Will Dormann  wd@cert.org
Will has been a software vulnerability analyst with the CERT Coordination Center (CERT/CC) since 2004. His focus areas include web browser technologies, ActiveX, and fuzzing. Will has discovered thousands of vulnerabilities using a variety of tools and techniques. He is the author & maintainer of the CERT Vulnerability note for Heartbleed (VU#720951).
URLs of Will’s work:

http://www.kb.cert.org/vuls/id/720951
https://www.cert.org/blogs/certcc/
http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=53466
Robert Seacord  rcs@cert.org
Robert is the Secure Coding Technical Manager. He is the author of The CERT C Secure Coding Standard (Addison-Wesley, 2014) and Secure Coding in C and C++ (Addison-Wesley, 2002) as well as co-author of two other books.

URLs of Robert’s work:

www.cert.org/secure-coding
www.securecoding.cert.org
http://url.sei.cmu.edu/k9
Christopher Clark  chris@codenomicon.com  
Chris, a twenty-two year veteran of the Information Technology world, is a Security Engineer at Codenomicon. Chris utilizes his extensive background and experience to help organizations effectively integrate meaningful security practices into their environments.
Brent Kennedy  bkennedy@cert.org
Brent Kennedy is a member of CERT's Cyber Security Assurance team focusing on penetration testing operations and research. Brent leads an effort that partners with the DHS National Cybersecurity Assessments and Technical Services (NCATS) team to develop and execute a program that offers risk and vulnerability assessments to federal, state, and local entities.
William Nichols  wrn@sei.cmu.edu
Bill joined the SEI in 2006 as a senior member of the technical staff and serves as a Personal Software Process (PSP) instructor and Team Software Process (TSP) Mentor Coach with the TSP Initiative within the Software Solutions Division (SSD). His interests include measuring software process with a focus on the economics of software quality in development.
URLs of Bill’s work:
http://works.bepress.com/william_r_nichols/
http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=59393
A Discussion on Heartbleed: Analysis, Thoughts, and Actions

© 2014 Carnegie Mellon University

Jason McCormick  jasonmc@sei.cmu.edu

Jason has been with SEI Information Technology Services since 2004 and is currently the Manager of Network and Infrastructure Engineering. He oversees datacenter, network, storage, and virtualization services and plays a key role in information security policy, practices, and technologies for the SEI.
HOW THE HEARTBLEED BUG WORKS:

SERVER, ARE YOU STILL THERE?
IF SO, REPLY "POTATO" (6 LETTERS).

User Meg wants these 6 letters: POTATO. User
Kida wants pages about "irl games". Unlocking
secure records with master key 513098573343.

POTATO
SERVER, ARE YOU STILL THERE? IF SO, REPLY "BIRD" (4 LETTERS).

User Meg wants these 4 letters: BIRD. There are currently 340 connections open. User Brendan uploaded the file self.crl (contents: 234b962e7ce9f6f9f1b6f8).

HMM...

User Meg wants these 4 letters: BIRD. There are currently 340 connections open. User Brendan uploaded the file self.crl (contents: 234b962e7ce9f6f9f1b6f8).
A Discussion on Heartbleed: Analysis, Thoughts, and Actions

© 2014 Carnegie Mellon University

This work is licensed under a Creative Commons Attribution-NonCommercial 2.5 License.
Heartbleed Vulnerability

```c
int dtls1_process_heartbeat(SSL *s) {
    unsigned char *p = &s->s3->rrec.data[0], *pl;
    unsigned short hbtype;
    unsigned int payload;
    unsigned int padding = 16; /* Use minimum padding */
    hbtype = *p++;
    n2s(p, payload);
    pl = p;
    if (hbtype == TLS1_HB_REQUEST) {
        unsigned char *buffer, *bp;
        int r;
        buffer = OPENSSL_malloc(1 + 2 + payload + padding);
        bp = buffer;
        *bp++ = TLS1_HB_RESPONSE;
        s2n(payload, bp);
        memcpy(bp, pl, payload);
        memcpyp(bp, pl, payload);
    }
}
```

Violates **INT04-C. Enforce limits on integer values originating from tainted sources**

Violates **ARR38-C. Guarantee that library functions do not form invalid pointers**