

Unix System Administration

Chris Schenk

Lecture 20 – Tuesday Apr 01

CSCI 4113, Spring 2008

Preventing SQL Injection

- Very, very simple ways to mitigate injection
- Do any of the following:
 - Check input fields for valid input
 - Characters and length (maybe with regex)
 - Use prepared statements
 - Inputs cannot modify the query itself
 - Use stored procedures
 - Even better than prepared statements, eliminates writing actual SQL queries!
 - Notify admins on query syntax errors
 - Error reporting on the application itself

PHP + MySQLi

- Rewritten mysql connector in PHP
 - Object-oriented in PHP5
- Package php5-mysqli under Ubuntu
 - You will install this for your lab
- Gives access to prepared statements
 - Very important for avoiding SQL injection
- Fantastic walkthrough page:
 - <http://devzone.zend.com/node/view/id/686>
 - Has everything you need to connect and query safely

PHP Prepared Statements

- We have to convert our mysql code to mysqli
 - Newer, better library in php5
 - Object oriented, must keep track of our connection within a variable (\$conn in the example)
- Three extra function calls required
 - `$stmt->prepare($query);`
`$stmt->bind_param($types, $var1, ...)`
`$stmt->execute();`
- Prepared statements effectively prevent your code from being susceptible to SQL injection

Using Prepared Statements

- Create a new mysqli object

- `$conn = new mysqli($host, $user, $pass, $db);`

- Uses question marks as placeholders in queries

- `$stmt = $conn->prepare("INSERT INTO theTable (col1, col2) VALUES (?, ?)");`

- Bind parameters (with associated types)

- `$stmt->bind("sd", $input1, $input2);`

- There are four types for binding

- s (strings and other data), i (int), d (doubles/floats), b (blobs/binary data)

- Remember to close statements & connections

- `$stmt->close(); $conn->close();`

Email

- One of the primary uses of the internet
 - Aside from porn
- Mail routing almost as complicated as IP
 - Enter MX records from DNS
- Email system consists of parts called 'agents'
 - Each agent is responsible for a part of the system
- Five different agents
 - user, submission, transfer, delivery, access
- Mail servers perform two roles (usually)
 - submission and transfer agents

Mail Relays

- The process of sending mail is called “relaying”
- Open relays mean spam
 - Our jobs are to configure servers to act nice
- Relay rules based on users and aliases
 - Accept mail for valid users on our machines
 - Maybe even forward if they so desire
 - Only allow mail not destined for this host to be sent by authenticated users (SMTP AUTH)
- Relaying is done with SMTP
 - Not to be confused with IMAP or POP

Email and DNS

- Email closely tied to DNS information
 - Many rules for mail routing based on DNS info
 - MX records, reverse PTR records, A records, etc
- MX records are the routing table for email
 - Like layer two's information
 - `netstat -nr`
 - Maintained by the admin of the domain
- Can restrict mail based on PTR records
 - IF A record != PTR record, don't route mail
 - For incoming hosts!

DNS MX records

- Usually an RR has 5 pieces of information
 - Node in tree, TTL, class, type, value
- Value can have more than one piece of info
 - SOA has 7 pieces
 - Refresh, retry, etc
 - MX has two
- MX contains the priority of the mail exchanger
 - The LOWER the number, the HIGHER the priority
 - IN MX 50 cs.colorado.edu.
 - IN MX 10 coolname.cs.colorado.edu.

Email – User and Access Agents

- Mail User Agent (MUA) – end-user programs
 - Pine, webmail, thunderbird, etc
- May have Mail Access Agent (MAA) as well
 - Allows reading mail via POP or IMAP
- Might have MIME rendering/encoding functionality
 - Converting binary files to base64 and back
- Should allow access to mail header fields
 - Sometimes it's hard to find them, but they are critical to debugging

Email – Submission and Delivery Agents

- Mail Submission Agent (MSA)
 - Receives mail from MUAs on port 587
 - Also useful for security and validity of email
 - Makes sure the right headers are intact and valid
 - Potentially scan for viruses
- Mail Delivery Agent (MDA)
 - Actually sends a piece of mail to a user's inbox
 - Once an MTA receives a piece of mail destined for the local machine
 - Can be any program that accepts STDIN
 - Most common is `procmail`

Email – Transfer Agent (MTA)

- Delivers mail between machines
 - May queue of mail in local files for future delivery
- Most machines communicate on port 25 for delivery
 - Now also listen on port 587 like MSAs
- Sending mail to another machine is called 'relaying'
 - 'open' relays send mail to anyone (evil)
- MTAs speak SMTP on port 25
 - We'll look at an example transaction

Anatomy of an Email Message

- Three main parts to an email message
 - envelope, headers, body
- Envelope
 - You never see this, it's used internally by the MTA
 - (Usually) corresponds to the 'From' and 'To' headers
 - But not always!
- Body
 - Actual content of the message
 - MUST be in plain text, all binary data is converted to base64
 - The body is separated from headers by a blank line

Anatomy of an Email Message (cont)

- Headers
 - These are Key: Value pairs that conform to RFC822 standards
 - Some are mandatory: From, To, and others
 - If omitted, usually considered spam
 - Others are optional: Subject, Cc, Reply-To
 - Still others are ignored by the mail system
 - All headers are propagated, whether read or not
 - Headers that start with 'X-' are application specific
 - But again, are still sent even if unused

Reading Email Headers

- Important to know for tracing emails to their source
 - and diagnosing other problems
- The `Received` lines are the most important
 - Each intervening MTA adds a `Received` header
 - Details the route taken by that piece of email
 - Bottom 'received' line is the first
- **Received: from refuge.Colorado.EDU** (refuge.colorado.edu [128.138.238.167])
by **mx13.Colorado.EDU** (Spam Firewall) with ESMTTP id 8DC451830CE
for <schenk@ucsub.Colorado.EDU>; Tue, 4 Apr 2006 11:36:48 -0600 (MDT)
- **Received: from refuge.Colorado.EDU** (localhost [127.0.0.1])
by **refuge.Colorado.EDU** (8.13.6/8.13.3/UnixOps+Hesiod+SSL) with ESMTTP id
k34HamxB000970
for <schenk@colorado.edu>; Tue, 4 Apr 2006 11:36:48 -0600 (MDT)

Other Header Lines

- `Message-Id` – Added by the originating MTA, **unique**
- `Return-Path` – Supposed to contain the *Envelope* sender's address, bounced mail is sent here
- `Date` – Added by the originating MUA
- `Received` – Mixed in with the above
- `From`, `To`, `Subject`, etc – All follow the above
- Many other fields (usually starting with `X-`)

MIME Encoding

- Makes use of header fields
- `MIME-Version: 1.0`
`Content-Type: multipart/mixed;`
`boundary="-----_NextPart_000_00DE_01511A02.DB1A02A0"`
`Content-Transfer-Encoding: 7bit`
- Content-Type indicated – there are many types
 - 'multipart' indicates there will be several documents
 - 'mixed' means they may be different types
- boundary – A string inserted between each attachment
 - So the MUA knows how to split the documents

Email Summary (so far)

- Five types of mail agents
 - User, access, transfer, submission, delivery
- Three parts of an email
 - Envelope, headers, body
- Envelope ONLY used by MTAs
 - Determines where mail is actually sent
 - Body contains all attachments
- MX records determine hosts that handle mail
 - Received-From lines show mail route

Mail Agents Summary

- User and Access agents
 - User renders emails (Pine, Thunderbird, etc)
 - Access reads email from mailbox files (IMAP)
- Transfer and Submission agents
 - Transfer is mail server that routes email
 - Submission agent receives email for delivery
- Delivery agent
 - Actually sends mail to the user's mailbox
 - 'Procmail' is a common delivery agent

Email Addresses

- Complexity of mail systems stems from different addressing schemes
 - UUCP addressing: ajax!boulder!mroe!target
 - Used to have to know the hops between you and destination
 - Modern addressing – lots of variation here
 - Chris Schenk <schenkc@colorado.edu>
 - “Joe Blow” <jb@blowme.net>
 - Hans.and.Franz@pumpUup.net
 - username@machine.com
 - Mixed Addressing - ucsb!schenkc@colorado.edu

Aliases

- Provide for the re-routing of email
 - From one to one or many different destinations
- The addresses in To, Cc and From fields may be modified
- Aliases are recursive
 - With limitations to avoid mail loops (usually large though)
- Two general locations for aliases
 - /etc/aliases
 - /etc/mail/aliases

Classic Sendmail Alias Files

- `/etc/aliases` is a flat file database
 - Converted to a hash database for efficient lookups
 - The 'newaliases' (historical) command creates the hash files
 - Other types of files use hashes as well
- **Basic format:** `key: value[,value2[,value3[,...etc]]]`
 - Comments (lines starting with `#`) and blank lines are ignored
 - Simple alias is a redirect to another user: `root: schenkc`
 - Redirect to another host: `root: schenkc@colorado.edu`
 - Alias a list of users: `csci4113: chris,phil,bob,jane`

Aliases File (cont)

- Advanced format
- **Include files:** `graders: :include:/etc/postfix/graders`
- **Append to a file:** `webmaster: "/dev/null"`
 - The quotes preserve the slashes and path is always absolute
- **May also invoke a command:**
 - `schenkc: "|/usr/bin/auto-responder"`
 - This pipes the mail to stdin of the listed command
- **Mailing lists use piping to manage list email**
 - `csci4113: "|/usr/local/majordomo/bin/wrapper -L -l csci4113"`

Postfix Alias Files

- Have a different syntax than traditional files
 - `key value[,value2]` (notice the space)
- Keys and values are many times fully qualified
 - `schenc@c@elijah.cs.colorado.edu schenc@c@colorado.edu`
- Aliases are looked up in files by order of appearance
- `alias_maps: hash:/etc/aliases, hash:/etc/extra_aliases`
- When mixing alias files, Postfix files take precedence
 - Sometimes, in terms of matching lookups – it's odd behavior

.forward Files

- Users can forward mail to another location
 - .forward behaves similarly to alias files
- File exists in user's home directory
 - Permissions must be only readable and writable by owner!
- Can cause mail loops if not carefully managed
 - Internal alias loops are handled by servers, not forwarding loops
- **Syntax:** `\schenk , " | /usr/bin/vacation -f vacation.msg"`
 - The preceding `\` says force delivery of mail without forwarding

Postfix

- A much easier alternative to sendmail
 - <http://www.postfix.org>
 - Or so I tell myself, I've only used Postfix
- One main config file `/etc/postfix/main.cf`
- Important config lines:
 - `mydestination` – for whom I handle mail
 - `alias_maps`, `alias_database` – alias files
 - `smtpd_client_restrictions` – who can send mail
 - `smtpd_recipient_restrictions` – 'rcpt to' restrictions

Dovecot IMAP server

- One of a couple of servers to use
 - Cyrus and Courier IMAP servers are alternatives
- Allows access to mailboxes
 - Either securely or insecurely, imap vs imaps (SSL!)
- This is only for reading email
 - Sending requires configuration on the MTA
 - Want to allow authenticated users from anywhere (SMTP AUTH) or users on 'mynetworks'