CVE-2015-1522

描述：analyzer/protocol/dnp3/DNP3.cc in Bro before 2.3.2 does not reject certain non-zero values of a packet length, which allows remote attackers to cause a denial of service (buffer overflow or buffer over-read) via a crafted DNP3 packet.

软件：bro-master

源码：src/analyzer/protocol/dnp3/DNP3.cc

出现位置： 138-146行、186-189行、197-201行

关键源码：

if ( endp->in\_hdr )

{

// We're parsing the DNP3 header and link layer, get that in full.

if ( ! AddToBuffer(endp, PSEUDO\_APP\_LAYER\_INDEX, &data, &len) )

return true;

// The first two bytes must always be 0x0564.

if( endp->buffer[0] != 0x05 || endp->buffer[1] != 0x64 )

{

if ( ! endp->in\_hdr )

{

assert(endp->pkt\_length);

int n = PSEUDO\_APP\_LAYER\_INDEX + (endp->pkt\_length - 5) + ((endp->pkt\_length - 5) / 16) \* 2

+ 2 \* ( ((endp->pkt\_length - 5) % 16 == 0) ? 0 : 1) - 1 ;

if ( ! AddToBuffer(endp, n, &data, &len) )

return true;

bool DNP3\_Base::AddToBuffer(Endpoint\* endp, int target\_len, const u\_char\*\* data, int\* len)

{

if ( ! target\_len )

return true;

int to\_copy = min(\*len, target\_len - endp->buffer\_len);

memcpy(endp->buffer + endp->buffer\_len, \*data, to\_copy);

\*data += to\_copy;

\*len -= to\_copy;

endp->buffer\_len += to\_copy;

return endp->buffer\_len == target\_len;

}

bool DNP3\_Base::ParseAppLayer(Endpoint\* endp)

{

bool orig = (endp == &orig\_state);

binpac::DNP3::DNP3\_Flow\* flow = orig ? interp->upflow() : interp->downflow();

u\_char\* data = endp->buffer + PSEUDO\_TRANSPORT\_INDEX; // The transport layer byte counts as app-layer it seems.

int len = endp->pkt\_length - 5;

// DNP3 Packet : DNP3 Pseudo Link Layer | DNP3 Pseudo Transport Layer | DNP3 Pseudo Application Layer

// DNP3 Serial Transport Layer data is always 1 byte.

// Get FIN FIR seq field in transport header.

// FIR indicate whether the following DNP3 Serial Application Layer is first chunk of bytes or not.

// FIN indicate whether the following DNP3 Serial Application Layer is last chunk of bytes or not.

int is\_first = (endp->tpflags & 0x40) >> 6; // Initial chunk of data in this packet.

int is\_last = (endp->tpflags & 0x80) >> 7; // Last chunk of data in this packet.

int transport = PSEUDO\_TRANSPORT\_LEN;

int i = 0;

while ( len > 0 )

{

int n = min(len, 16);

// Make sure chunk has a correct checksum.

if ( ! CheckCRC(n, data, data + n, "app\_chunk") )

return false;

// Pass on to BinPAC.

assert(data + n < endp->buffer + endp->buffer\_len);

flow->flow\_buffer()->BufferData(data + transport, data + n);

transport = 0;