BEAST TITLE TO-FILL

Yang Shichu
School of Cyber Science and Engineering
Huazhong University of Science and Technology
sigeryeung@gmail.com

Shu Yi
TO-FILL
TO-FILL
TO-FILL

Su Haochen
TO-FILL
TO-FILL
TO-FILL

Li Yucong
TO-FILL
TO-FILL
TO-FILL

Liao Haicheng
TO-FILL
TO-FILL
TO-FILL

ABSTRACT

Transport Layer Security (TLS) is an protocol that provides communication security over networks. However, there is a flaw in TLS 1.0 where the initial vectors for block ciphers are predictable. The BEAST attack, with some prerequisites and efforts, allows attackers in the middle to decrypt those encrypted messages without knowing the key. This paper will demonstrate the procedures of the BEAST attack, and propose methods in simulation and vulnerability detection.

Keywords

BEAST attack, TLS flaws, CBC exploits, vulnerability detection

1. INTRODUCTION

Transport Layer Security (TLS) has several versions. The specification for TLS 1.0 is RFC 2246[1].

2. BACKGROUND

2.1 A glance at TLS

TLS protocols

2.2 CBC in block ciphers

CBC is one of the modes of operation used in block ciphers.

Supposing that $P_1, P_2, \cdots P_n$ are the plaintext blocks, with a initial vector IV, we have:

$$C_1 = E_k(P_1 \oplus IV)$$
$$C_i = E_k(P_i \oplus C_{i-1})(i \geq 2)$$

to obtain ciphertext blocks $C_1, C_2, \cdots, C_n$.

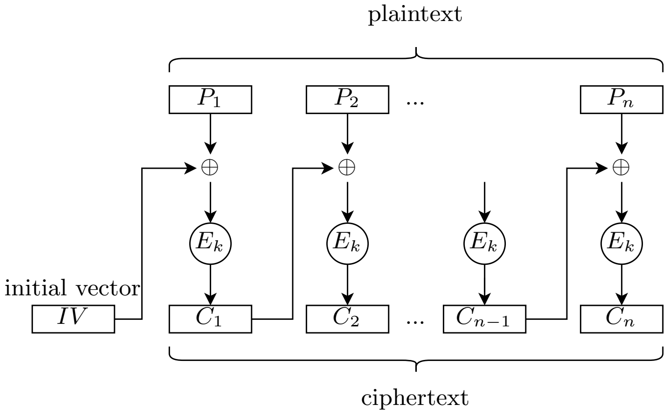


Figure 1: CBC encryptor

3. THREAT MODEL

4. DEMONSTRATION

5. FEASIBILITY AND DEFENSE

5.1 Feasibility

While BEAST attacks are theoretically feasible, with the enhancement of security features of browsers and other clients, BEAST attacks are less and less practical for an attacker to exploit.

5.2

5.3 Defense

BEAST attacks make use of a flaw in the specification of TLS 1.0, and it only works for block ciphers. That is to say, stream ciphers with TLS 1.0 are not vulnerable to BEAST attacks.

However, stream ciphers (e.g. RC4) with TLS 1.0 is still vulnerable to other attacks. Therefore, a much more direct way is just to abandon TLS 1.0, and update to later TLS versions.

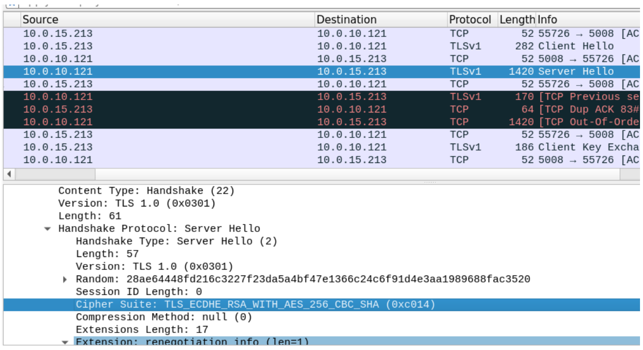
Many modern browsers and clients have also limited users to browse those sites with only TLS 1.0 enabled. This kind of action will boost organizations to update their websites TLS versions. Today there are only few sites supporting TLS 1.0.

6. DETECTION

We will propose a method to detect BEAST vulnerability of a server, together with a Python script which is easy to use.

At the stage of TLS handshake, a common cipher suite will be selected through steps:

1. (Client Hello) Client sent a list of accepted cipher suites.
2. (Server Hello) Server chose a best accepted cipher suite, or a handshake failure occured.



Source	Destination	Protocol	Length	Info
10.0.15.213	10.0.10.121	TCP	52	55726 → 5008 [AC
10.0.15.213	10.0.10.121	TLSv1	282	Client Hello
10.0.10.121	10.0.15.213	TCP	52	5008 → 55726 [AC
10.0.10.121	10.0.15.213	TLSv1	1420	Server Hello
10.0.15.213	10.0.10.121	TCP	52	55726 → 5008 [AC
10.0.10.121	10.0.15.213	TLSv1	170	[TCP Previous se
10.0.15.213	10.0.10.121	TCP	64	[TCP Dup ACK 83#
10.0.10.121	10.0.15.213	TCP	1420	[TCP Out-Of-Orde
10.0.15.213	10.0.10.121	TCP	52	55726 → 5008 [AC
10.0.15.213	10.0.10.121	TLSv1	186	Client Key Excha
10.0.10.121	10.0.15.213	TCP	52	5008 → 55726 [AC

```
    Content Type: Handshake (22)
    Version: TLS 1.0 (0x0301)
    Length: 61
  ▾ Handshake Protocol: Server Hello
        Handshake Type: Server Hello (2)
        Length: 57
        Version: TLS 1.0 (0x0301)
      ▸ Random: 28ae64448fd216c3227f23da5a4bf47e1366c24c6f91d4e3aa1989688fac3520
        Session ID Length: 0
        Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014)
        Compression Method: null (0)
        Extensions Length: 17
      ▾ Extension: renegotiation_info (len=1)
```

Figure 2: Negotiation on the cipher suite

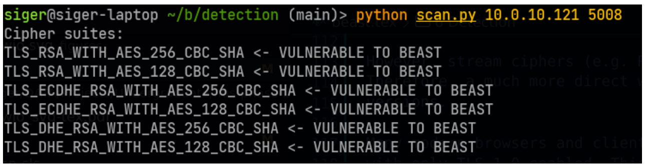
Based on this, a scanner could change the list of cipher suites to enumerate all cipher suites that the server will accept.

The server is vulnerable to BEAST attacks if it accepts TLS 1.0 handshake and support cipher suites with CBC modes.

```
python scan.py host port
```

The `openssl` utility is able to start a TLS server with many options.

```
openssl s_server \
    -CAfile ca_cert.pem \
    -cert server_cert.pem \
    -key server_key.pem \
    -HTTP -port 5008 -tls1
```



```
siger@siger-laptop ~/b/detection (main)> python scan.py 10.0.10.121 5008
Cipher suites:
TLS_RSA_WITH_AES_256_CBC_SHA <- VULNERABLE TO BEAST
TLS_RSA_WITH_AES_128_CBC_SHA <- VULNERABLE TO BEAST
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA <- VULNERABLE TO BEAST
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA <- VULNERABLE TO BEAST
TLS_DHE_RSA_WITH_AES_256_CBC_SHA <- VULNERABLE TO BEAST
TLS_DHE_RSA_WITH_AES_128_CBC_SHA <- VULNERABLE TO BEAST
```

Figure 3: Detection output on a TLS 1.0 server

7. REFERENCES

[1] C. Allen and T. Dierks. The TLS Protocol Version 1.0. RFC 2246, Jan. 1999.

APPENDIX