The Tenex password scheme

- **Directories can have passwords**
  - Stored in cleartext by the system
- `connect` **system call gets access to directories**
  - Passes in pointer to string
  - OS performs the following check:

```
for (i = 0; i <= strlen (directoryPassword); i++)
    if (directoryPassword[i] != passwordArgument) {
        sleep (3);
        return EACCES;
    }
return 0;
```

Weaknesses

- **Bad: Vulnerability of keeping plaintext passwords**
  - By compromising the system, can all passwords
- **Worse: Can guess password in linear time!**
  - Place first character of password as last byte on VM page
  - Make sure the next page is unmapped
  - Try all first letters, one by one
  - Got a page fault? Must have correct first letter
  - Repeat for second character of password, etc.

Unix hashed passwords

- **Store one-way function of password**
  - Only hashes first 8 characters
  - Encrypt 0s 25 times with password as key
  - Key generally hard to recover from ciphertext
  - So put hashed password in world-readable `/etc/passwd`
  - To validate password, hash it and compare to stored hash
- **Hash function “salted” with 12 extra bits**
  - Prevent attacker from building dictionary of hashes of common passwords
  - Permute the hash function based on 12-bit seed
  - Prepend seed to hashed password for use in verification

Weaknesses

- **Off-line password guessing attacks**
  - Attacker gets password file, guesses passwords at home
  - Computationally expensive, but undetectable by victims
- **Fails to account for hardware & software improvements**
  - DES crypt on 1977 VAX-11/780: 4 crypts/sec
  - Bitsliced implementation on 600 MHz alpha: 214,000 cr/sec
  - But users don't chose better passwords now than in 1977!
- **Server still gets plaintext password at login**
  - Attacker or server operator can modify login record it
  - But users often use same password on different systems

Design hints for systems with passwords

- **Any algorithm that uses passwords should:**
  - Take a salt as input (prevent hash dictionaries)
  - Take a cost parameter as input
    Administrators need to increase hashing cost as hardware gets faster!
- **Bad examples: PGP & ssh private keys**
  - Stored in user's home directory, encrypted with password
  - Given encrypted private key, can guess password off-line
  - Cost of guess even cheaper than password crypt!
- **Good example: OpenBSD/FreeBSD bcrypt**
  - Hashing password is exponential in cost parameter
  - Cost goes in hashed password along with salt

s/key password authentication

- **Goal: Protect against passive eavesdroppers**
  - Also: Minimize harm of bad clients (e.g., public terminals)
- **Idea: One time passwords, not valid after snooped**
- **Algorithm takes user's real password, p and salt s**
  - First one time password is $H^{(100)}(p, s)$ (for one way hash H)
  - Next password is $H^{(99)}(p, s)$, etc.
  - After 99 logins, must change salt or password
- **Benefit: Very convenient**
  - Carry list of one-time passwords, calculate on palm pilot

Weaknesses

- **Possible race conditions**
  - Attacker sends password before you press return
- **Vulnerable to off-line password guessing**
  - Attacker sees s and $H^{(n)}(p, s)$
  - Can verify guesses of p off-line
  - H not very expensive (should be $H^{(n)}(G(p, s))$ for expensive G—or maybe users should crank n)
- **Bad client can compromise session**
  - Before logout, can insert command to create back door:
    `echo 1024 35 145...3 badguy >> .ssh/authorized_keys`

Password-derived public keys

- **Derive public key from user's password**
  - E.g., use password as seed for pseudo-random generator
  - Client can regenerate private key given password
- **Server stores public key for each user**
- **Use public key authentication, E.g.:**
  - $S \to C : N_S$
  - $C \to S : K_u, \{C, S, u, \text{session}, N_S\}_{K_u^{-1}}$
  - Client checks sig, looks up pubkey K_u for credentials

Weaknesses

- **No salt**
  - Users with same password will have same public key
- **No cost parameter**
  - Can’t take too long to log in
  - But over time generating key will get faster
- **Public key is just like password hash**
  - Eavesdropper will see key, can mount off-line attack

Secure password protocols

- **Purpose of secure password protocol:**
  - Generate strong shared secret from user-chosen password
  - Use secret as key for mutually authenticated session
- **Goals of the SRP protocol**
  - Network attacker should be unable learn anything that will allow an off-line guessing attack
  - Forward secrecy
  - Server knows nothing plaintext-equivalent
- **Warning: SRP has not been proven correct**
  - No known bugs in SRP, but provably correct protocols exist
  - What follows is an *informal* argument about why SRP might be secure

Background: Diffie-Hellman key exchange

- **An unauthenticated key exchange protocol**
  - Provides forward secrecy against a passive adversary
- **The discrete log problem:**
  - Let p be a prime and g a generator of $\mathbf{Z}_p^*$
  - Given $x \in \mathbf{Z}_{p-1}$, easy to compute g^x from x, inverse hard
- **Diffie-Hellman protocol, given g and p**
  - A picks random x, sends B g^x
  - B picks random y, sends A g^y
  - A and B use $H(g^{xy})$ as session key
- **Note: Breaking DH may be easier than discrete log**

SRP protocol

- **User knows** p**, Server knows** $s, g^{H(s,p)}$
- **User picks** a**.** $U \rightarrow S : U, g^a$
- **Server picks** b, u**.** $S \rightarrow U : s, g^{H(s,p)} + g^b, u$
- **User computes:** $K = H\left(\left(g^b\right)^{(a+uH(s,p))}\right)$
- **Server computes:** $K = H\left(g^a \left(g^{H(s,p)}\right)^u\right)^b$
- **In either case:** $K = g^{ab} g^{buH(s,p)}$

Informal analysis

- **No obvious way for U to get K without p**
- S **doesn't know** p
  - Even if S's secret $g^{H(s,p)}$ stolen, thief can't impersonate user without mounting off-line guessing attack
- **No obvious way to mount off-line guessing attack**
  - Suppose attacker impersonates user, then learns K
    Will know a, s, u, $g^{H(s,p)} + g^b$, and $H(g^{ab} g^{buH(s,p)})$
    No obvious way to validate guess of p
  - Suppose attacker impersonates server and learns K
    Will know s, B, g^a, u, and $H\left(\left(B - g^{H(s,p)}\right)^{a+uH(s,p)}\right)$
    No obvious way to validate guess of p

Secure password protocols in practice

- **Passwords are sufficient for mutual authentication**
  - A distributed system in which a user types a password should never be susceptible to a man in the middle attack!
  - If user establishes a password in person, no need for PKI to contact server
- **Consider adding cost parameter c to algorithms**
  - For example hash password 2^c times
  - $U \rightarrow S : U, g^a$
  - $S \rightarrow U : s, c, g^{H^{(2^c)}(s,p)} + g^b, u$

SSL/TLS Overview

- **SSL offers security for HTTP protocol**
  - See protection handout for overview
- **Authentication of server to client**
- **Optional authentication of client to server**
  - Incompatibly implemented in different browsers
  - CA infrastructure not in widespread use
- **Confidentiality of communications**
- **Integrity protection of communications**

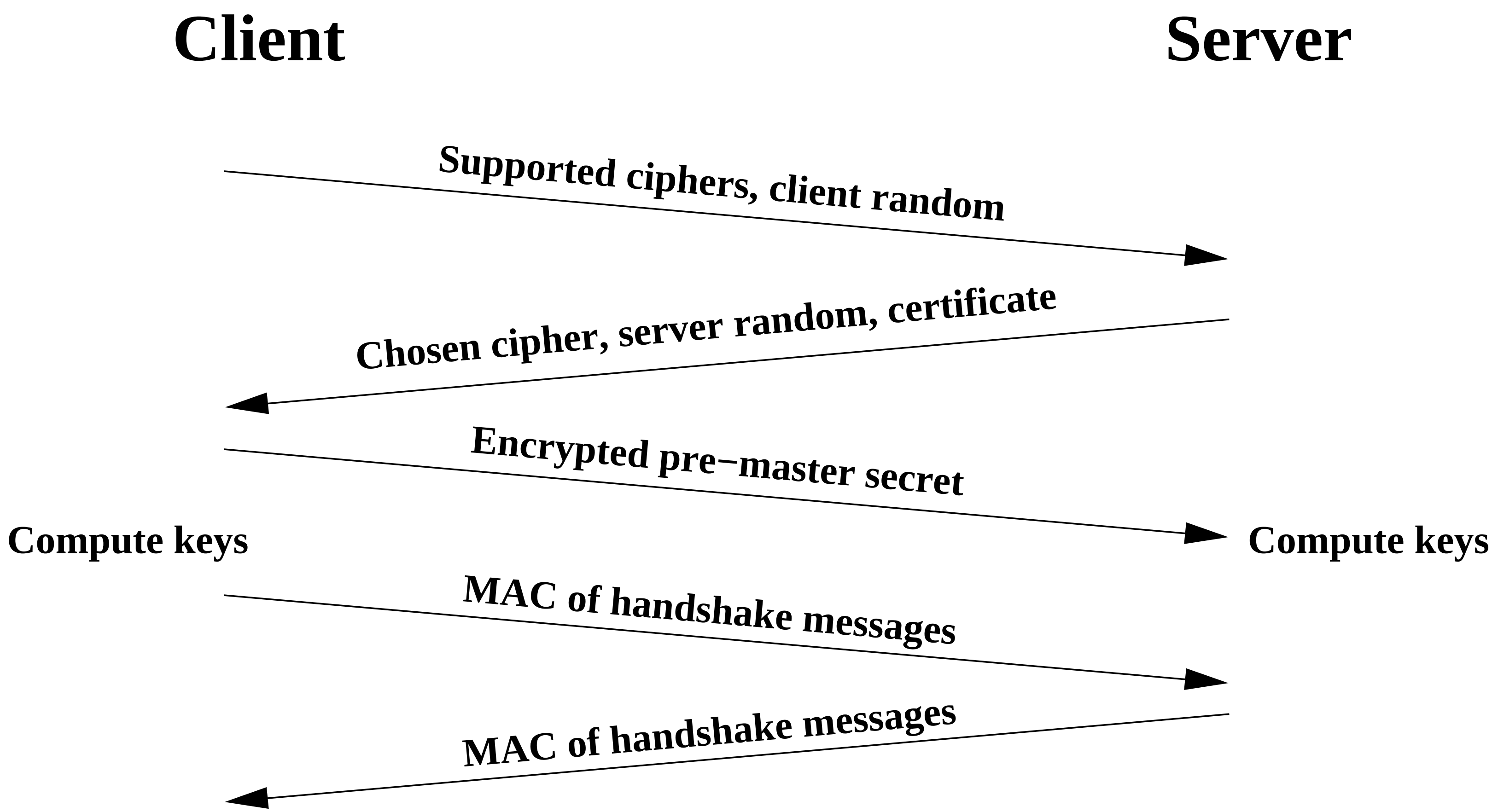
Purpose in more detail

- **Authentication based on certification authorities (CAs)**
  - Trusted third party with well-known public key
  - Certifies who belongs to a public key (domain name and real name of company)
  - Example: Verisign
- **What SSL Does Not Address**
  - Privacy
  - Traffic analysis
  - Trust management

Ciphersuites: Negotiating ciphers

- Server authentication algorithm (RSA, DSS)
- Key exchange algorithm (RSA, DHE)
- Symmetric cipher for confidentiality (RC4, DES)
- MAC (HMAC-MD5, HMAC-SHA)

Overview of SSL Handshake



From “SSL and TLS” by Eric Rescorla

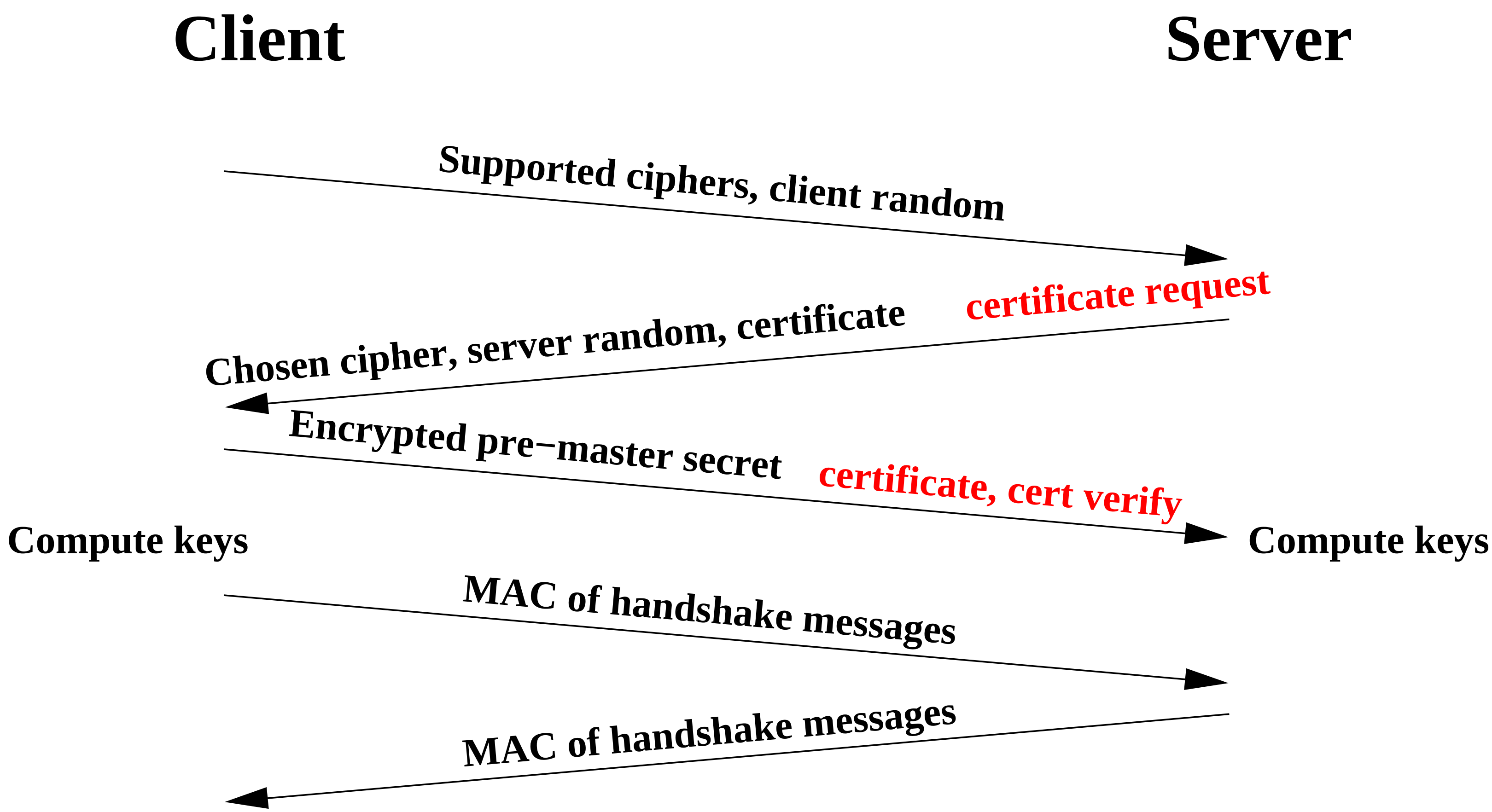
Simplified SSL Handshake

- Client and server negotiate on cipher selection.
- Cooperatively establish session keys.
- Use session keys for secure communication.

Client Authentication Handshake

- Server requests that client send its certificate.
- Client signs a signed digest of the handshake messages.

SSL Client Certificate

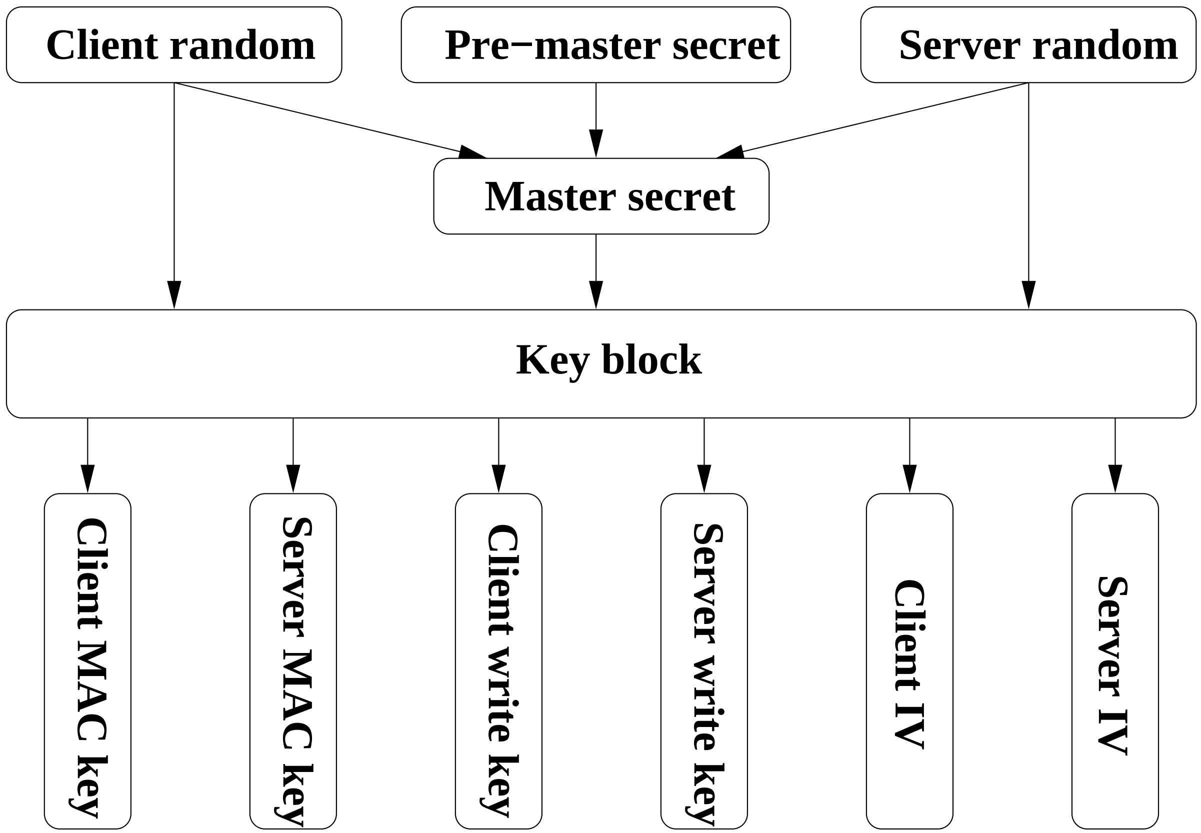


From “SSL and TLS” by Eric Rescorla

Establishing a Session Key

- Server and client both contribute randomness.
- Client sends server a “pre-master secret” encrypted with server’s public key.
- Use randomness and pre-master secret to create session keys:
  - Client MAC
  - Server MAC
  - Client Write
  - Server Write
  - Client IV
  - Server IV

Establishing a Session Key



From "SSL and TLS" by Eric Rescorla

Session Resumption

- **Problem: Public key crypto expensive**
- **New TCP connection, reuse master secret.**
  - Avoids unnecessary public key cryptography.
- **Combines cached master secret with new randomness to generate new session keys.**
- **Works even when the client IP changes (servers cache on session ID, clients cache on server hostname).**

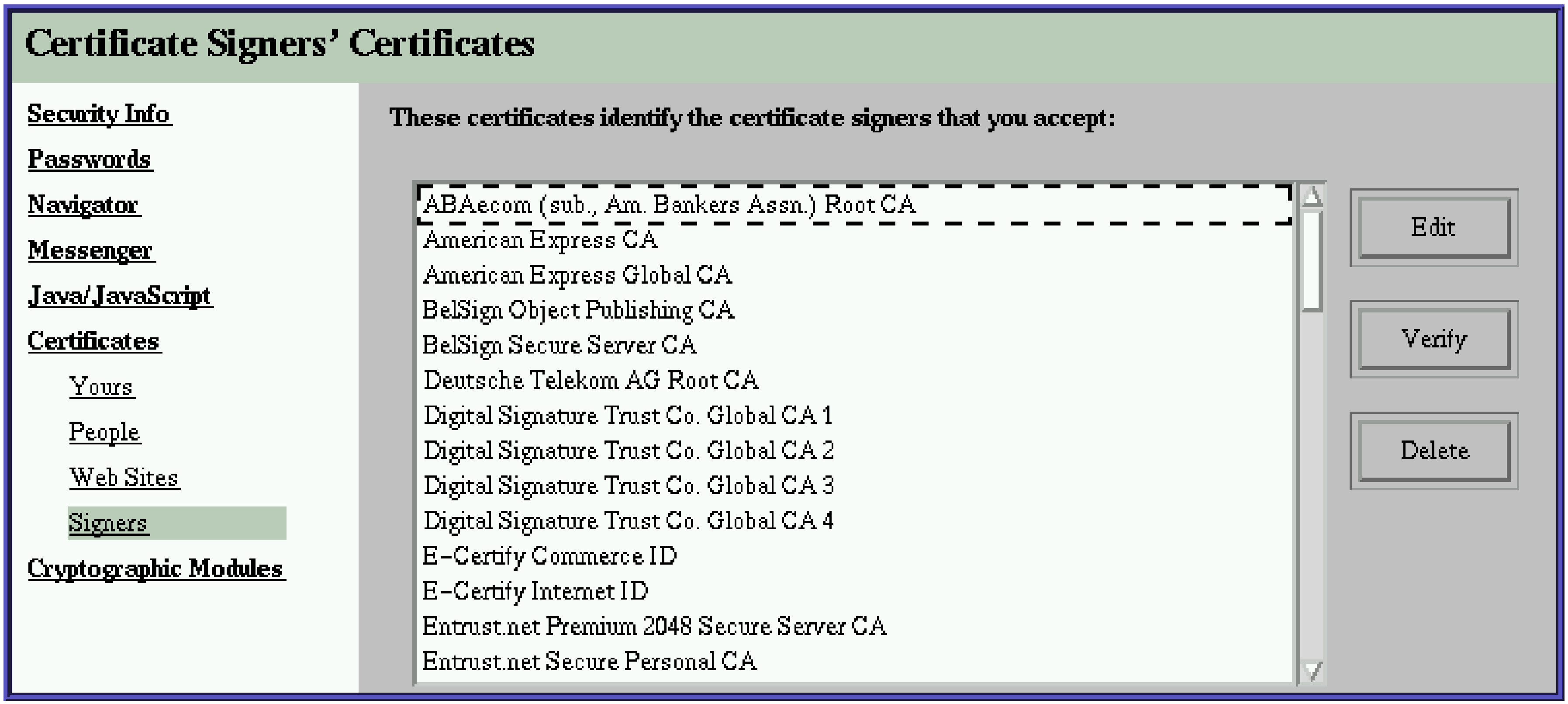
What does a CA-issued Certificate Mean?

- No one knows exactly.
- That a public key belongs to someone authorized to represent a hostname?
- That a public key belongs to someone who is associated in some way with a hostname?
- That a public key belongs to someone who has lots of paper trails associated to a company related to a hostname?
- That the CA has no liability?

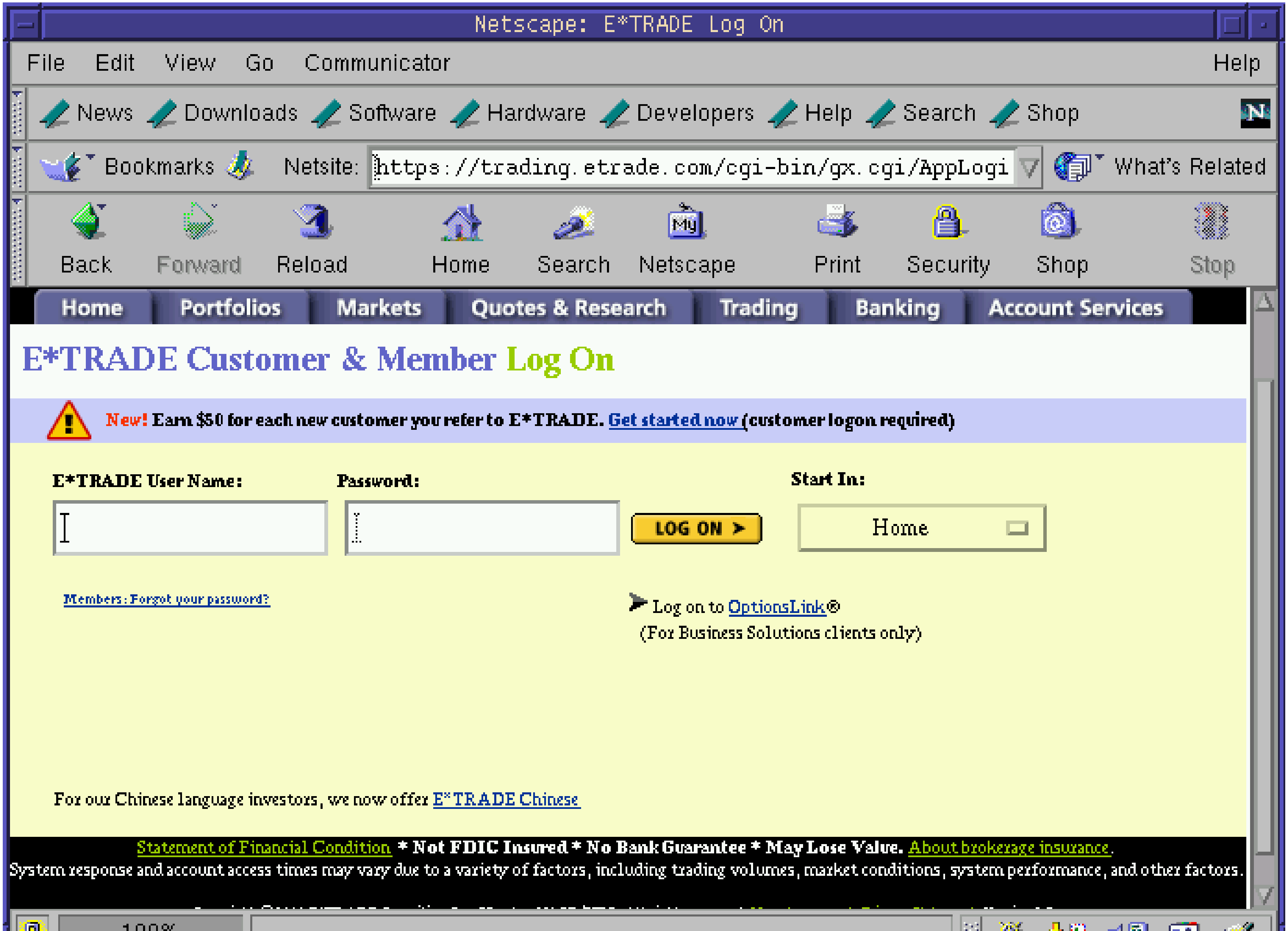
How to get a Verisign certificate

- **Pay Verisign ($300)**
- **Get DBA license from city call ($20)**
  - No on-line check for name conflicts…can I do business as Microsoft?
- **Letterhead from company ($0)**
- **Notarized document (need driver's license) ($0)**
- **Conclusions:**
  - Easy to get a fraudulent certificate
  - Maybe not so easy to avoid prosecution afterwards
- **But that's only Verisign's policy**
  - Many CAs can issue certificates

So many CAs...


Certificate Signers' Certificates
Security Info
Passwords
Navigator
Messenger
Java/JavaScript
Certificates
Yours
People
Web Sites
Signers
Cryptographic Modules
These certificates identify the certificate signers that you accept:
ABAecom (sub., Am. Bankers Assn.) Root CA
American Express CA
American Express Global CA
BelSign Object Publishing CA
BelSign Secure Server CA
Deutsche Telekom AG Root CA
Digital Signature Trust Co. Global CA 1
Digital Signature Trust Co. Global CA 2
Digital Signature Trust Co. Global CA 3
Digital Signature Trust Co. Global CA 4
E-Certify Commerce ID
E-Certify Internet ID
Entrust.net Premium 2048 Secure Server CA
Entrust.net Secure Personal CA
Edit
Verify
Delete

Client Authentication on the Web



Interrogative adversaries

- **Adaptively query a Web server a reasonable number of times**
- **Treat server as an oracle for an adaptive chosen message attack**
- **Don't need any eavesdropping or other network tampering**
- **Anyone can do it, but surprisingly powerful attack**
  - C.f., adaptive chosen-ciphertext attacks—sounded improbable

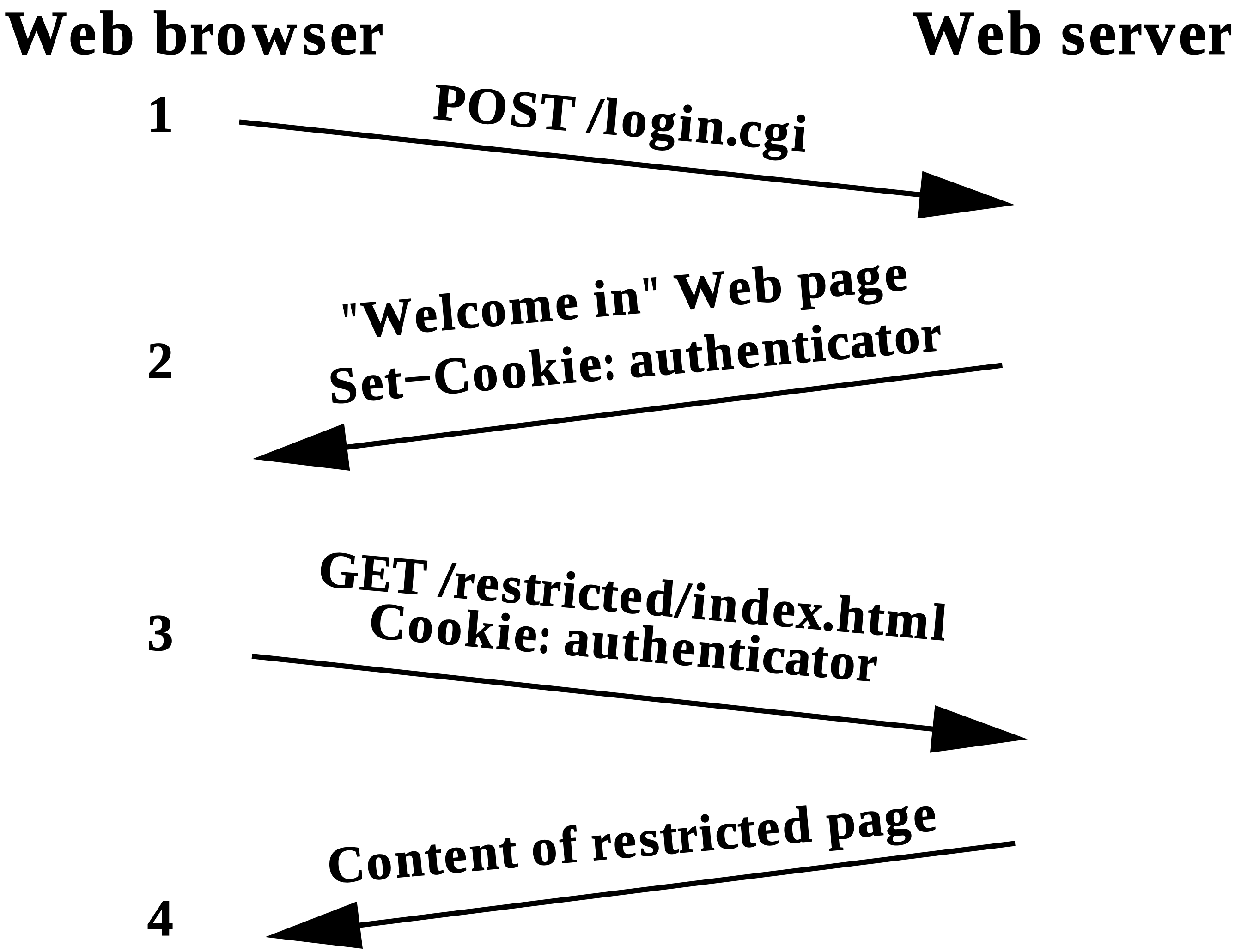
Cookies

- A Web server can store key/value pairs on a client
- The browser resends cookies in subsequent requests to the server
- Cookies can implement login sessions

Netscape cookie example

domain	.wsj.com
Path	/cgi
SSL?	FALSE
Expiration	941452067
Variable name	fastlogin
Value	bitdiddleMaRdw2J1h6Lfc

Cookies for login sessions



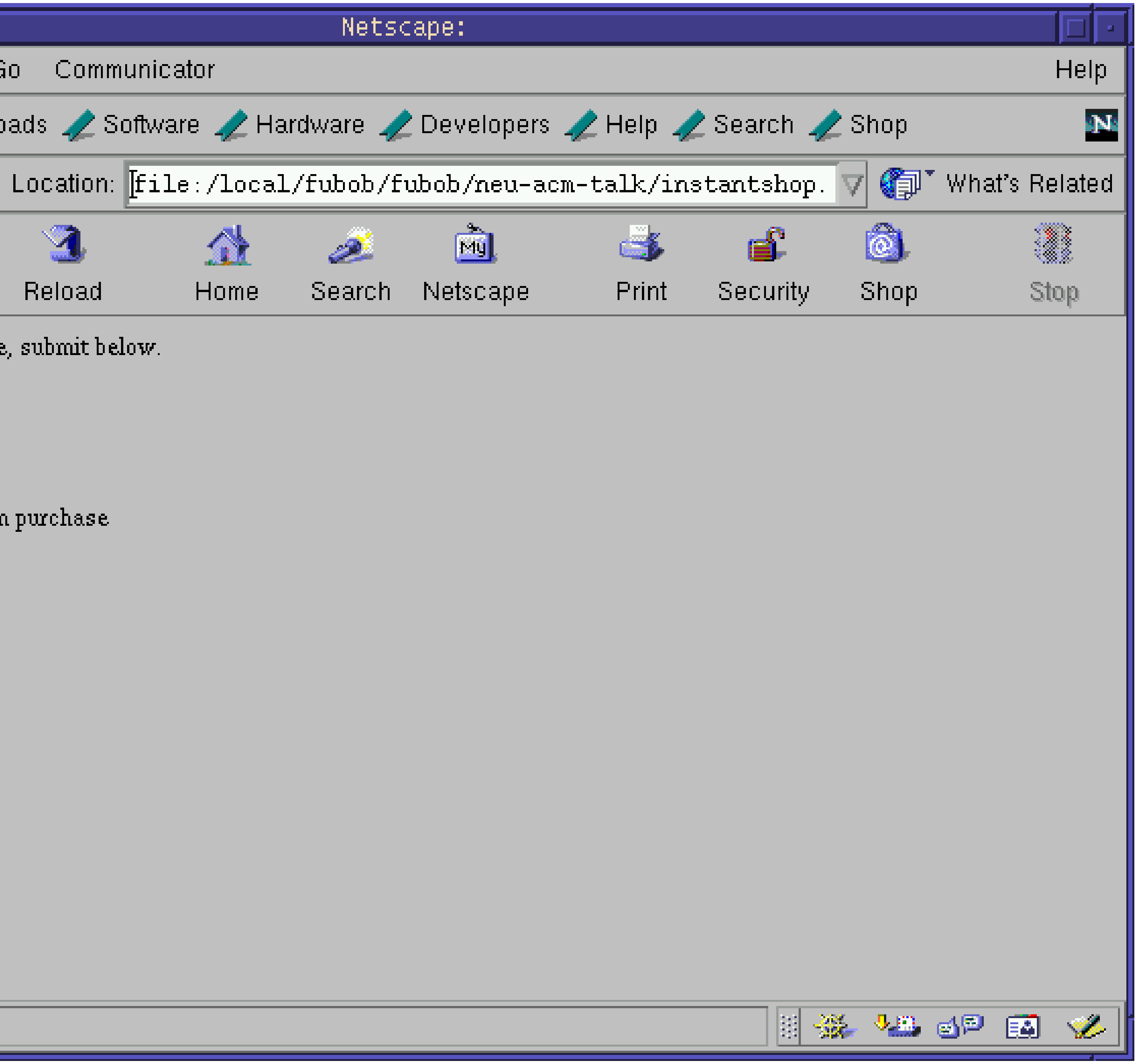
Why? Enter a password once per session

SSL can't protect data sent without SSL

- **Problem: Secure content can leak through plaintext channels**
- **Cookie file has flag to require SSL**
  - Not set by BankOnline.com
- **Trick user into visiting HTTP port**
  - Just need a link from an unrelated web page
  - Cookie automatically sent in the clear
  - Network eavesdropper can record it
  - Might as well not have used SSL

Letting clients name the price: Instant Shop

- Problem: Servers trust clients not to modify HTML variables.
- Price determined by hidden variable in Web page.
- Make a personal copy of the web page. Modify it.


Netscape:
Go Communicator Help
oads Software Hardware Developers Help Search Shop
Location: file:/local/fubob/fubob/neu-acm-talk/instantshop.
What's Related
Reload Home Search Netscape Print Security Shop Stop
e, submit below.
n purchase

Instant Shop example: What's inside

```
<html><body>
 <form action=commit_sale.cgi>

<input type=hidden name=item1 value=10>Batteries $10<br>
<input type=hidden name=item2 value=99>Biology textbook $99<br>
<input type=hidden name=item3 value=25>Britney Spears CD $25<br>
<input type=submit>Confirm purchase
</form>
 </body></html>
```

Instant Shop example: Malicious client

```
<html><body>
 <form action=commit_sale.cgi>


<input type=hidden name=item1 value=0>Batteries $10<br>
<input type=hidden name=item2 value=0>Biology textbook $99<br>
<input type=hidden name=item3 value=0>Britney Spears CD $25<br>
<input type=submit>Confirm purchase
</form>
 </body></html>
```

Security through obscurity: NeBride.com

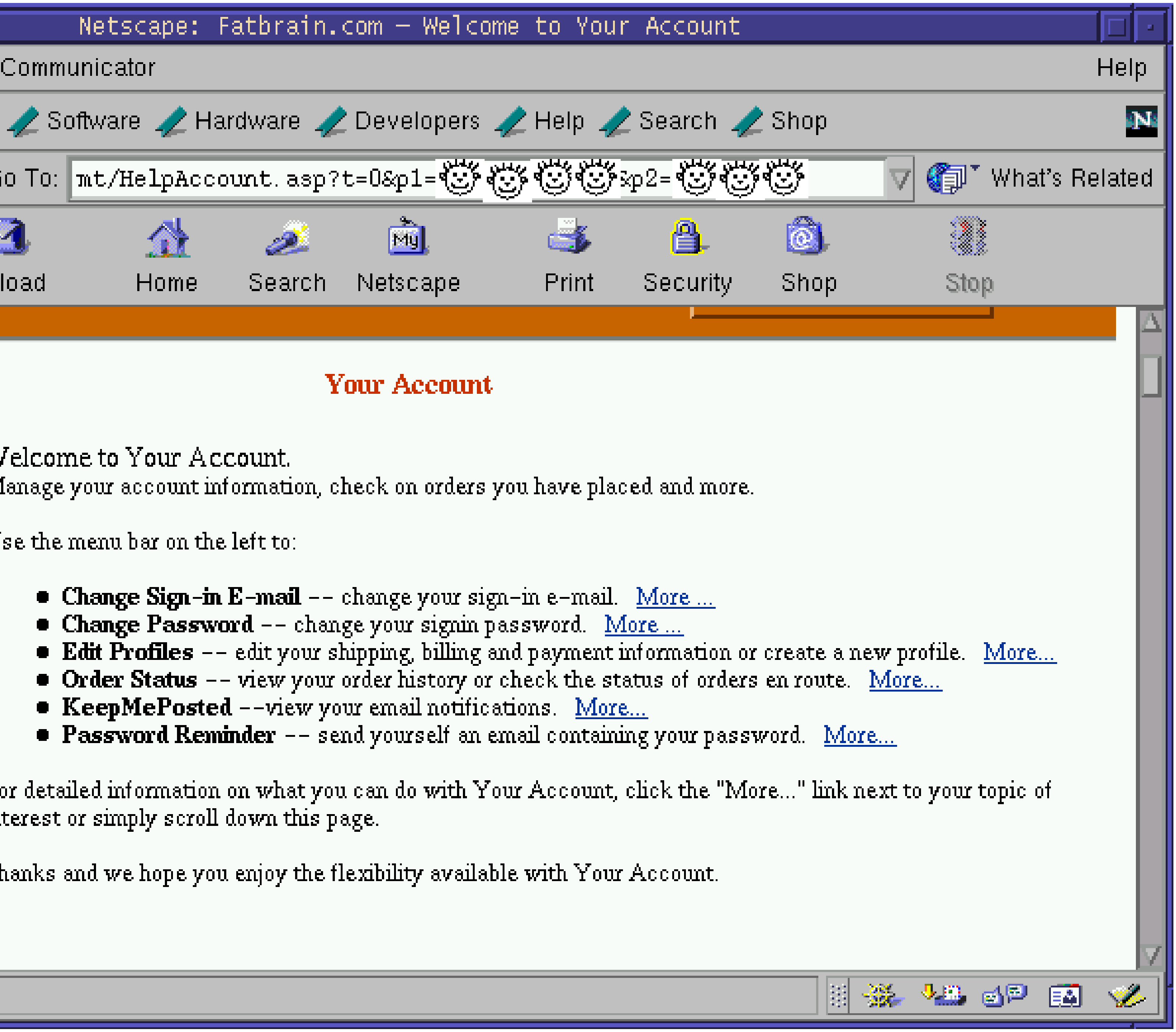
- **Problem: No cryptographic authentication at all**
- **Cookie (authenticator) is the username**
- **Create a cookie with someone's username**
  - Instant access to her name, address, phone number, e-mail address, wedding date and place, and password.

Predictable sequence numbers: fatbrain.com

- **Problem: Customer can determine the authenticator for any other user.**
- **Authenticators are sequence numbers in the URL.**

https://www.fatbrain.com/HelpAccount.asp?t=0&p1=fubob@mit.edu&p2=540555758
https://www.fatbrain.com/HelpAccount.asp?t=0&p1=nobob@mit.edu&p2=540555759

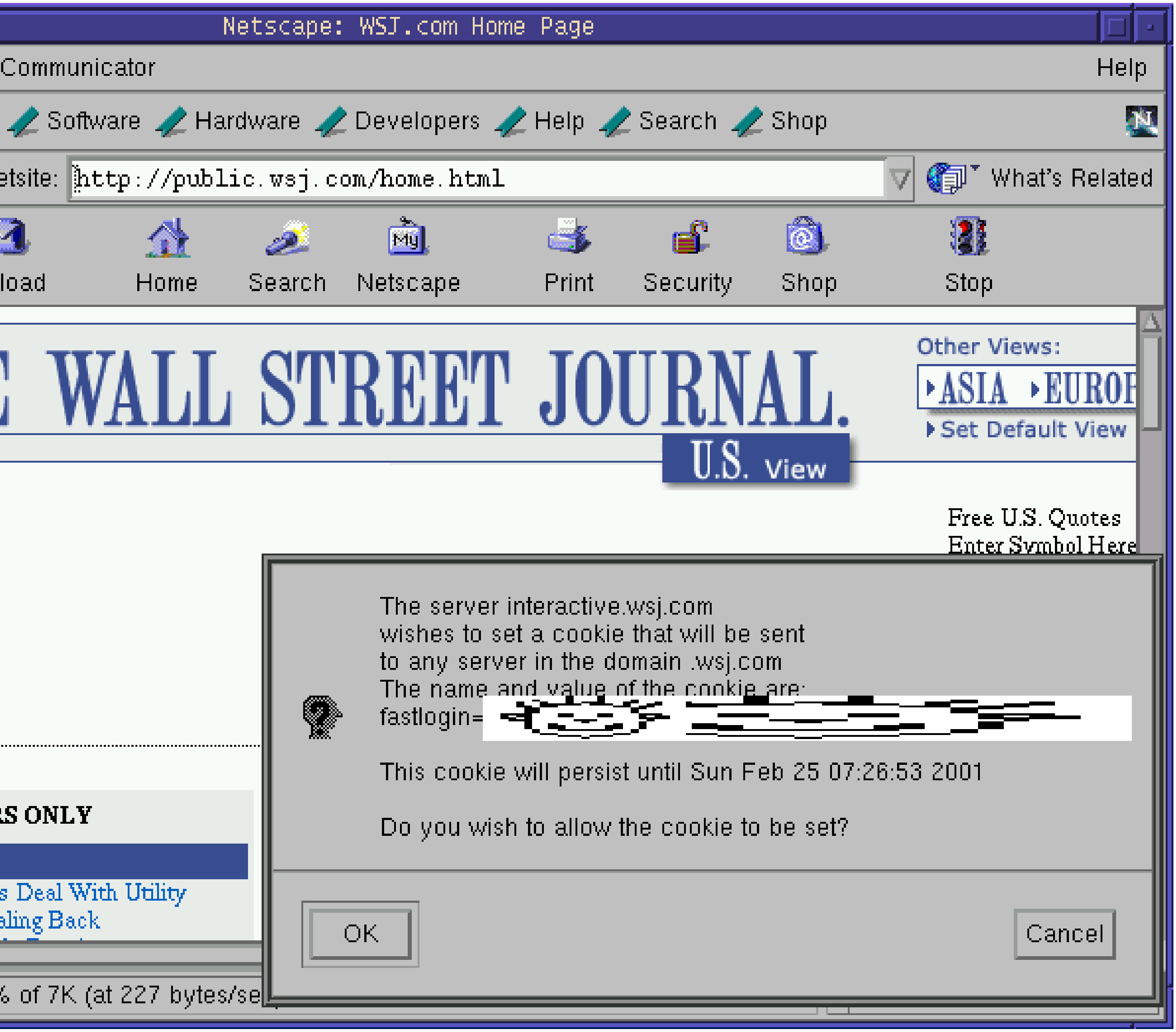
- **Guess a victim's sequence number by decrementing.**
- **Access to personal information**
- **Change address, receive password by email!**



Netscape: Fatbrain.com - Welcome to Your Account
Communicator
Help
Software
Hardware
Developers
Help
Search
Shop
o To: mt/HelpAccount.asp?t=0&p1=
p2=
What's Related
load
Home
Search
Netscape
Print
Security
Shop
Stop
Your Account
Velcome to Your Account.
lanage your account information, check on orders you have placed and more.
se the menu bar on the left to:
Change Sign-in E-mail -- change your sign-in e-mail. More ...
Change Password -- change your signin password. More ...
Edit Profiles -- edit your shipping, billing and payment information or create a new profile. More...
Order Status -- view your order history or check the status of orders en route. More...
KeepMePosted --view your email notifications. More...
Password Reminder -- send yourself an email containing your password. More...
or detailed information on what you can do with Your Account, click the "More..." link next to your topic of
terest or simply scroll down this page.
hanks and we hope you enjoy the flexibility available with Your Account.

wsj.com

- Authenticate subscribers with stateless servers
- Half million paid-subscriber accounts
- Purchase articles, track stock portfolios


Netscape: WSJ.com Home Page
Communicator
Help
Software
Hardware
Developers
Help
Search
Shop
etsite:
http://public.wsj.com/home.html
What's Related
load
Home
Search
Netscape
Print
Security
Shop
Stop
WALL STREET JOURNAL.
U.S. View
Other Views:
ASIA
EUROP
Set Default View
Free U.S. Quotes
Enter Symbol Here
RS ONLY
s Deal With Utility
aling Back
% of 7K (at 227 bytes/se
The server interactive.wsj.com
wishes to set a cookie that will be sent
to any server in the domain .wsj.com
The name and value of the cookie are:
fastlogin=[redacted]
This cookie will persist until Sun Feb 25 07:26:53 2001
Do you wish to allow the cookie to be set?
OK
Cancel

wsj.com analysis

- **Reality: fastlogin =**

  user + UNIX-crypt (user + server secret)

- **Easily produce fastlogin cookies**

username	crypt() Output	fastlogin cookie
bitdiddl	MaRdw2J1h6Lfc	bitdiddlMaRdw2J1h6Lfc
bitdiddle	MaRdw2J1h6Lfc	bitdiddleMaRdw2J1h6Lfc

- **Usernames matching first 8 characters have same authenticator**

Obtaining the server secret?

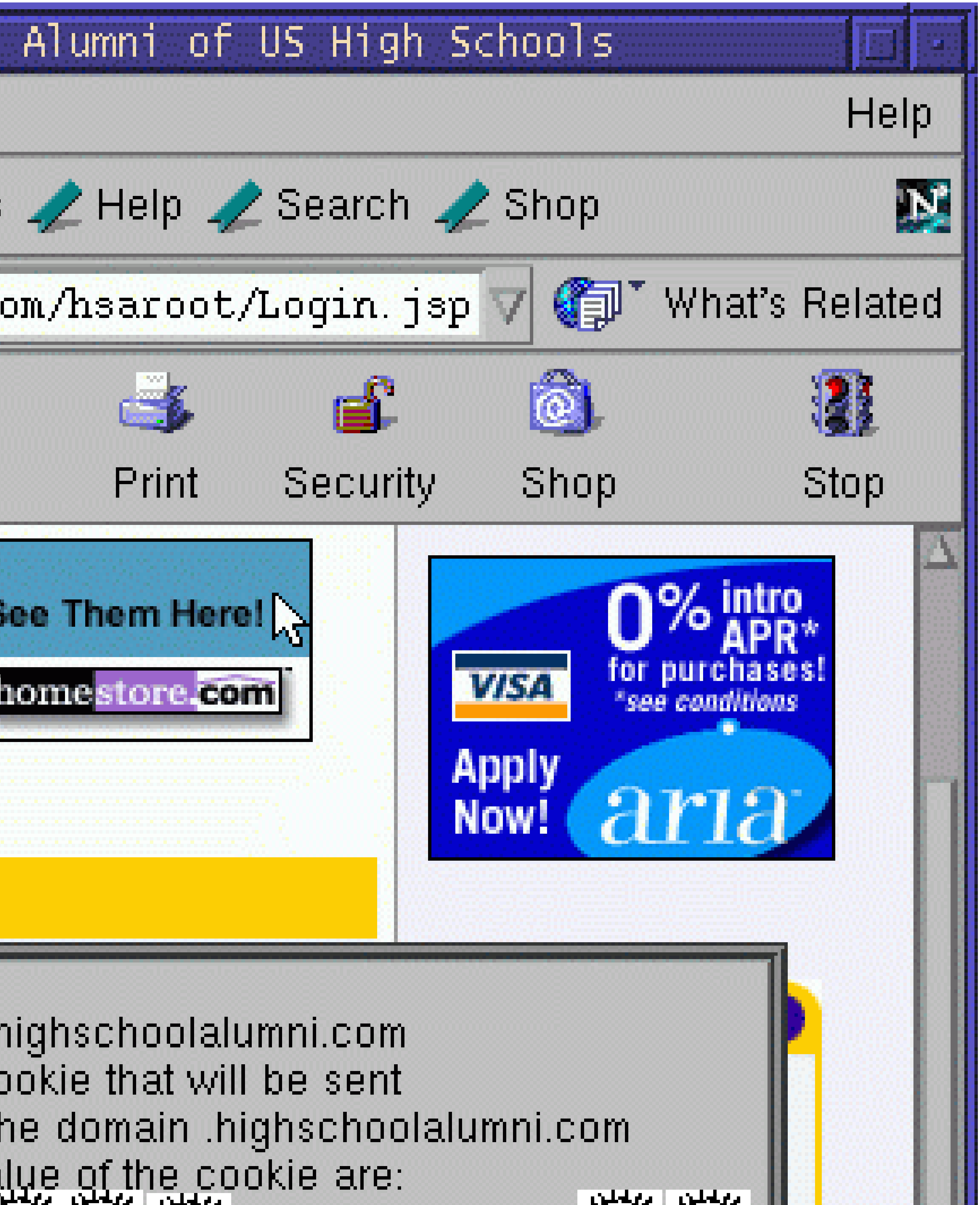
- **Adaptive chosen message attack**
- **Perl script queried WSJ with invalid cookies**
- **Runs in max 128×8 queries rather than intended 128^8 (1024 vs. 72057594037927936)**
- **1 sec/query yields 17 minutes vs. 10^9 years**
- **The key is "March20"**

How the attack works

Secret guess	username	crypt input	worked?
	bitdiddl	bitdiddl	Yes
A	bitdidd	bitdiddA	No
...	...	...	...
M	bitdidd	bitdiddM	Yes
MA	bitdid	bitdidMA	No
...	...	...	...
Ma	bitdid	bitdidMa	Yes
...	...	...	...
March20	b	bMarch20	Yes

Lack of cryptography: highschoolalumni.com

- Problem: No cryptographic authentication at all
- Cookie authenticator is the public username and public user ID


Alumni of US High Schools
Help
Help
Search
Shop
om/hsaroot/Login.jsp
What's Related
Print
Security
Shop
Stop
See Them Here!
homestore.com
0% intro APR*
for purchases!
*see conditions
VISA
Apply Now!
aria
highschoolalumni.com
ookie that will be sent
he domain .highschoolalumni.com
alue of the cookie are:

A simple scheme that works

$$\textbf{auth} = \texttt{expire} + \texttt{data} + \texttt{MAC}_k(\texttt{expire} + \texttt{data})$$

where `MAC` **could be HMAC-SHA1,**

`data` **could be a username or capability, and**

'+' denotes concatenation with a delimiter

Secure against interrogative adversary

But of course, MAC what you mean!

- **badauth = MAC (key, username + expiration)**
  - (Alice, 21-Apr-2001) → MAC (key, Alice21-Apr-2001)
  - (Alice2, 1-Apr-2001) → MAC (key, Alice21-Apr-2001)
- **Same authenticator!**
- **Use unambiguous representation or delimiters**