Modern Information Security for the Internet

Thomas Beebe
Advanced DataTools Corporation

Change Session Code
Day, April 0, 2008 • 00:00 a.m. – 00:00 a.m.

2008 IIUG Informix Conference

www.iiug.org

Modern Security Threats

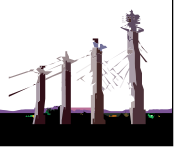
- Most cases of systems being compromised are internal
- Outside hackers are often no longer bored teenagers, there are organized rings of crackers out there, many work for assorted groups of Russian mafia.
- There is always the threat of accidental data loss: 'delete from customers ' -oops I hit enter.
- It is far too easy sometimes to just lose a laptop or backup tape.

What Happens To Data

- Sold to spammers
- Sold to scammers
- Used to blackmail the company
- Compromised systems are used to send spam or launch DOS attacks
- Once one system is compromised, it is often trivial to compromise others.
- This can greatly hurt a companies public image.

How Do They Get In?

- Internally:
  - They already have access
  - Lack of firewalls inside the company
  - Internal servers are not kept up to date on patches.
  - Social engineering inside the company
  - Easy access to the physical devices
  - Network taps can go unnoticed.

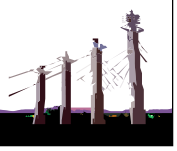


How Do They Get In

- Outsiders:
  - SQL Injection
  - Weak Passwords
  - Client side data verification
  - Improper validation of user input
  - Missing OS level patches
  - Open ports
  - Cross Site Scripting

SQL Injection

- Far far too common in many applications.
- Most languages have ways to prevent against it, but they are often avoided because they can be cumbersome.
- Select user_id from users where user_id = '$username' and password = '$password';
- What happens if password is inputted as:
  1' or 1=1 --;
- Or worse
  1'; delete from users;



Client Side Verification

- You cannot trust anything from users.
- Many sites use javascript to make sure data is valid before being inputted. You can turn javascript off, or some people on older browsers will not have it enabled to begin with.
- Users can get around anything that is installed on their own systems, binary files (activex) can be modified, replaced, and used.
- Network packets can be edited in real time from the user.

Cross Site Scripting

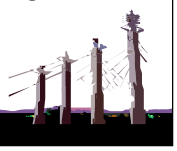
- Many sites expect the form data they are passing to the scripts is stable and what they expect.
- It is easy to write a local html file that has the form action=http://your.site/script.php
- That can bypass all client side prevention.
- Also you never know what innocent users may try to input into fields anyway with no malicious intent.

What Happens If They Get In

- What data in the database can they access without a key?
- What other databases does that db login have access to?
- Can the webserver user modify any scripting language files to put up harmful data on YOUR website?
- Do the firewalls block outbound traffic on other ports?
- Does the system have passwordless logins to other server?

What Can They Do

- Is the OS fully up to date with patches?
  - Are you sure?
- If a cracker gets root, you can no longer trust anything on that system.
- It is possible to set up shell access through port 80.
- How will the public view the breakin?
- Does your state require notification of any potentially affected people? (You should do this anyway)

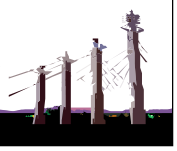


Common Attack Outline (Scriptkiddie)

- Common software package is installed and not fully patched (phpBB, awstats, webmin)
- Automated tool is used to scan large swaths of public IP addresses for that software in standard directories.
- If found it runs the attack and gives a shell to the kiddie
- Goes in, updates your website with their personal 'hacked by xxxx' tag and goes to see what they can find on the system.

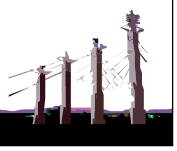
Common Attack Outline (Insider)

- Already has access, or asks their buddy down in IT for 'temporary access' to fix a problem.
- Here it depends if they are motivated by greed or revenge.
  - Greed:
    - Takes the data and sells it to the highest bidder, often spammers or scammers.
  - Revenge:
    - Sets up timeboms.
    - Deletes vital data, as well as backups.
    - Removes all internal knowledge of widget A



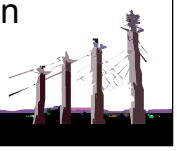
Common Attack Outline (Spammer)

- Finds a target they want to use, often also using the automated tools.
- Gains access to the system, tried to gain root using a privilege escalation vulnerability.
- Adds a rootkit to the system
- Sets up a way to access the system, this is often an IRC bot.
- Sets up a hidden mail server relay
- Starts pumping spam out from your IP.



Common Attack Outline (Pro)

- Analyzes your network for open ports or well known vulnerable software packages.
- Creates new attacks against the software you have installed based on standard mistakes (weak passwords, sql injection, etc)
- Gains access, adds a rootkit, clears the logs, remains low profile and sets up ways to pull as much vital data as they can.
- Uses the data in blackmail or in rings of stolen personal information.

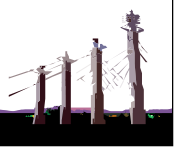


Prevention

- Plan out your application network.
- Keep systems logically separated
- Keep employee access limited to what they need to do their job. It is a tightrope walk to do this right.
- Keep a list of all software packages installed, their versions, and someone in charge of making sure each one is up to date.
- Strong password policies are a must, as well as making it very clear users do not share them.

Application Development

- Plan out your application with an eye for security.
- Make sure all database connection information is private.
- Make sure **ALL** data inputted from outside the database is valid, don't assume that it will be a number.
- Make sure all quotes are handled properly, most anyone named O'Kelley will have a story relating to this one.



Password Handling

- Unless you must, never store passwords in plain text.
- Using outside authentication systems (LDAP, AD, PAM) can be a good way to remove that as a weak point.
- If possible just store a salted password hash: 'mypass' is stored as md5('mypass' + 'random_string')
- MD5 can be matched if someone has the hash, however salted hashes prevent this.

Data Input

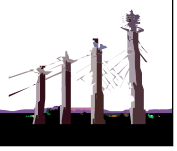
- Quotes can be really tricky.
- Most languages have prebuilt functions to do quote escaping (\'), don't reinvent the wheel.
- Bound parameters are easy and commonly available.
- When possible force the data type, there is no reason not to force INT when that is the expected datatype.
- Never underestimate users ability to put in bad data, validate everything.

Public Information

- How much data does your application give back to the user.
- Is it possible for someone to figure out it is a valid login but an invalid password?
- Are you storing anything that can be edited or used in cookies?
- What about in hidden forms?
- Is the database connection file in a directory someone might be able to access? What about from the web?

Encryption

- Most financial or medical data, as well as many government agencies have requirements on encryption.
- Informix has it, use it if you need it.
- Only encrypt the sensitive data for speed purposes
- If someone gets access to your web server, and has a public login to your database, can they select anything that will get you quickly out of your job?

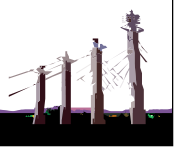


Cross Site Scripting

- This is easy to avoid but often overlooked.
- Most webservers will return data of where a form is submitted from.
- Apache: [HTTP_REFERER] 216.177.38.211/test.php
- Just do a validation check to make sure it is coming from where you expect it to.

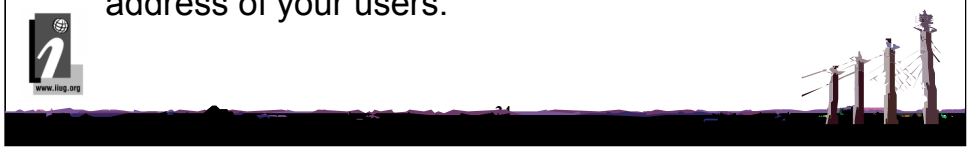
Other Suggestions

- Keep md5 sums of all of your webserver files on a read only medium. Track if any change unexpectedly.
- Insiders are the hardest to prevent against, much of that prevention is just making sure you trust your admins and people with sensitive data. Also take careful precautions when firing someone with that kind of access.



Other Suggestions

- Log basic information about who uses your application (IP, time, login name etc). This is good for marketing, and also useful to see if you have people trying automated attacks against you.
- Set up a way to blacklist an IP. If you get 3000 hits from a script kiddie, set up a script to add that IP to a firewall to block the person from trying again.
- Sysadmins: Read your logs. Keep an eye on the IP address of your users.



Session ####
Session Title

Thomas Beebe
Advanced DataTools Corporation
tom@advancedatatools.com