"But, after all," he reflected before turning into her room,
"what has occurred? Nothing. She had a long conversation
with him. Well, what harm is there in that?"

... and as soon as he re-entered the dark drawing room
a voice would whisper that it was not so, and that if
others noticed, that showed there was something to notice.

- Tolstoy
Anna Karenina

from Charles Rackoff, Daniel R. Simon, "Cryptographic Defense Against Traffic Analysis" (1993)

SSH Traffic Analysis Attacks

Solar Designer
<solar@openwall.com>

Dug Song
<dugsong@monkey.org>

Introduction: SSH

- SSH: Secure Shell
- Developed by Tatu Ylonen
- Secure remote login, Berkeley r-command replacement
- Provides authentication, encryption, message integrity

Introduction: Traffic analysis

- Yin Zhang, Vern Paxson. "Detecting Backdoors" (2000)
  - Identification of interactive traffic via passive network monitoring
  - Traffic contents, sizes, timing structure, directionality

SSH vulnerabilities

- Packet sizes during initial login reveal
  - authentication methods
  - username and password lengths
  - number of RSA authorized_keys options
  - successes, failures, and refusals

SSH vulnerabilities (cont.)

- Echo processing easily discriminates password entry
- Inter-packet timing and packet sizes allow for inference of keystrokes and commands
- Discovery of password length narrows search space for dictionary attack

SSH-1

- Specified in draft-ylonen-ssh-protocol-00.txt

length	padding (8 - (length % 8))	type
payload (length - 5)	crc32	

- Length field sent in the clear
- Padding, type, data encrypted
- Padded to 8-byte boundary

SSH-2

- Specified in IETF secsh working group I-Ds

length	plen	payload (length - plen - 1)
random padding (plen)	MAC	

- Lengths, data, and padding encrypted
- Total length (data+padding) must be a multiple of the cipher blocksize
- As implemented, only padded to next blocksize boundary

SSHOW - Overview

- Monitor concurrent SSH-1 and SSH-2 sessions
- Identify successful, failed, and refused RSA, DSA, password authentication attempts
- Identify username, password, command lengths
- Print payload sizes and inter-arrival times

SSHOW - Design

- Stateful, passive network monitor for SSH-1 and SSH-2
- Maintain session packet history, including timing and directionality
- Identify events based on simple signatures and packet history

SSHOW - Implementation

- libnids provides TCP session tracking and per-connection stateful callbacks
- SSH-1 application layer packet reassembly
- Client-to-server and server-to-client callbacks accumulate session history
- Debugging mode prints summary info per reassembled SSH packet

SSHOW - Implementation (cont.)

- Initial login
  - Success, failure, refusal
  - RSA / DSA authentication methods and options
  - Username and password length
- Interactive session analysis
  - Shell command length
  - Password length (e.g. 'su', 'enable')

SSHOW - Implementation (cont.)

- Per-packet state

```
typedef struct {
        int direction;                  /* 0 for client to server */
        clock_t timestamp;              /* timestamp of this packet */
        u_int cipher_size;              /* ciphertext size */
        range plain_range;              /* possible plaintext sizes */
} record;
```

- Per-session state

```
struct history {
        record packets[HISTORY_SIZE]; /* recent packets (circular list) */
        int index;                    /* next (free) index into packets[] */
        u_int directions;             /* recent directions (bitmask) */
        clock_t timestamps[2];        /* last timestamps in each direction */
};
```

SSHOW - Implementation (cont.)

- Username length

```
if (session->state == 0 && session->protocol == 1 &&
    ((session->history.directions >> skip) & 7) == 5 &&
    plain_range->min == 0 &&
    get_history(session, skip + 1)->plain_range.min > 4 &&
    get_history(session, skip + 2)->plain_range.min == 0)
```

- Output:

```
+ 217.155.34.193:24998 -> 204.181.64.8:22: SSH protocol 1
- 217.155.34.193:24998 <- 204.181.64.8:22: DATA (262 bytes, 0.00 seconds)
- 217.155.34.193:24998 -> 204.181.64.8:22: DATA (143 bytes, 0.00 seconds)
- 217.155.34.193:24998 <- 204.181.64.8:22: DATA (0 bytes, 0.00 seconds)
- 217.155.34.193:24998 -> 204.181.64.8:22: DATA (11 bytes, 0.00 seconds)
- 217.155.34.193:24998 <- 204.181.64.8:22: DATA (0 bytes, 0.00 seconds)
+ 217.155.34.193:24998 -> 204.181.64.8:22: GUESS: Username length is 7
```

SSHOW - Implementation (cont.)

- RSA authentication success (w/options) or failure

```
if (session->state == 1 && session->protocol == 1 && skip >= 1 &&
    ((session->history.directions >> (skip - 1)) & 037) == 013 &&
    plain_range->min == 0 &&
    get_history(session, skip - 1 + 2)->plain_range.min == 16 &&
    get_history(session, skip - 1 + 3)->plain_range.min == 130 &&
    get_history(session, skip - 1 + 4)->plain_range.min == 130) {
...
      switch (get_history(session, 1)->plain_range.min - 4) {
      case 28:
            /* "RSA authentication accepted." */
            if (skip > 1)
                  /* "skip - 1" is the number of authorized_keys options */
...
      case 47:
            /* "Wrong response to RSA authentication challenge." */
```

SSHOW - Implementation (cont.)

- Output of RSA authentication success with authorized_keys options:

```
- 217.155.34.172:1048 -> 204.181.64.8:22: DATA (130 bytes, 1.01 seconds)
- 217.155.34.172:1048 <- 204.181.64.8:22: DATA (130 bytes, 0.00 seconds)
- 217.155.34.172:1048 -> 204.181.64.8:22: DATA (16 bytes, 0.00 seconds)
- 217.155.34.172:1048 <- 204.181.64.8:22: DATA (29 bytes, 0.00 seconds)
- 217.155.34.172:1048 <- 204.181.64.8:22: DATA (28 bytes, 0.00 seconds)
- 217.155.34.172:1048 <- 204.181.64.8:22: DATA (30 bytes, 0.00 seconds)
- 217.155.34.172:1048 <- 204.181.64.8:22: DATA (28 bytes, 0.01 seconds)
- 217.155.34.172:1048 <- 204.181.64.8:22: DATA (29 bytes, 0.00 seconds)
- 217.155.34.172:1048 <- 204.181.64.8:22: DATA (32 bytes, 0.00 seconds)
- 217.155.34.172:1048 <- 204.181.64.8:22: DATA (0 bytes, 0.00 seconds)
+ 217.155.34.172:1048 -> 204.181.64.8:22: GUESS: RSA authentication accepted (5+
authorized_keys options)
```

SSHOW - Implementation (cont.)

- RSA authentication refused

```
if (session->state == 1 && session->protocol == 1 &&
    (session->history.directions & 3) == 1 && plain_range->min == 0 &&
    get_history(session, 1)->plain_range.min == 130)
```

- Output:

```
- 127.0.0.1:40190 -> 127.0.0.1:22: DATA (130 bytes, 0.00 seconds)
- 127.0.0.1:40190 <- 127.0.0.1:22: DATA (0 bytes, 0.00 seconds)
+ 127.0.0.1:40190 -> 127.0.0.1:22: GUESS: RSA authentication refused
```

SSHOW - Implementation (cont.)

- Login password length

```
if (session->state == 1 &&
    now - get_history(session, 2)->timestamp >= CLK_TCK &&
    session->protocol == 1 &&
    (session->history.directions & 7) == 5 && plain_range->min == 0 &&
    get_history(session, 1)->plain_range.min > 4 &&
    get_history(session, 2)->plain_range.min == 0)
```

- Output:

```
- 127.0.0.1:41264 -> 127.0.0.1:2022: DATA (17 bytes, 10.16 seconds)
- 127.0.0.1:41264 <- 127.0.0.1:2022: DATA (0 bytes, 10.16 seconds)
+ 127.0.0.1:41264 -> 127.0.0.1:2022: GUESS: Password authentication, password length
is 13
```

SSHOW - Implementation (cont.)

- Interactive session tracking

```
struct line {
        int input_count;                /* input packets (client to server) */
        int input_size;                 /* input size (estimated) */
        int input_last;                 /* last input packet size */
        int echo_count;                 /* echo packets (server to client) */
};
```

```
if (session->state == 2) {
...
        /* Check for backspace */
        if (session->protocol == 1 && !session->compressed &&
            plain_range->min == 4 + 3 &&
            session->line.input_size >= 2)
                session->line.input_size -= 2;
```

SSHOW - Implementation (cont.)

- Command / password length

```
if (session->state == 2) {
...
        if (plain_range->min > 4 + session->line.input_last &&
            session->line.input_count >= 2 &&
            session->line.input_size >= 2) {
...
                size = session->line.input_size;
                if (session->line.echo_count + 1 >= session->line.input_count &&
                    size <= (session->line.input_count << 2) && size < 0x100)
                        /* command */
    ...
                else if (session->line.echo_count <= 2 &&
                    size <= (session->line.input_count << 1) &&
                    size >= 2 + 1 && size <= 40 + 1)
                        /* password */
```

SSHOW - Future work

- X keyboard / mouse event detection
- Shell command identification via timing signatures
- Application to other encrypted protocols
  - Kerberized telnet/rsh
  - SSL telnet
  - Telnet over IPSEC

Related work

- Dawn Song, David Wagner, Xuqing Tian. "Timing Analysis of Keystrokes and Timing Attacks on SSH"

  http://www.cs.berkeley.edu/~dawnsong/ssh-timing.html

  - Hidden Markov Model + key sequence inference algorithm
  - Inter-keystroke timings leak 1 bit per character pair

- Yang Yu. "SSH Traffic Analysis Preliminary Report"

Proposed fixes

- For SSH-1, pad username, password with NULs
  - assumes C strings at the server end - not a part of the protocol
- From Simon Tatham: hide real password message among multiple SSH_MSG_IGNORE messages of increasing sizes
  - about 1 KB overhead to hide passwords up to 32 characters

Proposed fixes (cont.)

- Use SSH_MSG_IGNORE to simulate input echoing during password entry
- Use padding capability for SSH-2

Vendor response

- OpenSSH fixes included since version 2.5.2
- PuTTY fixes to appear in version 0.52
- TTSSH fixes included since version 1.5.4
- Cisco IOS and CatOS fixes included in recent versions
- Unofficial patches for ssh-1.2.x included in original advisory

Conclusions

- Difficult to strike a balance between security and usability
- Perfect resistance to traffic analysis requires complete uniformity
  - detectable entropy (size changes due to compression, etc.)
  - timing
  - directionality

Further information

- Updated Openwall advisory available from

  http://www.openwall.com/advisories/

- SSHOW included in the dsniff toolkit

  http://www.monkey.org/~dugsong/dsniff/

- Any questions?