SnapSmack Security Audit

Audit #009 · Version 0.7.102 · 2026-05-10 · Scope: Multisite remote-admin attack surface

---

Executive Summary

This audit was conducted prior to the 0.7.102 release in response to significant new attack surface introduced by the multisite remote-admin feature set: hub-to-spoke update push, AJAX update UI, SMACKATTACK spoke registration, SSO remote login, and the multisite API endpoint layer. Five findings were identified — two exploitable vulnerabilities and three defence-in-depth gaps. All five were remediated before the release was tagged and pushed to GitHub.

Scope

File / Endpoint	Role
smack-multisite.php	Hub admin dashboard — disconnect, ping, register spoke, update push
core/multisite-api.php	Spoke-side API: handshake, heartbeat, comments, posts, updates, SSO, blogroll
smack-multisite-sso.php	Hub-side SSO redirect handler
sso.php	Spoke-side SSO token validator
core/csrf.php + core/auth.php	CSRF engine and admin authentication gate
smack-settings.php	SMACKATTACK registration flow

Findings

SS-009-01 | **HIGH** | **FIXED ✓**

CSRF via GET-based state mutation (disconnect/ping)

Before

smack-multisite.php exposed ?disconnect=NODE_ID and ?ping=NODE_ID as plain GET links. PHP's CSRF engine (core/csrf.php) only runs on POST requests. Any page an authenticated hub admin visited while logged in could silently trigger a spoke disconnect via a crafted image or anchor tag pointing at the hub URL.

```
// Handler (GET)
if (isset($_GET['disconnect'])) {
    $node_id = (int)$_GET['disconnect'];
}

// Link (no CSRF protection)
<a href="?disconnect=<?php echo $n['id']; ?>">DISCONNECT</a>
```

After

Both handlers converted to POST-only. Links replaced with inline forms (method=POST), which causes CSRF tokens to be auto-injected by admin-footer.php's output-buffer pass. Same fix applied to the ping handler.

```
// Handler (POST-only)
if (isset($_POST['disconnect'])) {
    $node_id = (int)$_POST['disconnect'];
}

// Form (CSRF auto-injected by admin-footer.php)
<form method="POST" style="display:inline;">
    <input type="hidden" name="disconnect" value="<?= $n['id']; ?>">
    <button type="submit" class="action-delete">DISCONNECT</button>
</form>
```

SS-009-02 **HIGH** **FIXED ✓**

Timing attack on handshake token comparison

Before

The spoke registration handshake endpoint (multisite/handshake) compared the one-time registration token using PHP's !== operator, which is not constant-time. A sufficiently precise timing oracle against a network-accessible endpoint could leak information about matching token prefixes. The SSO token comparison in sso.php already used hash_equals() correctly — the handshake was inconsistent.

```
if (!$stored_token || $token !== $stored_token || time() > $token_expires) {
```

After

Token comparison now uses hash_equals(), consistent with the rest of the codebase.

```
if (!$stored_token || !hash_equals($stored_token, $token) || time() > $token_expires) {
```

SS-009-03 **MEDIUM** **FIXED ✓**

SSRF in spoke registration (private IPs not blocked)

Before

The register-spoke handler accepted any URL for spoke_url and immediately fired a cURL request to it. A compromised admin account could supply an internal URL (192.168.x.x, localhost) to probe internal network services via the hub server. SSL_VERIFYPEER was already true, mitigating HTTPS-target probing, but HTTP internal targets were exposed.

```
// No validation — any URL accepted
$spoke_url = rtrim($spoke_url, '/');
// cURL fires immediately against whatever was supplied
```

After

snap_is_private_url() added: parses hostname, checks loopback/private name patterns, resolves via gethostbyname(), validates with FILTER_FLAG_NO_PRIV_RANGE | NO_RES_RANGE. Registration fails with a user error before cURL is called.

```
function snap_is_private_url(string $url): bool {
    $host = parse_url($url, PHP_URL_HOST);
    if (!$host) return true;
    if (in_array(strtolower($host), ['localhost','::1','0.0.0.0'], true)) return true;
    $ip = gethostbyname($host);
    return filter_var($ip, FILTER_VALIDATE_IP,
        FILTER_FLAG_NO_PRIV_RANGE | FILTER_FLAG_NO_RES_RANGE) === false;
}
if (snap_is_private_url($spoke_url)) {
    $err = 'Spoke URL must be a publicly accessible HTTPS address.';
}
```

SS-009-04 **MEDIUM** **FIXED ✓**

comments/action API missing hub role enforcement

Before

The POST multisite/comments/action endpoint validated only that the caller had a valid Bearer token — it did not check caller role. Other sensitive endpoints (updates/trigger, ban-sync) already enforced role = hub. The absence here was an inconsistency that would matter if a spoke's API key were ever compromised or shared.

```
if ($resource === 'comments' && $sub_action === 'action') {
    // No role check
    $comment_id = (int)($_POST['comment_id'] ?? 0);
```

After

Role check added, matching the pattern used on updates/trigger and ban-sync.

```
if ($resource === 'comments' && $sub_action === 'action') {
    if ($node['role'] !== 'hub')
        ms_err('Only a hub may moderate comments on this spoke', 403);
    $comment_id = (int)($_POST['comment_id'] ?? 0);
```

SS-009-05 **MEDIUM** **FIXED ✓**

Cross-post: image content not validated before write

Before

multisite/posts/create fetched a hub-supplied image URL, validated only the file extension against an allowlist, and wrote the bytes directly to disk. A compromised hub could supply a URL returning non-image content (PHP, HTML) with a .jpg extension. Default server config serves by extension (safe), but a misconfigured server could execute it.

```
if (!$img_binary || $fetch_code !== 200) {
    ms_err('Could not fetch image from hub: HTTP ' . $fetch_code);
}
// Wrote directly to disk — no content check
```

After

getimagesizefromstring() called on fetched bytes before write. Non-image content rejected regardless of extension.

```
$img_info_pre = @getimagesizefromstring($img_binary);
if (!$img_info_pre) {
    ms_err('Fetched content is not a valid image');
}
```

Summary

ID	Severity	Title	Status
SS-009-01	HIGH	CSRF via GET-based state mutation (disconnect/ping)	**Fixed**
SS-009-02	HIGH	Timing attack on handshake token comparison	**Fixed**
SS-009-03	MEDIUM	SSRF in spoke registration (private IPs not blocked)	**Fixed**
SS-009-04	MEDIUM	comments/action API missing hub role enforcement	**Fixed**
SS-009-05	MEDIUM	Cross-post: image content not validated before write	**Fixed**

2 High · 3 Medium · 0 Low · All findings fixed before release tag.