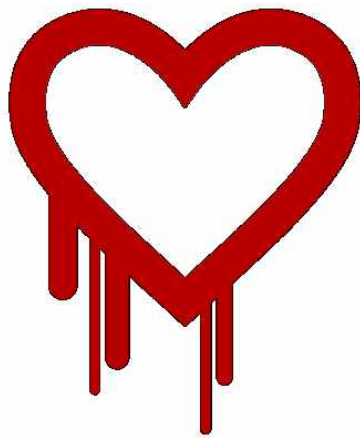


HeartBleed



지용빈

목차

- 1) 개요-----
- 2) HTTPS & SSL-----
- 3) HeartBleed란?-----
- 4) HeartBleed의 원리-----
- 5) 소스코드-----
- 6) 실습-----
- 7) 대응방안-----

1. 개요

이 문서는 2014년 4월에 발표된 CVE-2014-0160 통칭 'HeartBleed' 취약점에 대한 실습과 원리에 대해 서술한 문서로서 이해를 돕기 위해 PPT와 같이 제작되었습니다. 사실 영어 문서 찾다가 지쳐서 필자가 만들. HeartBleed 취약점을 웹에서 점검 할 수 있도록 <https://filippo.io/Heartbleed/> 에서 취약점 점검을 할 수 있습니다. 또한 <http://heartbleed.com/> 에서 별도로 정보를 제공하고 있습니다.

2. HTTPS & SSL

일반적인 HTTP는 80번 포트를 이용하여 통신을 하게 됩니다. 그러나 HTTP는 암호화가 되어 있지 않기 때문에 보안에 있어서는 좀 더 취약합니다. 그래서 보안이 좀 더 요구되는 인터넷 뱅킹 같은 업무를 할 때는 HTTPS가 사용됩니다.

HTTPS란 일반적인 HTTP 통신을 SSL이란 암호화 프로토콜로 암호화한 것입니다. SSL(Secure Socket Layer) 중에서 가장 널리 쓰이는 것은 OpenSSL이란 암호화 방식으로 오픈소스 환경에서 개발되었습니다. SSL의 정식적인 이름은 TLS(Transport Layer Security)로서 Server와 Client 사이의 통신 보안을 위한 프로토콜로서 개발 되었습니다.

3. HeartBleed란 ?

HeartBleed란 이름이 붙은 이유는 이 취약점이 OpenSSL의 TLS/DTLS HeartBeat Extension 코드에서 문제가 발생하였는데 처음 HeartBleed를 발표한 회사인 Codenomicon에서 피를 흘리는 하트모양의 이미지를 처음으로 사용해서입니다.

HeartBleed는 OpenSSL 1.0.1 ~ 1.0.1f 그리고 1.0.2-beta1 버전에서 발견된 취약점으로 OpenSSL에서 TLS/DTLS 구현이 미숙한 점을 이용한 것으로 HeartBeat 프로토콜의 패킷 조작만으로 Sever의 메모리에서 최대 64kb의 민감한 정보를 누출시킬 수 있습니다.

현재 취약한 버전을 탑재한 운영체제의 목록입니다.

Debian Wheezy (stable), OpenSSL 1.0.1e-2+deb7u4

Ubuntu 12.04.4 LTS, OpenSSL 1.0.1-4ubuntu5.11

CentOS 6.5, OpenSSL 1.0.1e-15

Fedora 18, OpenSSL 1.0.1e-4

OpenBSD 5.3 (OpenSSL 1.0.1c 10 May 2012) & 5.4 (OpenSSL 1.0.1c 10 May 2012)

FreeBSD 10.0 - OpenSSL 1.0.1e 11 Feb 2013

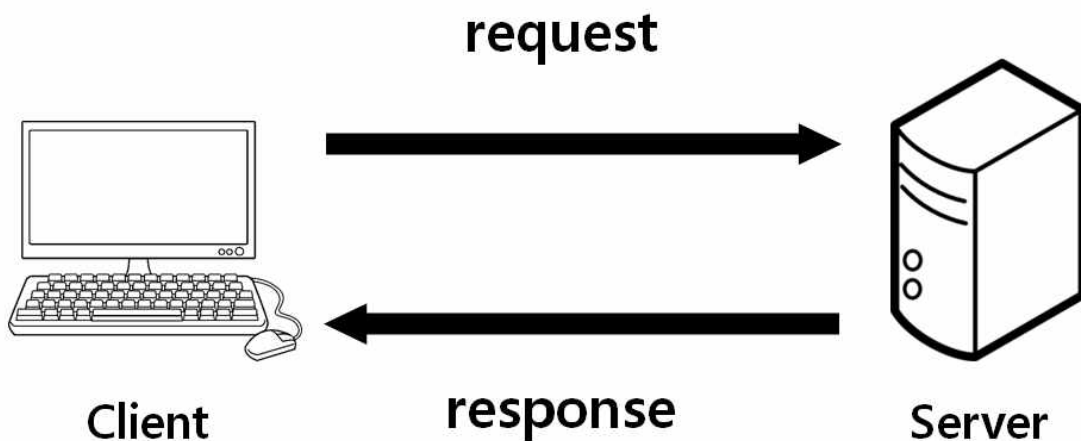
NetBSD 5.0.2 (OpenSSL 1.0.1e)

OpenSUSE 12.2 (OpenSSL 1.0.1c)

4. HeartBleed의 원리

HeartBleed 취약점은 TLS extension Library인 HeartBeat 프로토콜에서 발견되었습니다.

HeartBeat 프로토콜은 Server와 Client의 연결을 유지하기 위해 일정 신호를 주고 받는 OpenSSL 확장 프로토콜입니다.

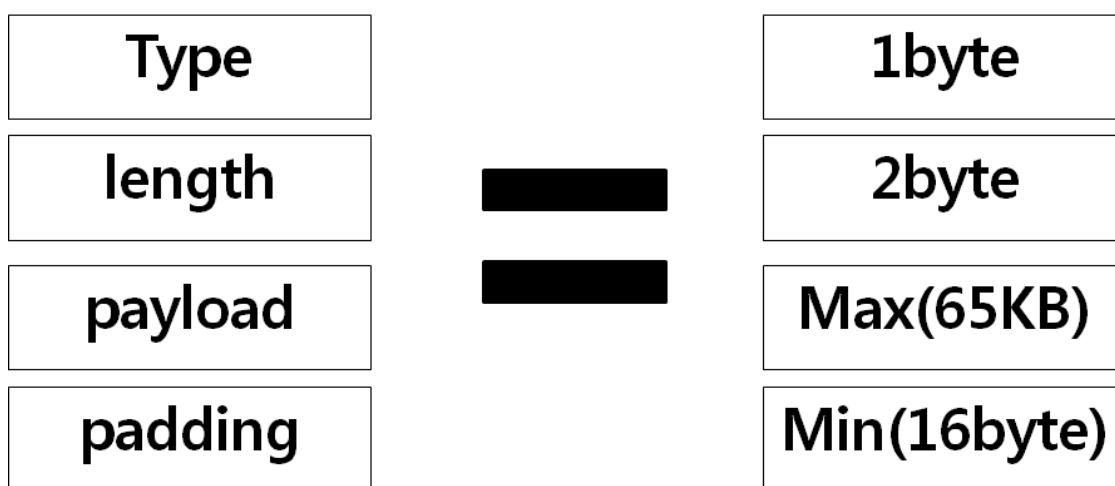


Type
length
payload
padding

이렇게 Client가 HeartBeat Request를 Server에 날리게 되면 Server는 Client에게 HeartBeat Response를 날리게 됩니다. 이 HeartBeat 프로토콜의 Packet을 뜯어보면 이렇게 4가지 구조를 이루고 있습니다.

먼저 Type은 날아온 패킷이 Response인지 Request인지를 판단하는 정보가 들어있습니다. payload_length에는 payload의 길이 값이 들어있고 payload에는 가변적인 길이의 문자열이 들어가게 됩니다. padding에는 16Byte의 랜덤 문자열이 들어가게 됩니다. 그리고 이렇게 날아온 Request와 Response 패킷은 payload_length와 payload의 값과 길이가 같아야지 만이 서로 살아있다고 판단하고 연결을 유지하게 됩니다. 이게 HeartBeat 프로토콜의 기본적인 원리입니다.

HeartBeat Request의 데이터 구조체입니다.



5. Source Code

문제가 된 HeartBeat 프로토콜의 핵심 코드입니다. 설명을 위하여 빨간색으로 주석을 달아놓았습니다.

```
#ifndef OPENSSSL_NO_HEARTBEATS
int
tls1_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 */
    /* Read type and payload length first */

```

HeartBeatRequest 구조체 선언

```

hbtype = *p++;
n2s(p, payload);
Request 패킷의 데이터에서 payload의 길이 만큼 저장
pl = p;
if (s->msg_callback)
s->msg_callback(0, s->version, TLS1_RT_HEARTBEAT,
&s->s3->rrec.data[0], s->s3->rrec.length,
s, s->msg_callback_arg);
if (hbtype == TLS1_HB_REQUEST)
unsigned char *buffer, *bp;
int r;
/* Allocate memory for the response, size is 1 bytes
* message type, plus 2 bytes payload length, plus
* payload, plus padding
*/
buffer = OPENSSL_malloc(1 + 2 + payload + padding);
bp = buffer;
/* Enter response type, length and copy payload */
*bp++ = TLS1_HB_RESPONSE;

```

Response 패킷의 크기만큼 메모리를 확보합니다.

```

s2n(payload, bp);
memcpy(bp, pl, payload);

```

이때 memcpy라는 함수가 문제를 일으켰습니다.

바로 payload의 실제 크기를 비교하지 않는다는 점입니다.

만약 hello라는 payload가 서버의 메모리상에 저장되 있을 때 length값은 5byte 여야 합니다. 근데 만약 length값을 10byte라고 속이게 된다면 서버의 메모리상에서 hello라는 5byte 외에도 서버의 메모리에 있는 나머지 5byte의 정보를 전송하게 됩니다.

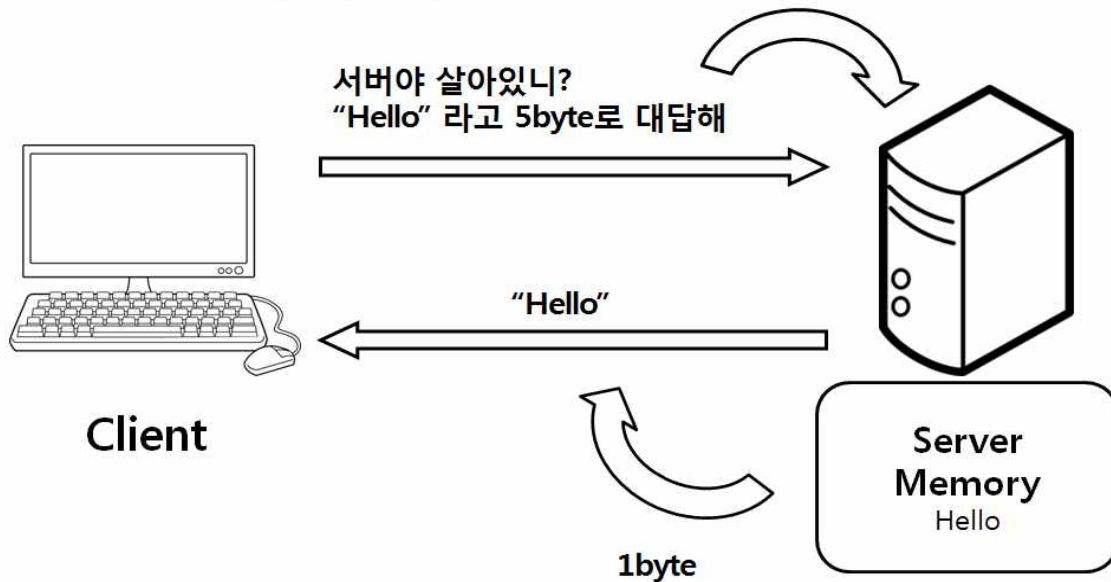
```

bp += payload;
/* Random padding */
RAND_pseudo_bytes(bp, padding);
r = ssl3_write_bytes(s, TLS1_RT_HEARTBEAT, buffer, 3 + payload + padding);

```

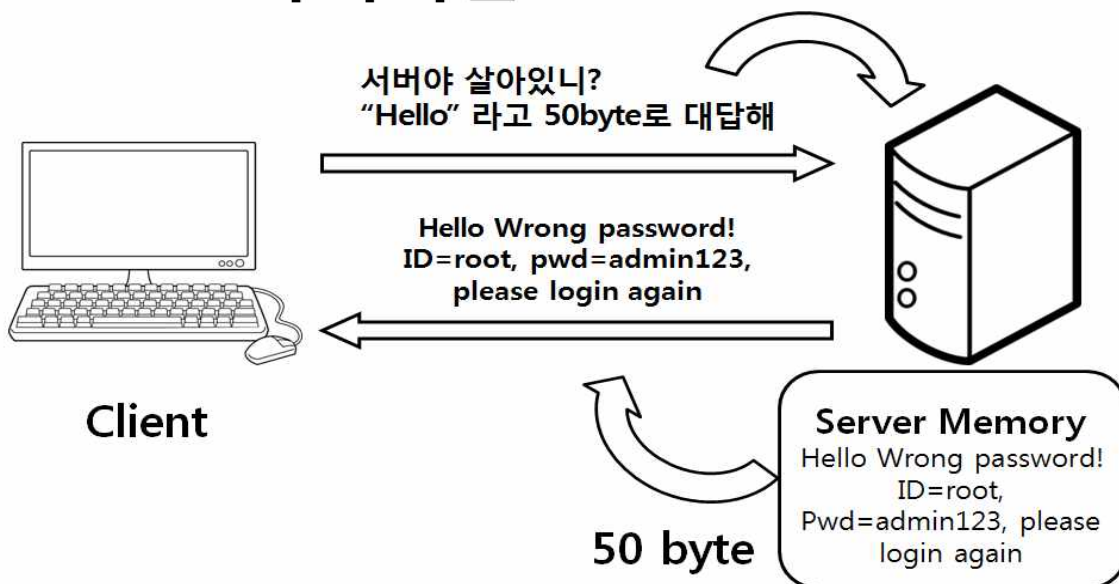
정상적인 HeartBeat 프로토콜의 구조입니다.

정상적인 HeartBeat



HeartBleed 공격의 원리입니다.

악의적인 HeartBeat



이렇게 유출된 정보는 중에는 서버의 민감 정보들이 포함될 가능성이 있습니다.

Private Key

User Data

Session

실제로 야후는 HeartBleed 취약점 발표직후에 실제 한 해커에게 공격을 당해 한 사용자의 아이디와 패스워드가 노출되는 것이 확인되었습니다.

6. 실습환경

Ubuntu 12.04

OpenSSL 1.0.1

CVE-2014-0160

<https://github.com/sensepost/heartbleed-poc>

취약한 OpenSSL 버전이 탑재된 Ubuntu 12.04를 사용했습니다.

SSL 모듈을 설치하기 전에 먼저 웹 서버가 설치되어야 합니다.

OpenSSL을 사용하기 위해서는 먼저 인증서를 발급해야 합니다.

먼저 SSL 모듈을 추가합니다.

```
root@Tempus:/home/tempus# a2enmod ssl
```

```
root@Tempus:/home/tempus# service apache2 restart
```

```
root@Tempus:/home/tempus# mkdir /etc/apache2/ssl
```

그리고 SSL 인증서를 발급합니다.

```
root@Tempus:/etc/apache2/ssl# openssl req -x509 -nodes -days 365 -newkey
```

```
rsa:2048 -keyout /etc/apache2/ssl/webserver.key -out /etc/apache2/ssl/webserver.crt
```

Generating a 2048 bit RSA private key

.....+++

.....+++

writing new private key to '/etc/apache2/ssl/webserver.key'

You are about to be asked to enter information that will be incorporated
into your certificate request.

What you are about to enter is what is called a Distinguished Name or a DN.

There are quite a few fields but you can leave some blank

For some fields there will be a default value,

If you enter '.', the field will be left blank.

Country Name (2 letter code) [AU]:KO

State or Province Name (full name) [Some-State]:Seoul

Locality Name (eg, city) []:Seoul

Organization Name (eg, company) [Internet Widgits Pty Ltd]:Tempus

Organizational Unit Name (eg, section) []:Education

Common Name (e.g. server FQDN or YOUR name) []:Tempus

Email Address []:ben3679@naver.com

```
root@Tempus:/home/tempus# ifconfig eth0
```

```
eth0      Link encap:Ethernet  HWaddr 00:0c:29:48:ed:58
          inet addr:192.168.106.128      Bcast:192.168.27.255
          Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe48:ed58/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:10289 errors:0 dropped:0 overruns:0 frame:0
          TX packets:7061 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3496929 (3.4 MB)  TX bytes:1605943 (1.6 MB)
          Interrupt:19 Base address:0x2024
```

설정 파일의 ServerName을 서버의 IP로 설정해주고

```
root@Tempus:/home/tempus# vi /etc/apache2/sites-available/default-ssl
```

```
1 <IfModule mod_ssl.c>
2 <VirtualHost _default_:443>
3     ServerAdmin webmaster@localhost
4     ServerName 192.168.106.128
5
   . . . . .
```

```
#    SSL Engine Switch:
```

```
44      #   Enable/Disable SSL for this virtual host.
45      SSLEngine on
46
47      #   A self-signed (snakeoil) certificate can be created by installing
48      #   the ssl-cert package. See
49      #   /usr/share/doc/apache2.2-common/README.Debian.gz for more
info.
50      #   If both key and certificate are stored in the same file, only the
51      #   SSLCertificateFile directive is needed.
52      SSLCertificateFile    /etc/apache2/ssl/webserver.crt
53      SSLCertificateKeyFile /etc/apache2/ssl/webserver.key
```

SSL 인증서가 있는 디렉토리로 바꿔줍니다.

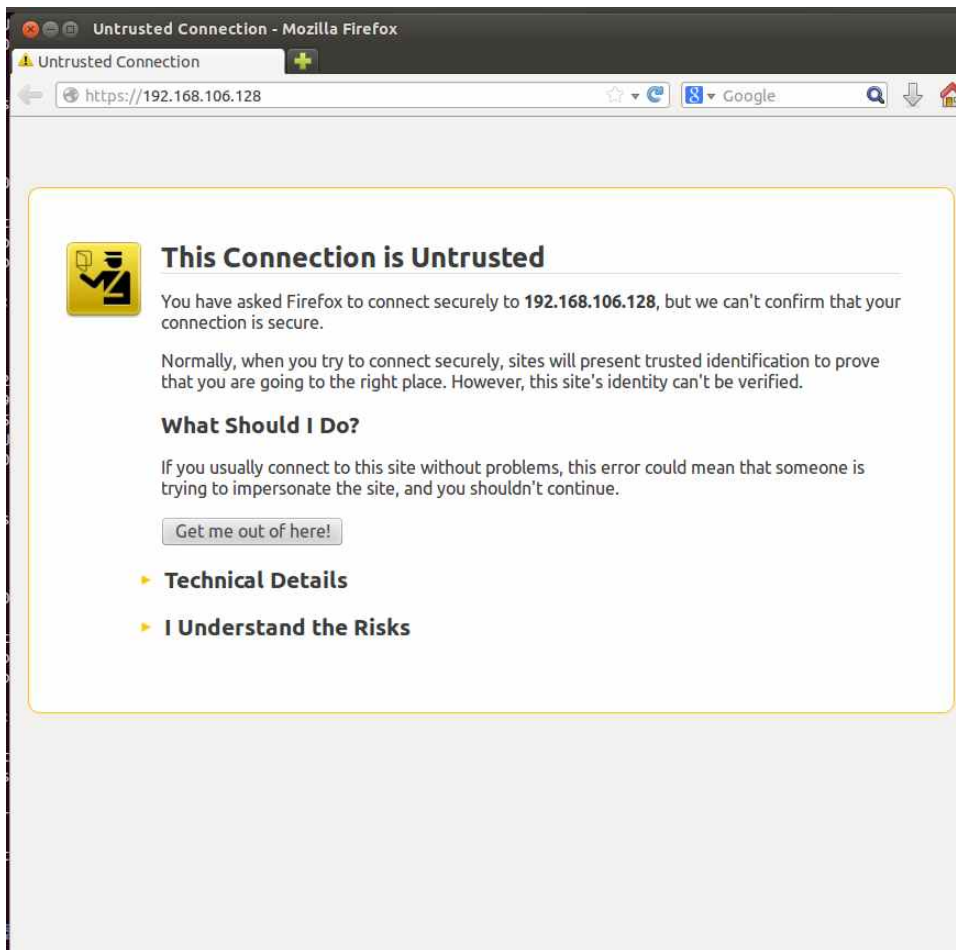
```
root@Tempus:/home/tempus# a2ensite default-ssl
```

```
root@Tempus:/home/tempus# service apache2 restart
```

이제 취약한 OpenSSL 1.0.1 버전이 탑재된 웹 서버를 완성했습니다.

이 화면이 뜨면은

I Understand the Risks를 눌러주시면 됩니다.



설치한 환경이 HeartBeat 프로토콜이 활성화 되어있는지 확인 합니다.

```
root@Tempus:/home/tempus# openssl s_client -connect 192.168.106.128:443 -tlsextdebug -debug -state | grep -i heartbeat
```

SSL_connect:before/connect initialization

SSL_connectunknown state

TLS server extension "heartbeat" (id=15), len=1

SSL_connect:SSLv3 read server hello A

depth=0 C = KO, ST = Seoul, L = Seoul, O = Tempus, OU = Education, CN = Tempus, emailAddress = ben3679@naver.com

verify error:num=18:self signed certificate

verify return:1

depth=0 C = KO, ST = Seoul, L = Seoul, O = Tempus, OU = Education, CN = Tempus, emailAddress = ben3679@naver.com

verify return:1

SSL_connect:SSLv3 read server certificate A

```

SSL_connect:SSLv3 read server key exchange A
SSL_connect:SSLv3 read server done A
SSL_connect:SSLv3 write client key exchange A
SSL_connect:SSLv3 write change cipher spec A
SSL_connect:SSLv3 write finished A
SSL_connect:SSLv3 flush data
SSL_connect:SSLv3 read server session ticket A
SSL_connect:SSLv3 read finished A
SSL3 alert read:warning:close notify
SSL3 alert write:warning:close notify

```

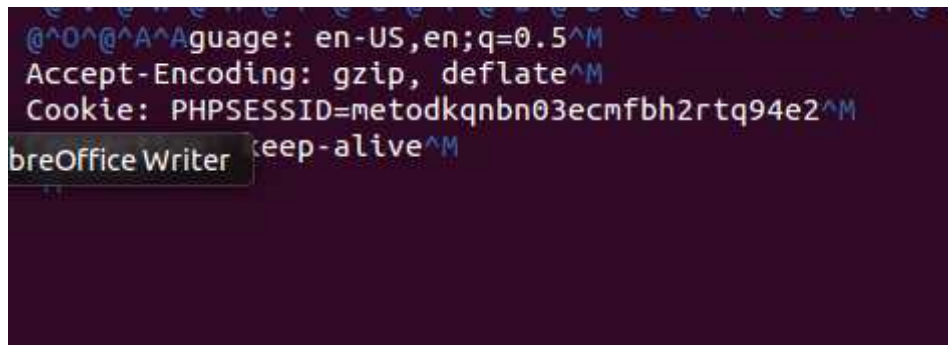
HeartBeat 라는 문자열이 검색된다면 HeartBeat 프로토콜이 활성화 되어 있는걸 확인 할 수 있습니다.

GitHub에 공개되어 있는 CVE-2014-0160 POC 코드를 이용하여 서버에 테스트를 합니다.

```

root@Tempus:/home/tempus/바탕화면/CVE-2014-0160-master#
python heartbleed-poc.py -n10 -f dump 192.168.106.129

```



위처럼 서버의 메모리내에 있는 정보가 유출되고 있다는걸 확인할 수 있습니다.

7. 대응방안

HeartBleed 취약점을 방지하려면 OpenSSL을 1.0.1g 이상 버전으로 업데이트를 해줘야 합니다.

Centos/Fedora 계열

```
yum update  
sudo pacman -Syu
```

Ubuntu 계열

```
sudo apt-get update  
sudo apt-get dist-upgrade  
sudo apt-get install --only-upgrade openssl  
sudo apt-get install --only-upgrade libssl1.0.0
```