periodically
Policy and Procedure

- Force users to use machine generated passwords
  - Random passwords are difficult to memorize
  - Password generator may become known to the attacker through analysis
- Actively attempt to break user’s passwords; force users to change those that are broken
- Screen password choices; if a choice is weak force user to make a different choice