An in-depth look at Kippo: an integration perspective

Kamil Kołtyś
Network and Information Security Methods Team

Contents

1 Introduction

Kippo is a honeypot that emulates the SSH service. The emulation involves SSH session establishment and, in the case of user's successful authorization, also shell interaction with the user. The honeypot can be used to register brute force attacks aimed at obtaining users SSH service passwords. Moreover, it allows for analyzing further actions of the attacker who manages to obtain the password and thus has access to the emulated system. This is possible because Kippo registers all shell commands inputted by the user and saves in selected directory all files downloaded by means of emulated *wget* command.

Kippo emulates SSH service so a degree that may give an attacker the impression of using real SSH service. Thus, data gathered by the honeypot may provide many valuable information about the purpose and the way of performing such type of attacks. These data may be useful for many different IT security systems. The aim of this article is to present the Kippo honeypot from the perspective of integration with other threat detection systems, which may benefit from Kippo's functionality.

The first sections of this document present issues involved with installation, configuration, start up and reconfiguration of the honeypot. Afterward, the data gathered by Kippo are characterized and the details about data format are provided.

2 Installation

To install Kippo, the following items from SVN repository `http://kippo.googlecode.com/svn/trunk/` have to be copied to the directory selected as working directory for Kippo:

- *kippo* - directory with Kippo source files (python);
- *kippo.tac* - configuration file for *twistd* (*twistd* is required to run Kippo);
- *start.sh* - shell script for starting Kippo;

3 Configuration

3.1 Main configuration file

The main configuration file is a text file with INI-like structure [1]. This format is supported by e.g. *ConfigParser* python module. An example of the main configuration file can be found in the Kippo project repository: `http://kippo.googlecode.com/svn/trunk/kippo.cfg.dist`.

Kippo assumes that the main configuration file resides in Kippo's working directory and its name is *kippo.cfg*. Changing default path to the configuration file is not possible without altering the Kippo sources.

The main configuration file has three sections:

- [honeypot] - mandatory section including core honeypot configuration;
- [database_mysql] - optional section including configuration of the MySQL logging module;
- [database_xmpp] - optional section including configuration of the XMPP logging module;

All parameters of these sections are discussed below.

Section [honeypot]

The following parameters can be defined in the [honeypot] section:

- ssh_addr - IP address on which Kippo listens for new connections (default is 0.0.0.0 that means any address IP);
- ssh_port - TCP port on which Kippo listens for new connections (default is 2222);
- hostname - hostname displayed by shell prompt;
- log_path - directory for saving log files (default is *log*; this directory must have subdirectory *tty*, in which files with SSH session history are saved; when a directory other than default one is used, the option -l specified for twistd in script *start.sh* should be appropriately changed - see 3.2);
- download_path - directory for saving files downloaded by the emulated *wget* command;

[1] A hash (#) is used to start single line comment instead of a semicolon (;).

- download_limit_size - maximum size of downloaded file (default is 0 that means no limit; if maximum size greater than 0 is specified, files larger than this maximum size will not be saved on disk);
- filesystem_file - path to the file containing emulated filesystem (see 3.3);
- contents_path - directory with files contents displayed by emulated *cat* command (see 3.4);
- txtcmds_path - directory with messages generated by executable files (see 3.5);
- data_path - directory including files with additional configuration data (see 3.6);
- public_key - path to the file with public key used to authenticate emulated SSH server;
- private_key - path to the file with private key used to authenticate emulated SSH server;
- out_addr - IP address used by emulated *wget* for opening outgoing connections (optional parameter);
- sensor_name - name identifying the instance of Kippo in MySQL database (optional parameter, if it is not specified, the IP address is used to identify the Kippo instance);
- fake_addr - fake IP address displayed as a source IP address of SSH connection by emulated *w* command (optional parameter, if it is not specified, the emulated *w* displays actual IP address);
- banner_file - file including message that is displayed before the first prompt for password (optional parameter);
- interact_enabled - flag (true or false) indicating if interactive access to Kippo using telnet is enabled (by default is disabled);
- interact_port - port, on which Kippo handles telnet communication;

Section [database_mysql]

The following parameters can be defined in the [database_mysql] section:

- host - name of the server running database;
- database - database name;
- username - name of the database user;
- password - password of the database user;
- port - port on which the database listens for new connections;

Section [database_xmpp]

The following parameters can be defined in the [database_xmpp] section:

- server - name of the XMPP server;
- user - name of the XMPP user;
- password - password of the XMPP user;
- muc - name of the MUC service;
- signal_createsession - name of the MUC room to which messages about creating new TCP connection/SSH session are sent;
- signal_connectionlost - name of the MUC room to which messages about termination of TCP connection/SSH session are sent;
- signal_loginfailed - name of the MUC room to which messages about failed authentication attempt are sent;
- signal_loginsucceeded - name of the MUC room to which messages about successful authentication attempt are sent;
- signal_command - name of MUC room to which messages about executing shell command are sent;
- signal_clientversion - name of MUC room to which messages about SSH client version are sent;
- debug - flag (true or false) indicating if additional information concerning communication with XMPP server has to be logged in the main log file;

3.2 Options for twistd

The following options for *twistd* can be set in the *start.sh* shell script:

- option -l: path to the file in which Kippo log entries are recorded (default is *log/kippo.log*);
- option --pidfile: path to the file in which PID of process executing Kippo is recorded (default is *kippo.pid*);

3.3 File with emulated filesystem

Kippo requires a file (in python pickle format) that contains the structure of emulated filesystem. In the Kippo project repository there is an example file with filesystem resembling Debian 5.0 (`http://kippo.googlecode.com/svn/trunk/fs.pickle`). However, using tool *createfs.py* from the Kippo repository (`http://kippo.googlecode.com/svn/trunk/createfs.py`) it is possible to generate a fake filesystem (in python pickle format required by Kippo) which structure is based on the host filesystem.

3.4 File contents

File with emulated filesystem contains only the structure of the emulated filesystem and does not include the files contents. Thus, emulated command *cat* by default displays message No such file or directory for each file, even if this file exists in the structure of emulated filesystem (it is displayed by emulated command *ls*).

However, for each file in the emulated filesystem it is possible to define its content that will be displayed by emulated *cat*. Let's consider the file file_name located on the emulated filesystem in directory file_dir. In order to define the content of this file, a file file_name with relevant content has to be created on the host filesystem in the directory contents_path/file_dir (see contents_path parameter description in 3.1).

In the Kippo project repository a directory `http://kippo.googlecode.com/svn/trunk/honeyfs` comprises example content for selected files of Debian 5.0 system.

3.5 Executable file messages

In the system emulated by Kippo an attempt to run any executable file by default causes showing an error message command not found.

However, for each executable file in the emulated filesystem it is possible to define the message that will be displayed as a result of its execution. To do so, for given executable file file_name located on the emulated filesystem in directory file_dir, the file containing relevant message and having name file_name has to be created on the host filesystem in the directory txtcmds_path/file_dir (see txtcmds_path parameter description in 3.1).

In the Kippo project repository a directory `http://kippo.googlecode.com/svn/trunk/txtcmds` comprises example messages for selected executable files of Debian 5.0 system.

3.6 Additional configuration data

Additional configuration data are included in two text files located in directory specified in the main configuration file by the parameter data_path:

- *userdb.txt* - text file containing in each row a trio login:uid:password; file *userdb.txt* is used for:
  - user authentication: authentication relies on checking whether for given login and password there exists a trio in file *userdb.txt* containing this login and password; file *userdb.txt* is automatically modified by the emulated *passwd* command that causes appending a trio cur_user:cur_uid:new_pass where cur_user is a name of the user that executes the command, cur_uid is the uid of this user and new_pass is the new password;
  - diplaying uid by emulated *id* command: *id* displays uid from the first trio in which the login of user invoking the command is specified;

  in the Kippo project repository there is an example of the file *userdb.txt*: `http://kippo.googlecode.com/svn/trunk/data/userdb.txt`
- *last.log* - text file containing information about last logins to the system emulated by Kippo; after the termination of authenticated SSH session Kippo appends to the file *last.log* information about terminated session (registered user name is always root even if actual name of user that was logged in is different);

4 Start up and reconfiguration

To start up Kippo the script *start.sh* has to be executed.

To reconfigure the honeypot selected configuration elements mentioned in subsections 3.1-3.6 have to be appropriately altered. Changes made in the main configuration file, in the script *start.sh* and/or in the file with emulated filesystem (i.e. changes concerning configuration elements discussed in subsections 3.1-3.3) require honeypot restart. In turn, changes involving files contents, executable files messages or additional configuration data (i.e. changes concerning configuration elements discussed in subsections 3.4-3.6) are immediately taken into account by running Kippo instance.

5 Output characteristics

Kippo gathers and stores data about different kind of events occurring during TCP connections/SSH sessions. All those data are registered in the text file serving as a main log of honeypot. Additionally the history of each SSH session is recorded in separate binary file. Optionally a part of data gathered by Kippo can be stored in MySQL database and/or communicated to XMPP server. However, it should be noted that in the MySQL database or in the XMPP server not all output data can be found that are available in the main log. For each TCP connection/SSH session following data can be found in main log (L), database (B) and XMPP server (X) about different events:

- establishment of TCP connection:
  - source IP address: L, B, X;
  - source TCP port: L, X;
  - target IP address: L, X;
  - target TCP port: L, X;
  - time of establishment of TCP connection: L, B;
- termination of TCP connection/SSH session:
  - reason of termination of TCP connection/SSH session: L;
  - time of termination of TCP connection/SSH session: L, B*, X*;
  - in the case of termination of SSH session:
    - content of the file with SSH session history: B*;
- receiving of information about SSH client version:
  - client version: L, B*, X*;

  - time of receiving of information about client version: L;

- establishment of encrypted connection:
  - encryption algorithm: L;
  - time of establishment of encrypted connection: L;
- client's authentication attempt:
  - login: L, B, X;
  - password: L, B, X;
  - information if attempt was successful: L, B, X;
  - time of performing the client's authentication attempt: L, B;
  - in the case of successful authentication (i.e. creating SSH session):
    * size of terminal: L, B;
    * name of created file to record SSH session history: L;
    * values of environment variables (e.g. XMODIFIERS, LANG): L;
- executing shell command:
  - command name: L, B, X;
  - information if command was recognized by Kippo: L, B, X;
  - time of command execution: L, B;
  - in the case of *wget* command:
    * URL of downloaded file: L, B;
    * name of the file where downloaded file was saved on the host machine: L, B;
    * time of starting the file download: L, B;
    * time of finishing the file download: L;
- inputting data for shell command (e.g. *passwd*):
  - command name: L, B;
  - input data: L, B;
  - time of data input: L, B;

At the time of writing the newest Kippo revision (234) fails to communicate some output to MySQL database (B*) and XMPP server (X*) although the earlier revision (228 [2]) does it well.

[2]Revision 228, in turn, does not store information about downloaded files in the MySQL database

6 Output format

6.1 Main log

Kippo records all gathered data about current events in the text file *kippo.log* created by default in subdirectory *log* of Kippo's working directory. Name and location of this file can be changed in the script *start.sh* (see 3.2). When the size of *kippo.log* exceeds 1MB, its contents is moved to the file *kippo.log.1*, old content of the *kippo.log.1* (if it exists) is moved to the *kippo.log.2*, old content of the *kippo.log.2* (if it exists) is moved to the *kippo.log.3* and so on. Thus, Kippo main log consists of one text file, in which data about current events are recorded and zero or more text files with data about previous events.

Each entry of the main log consists of three consecutive parts separated by single spaces:

- timestamp - log entry creation time;
- context - log entry context;
- message - log entry message (may consists of multiple lines);

The log entry creation time is the local time reported by the operating system. The creation time is recorded in the following format: YYYY-MM-DD HH:mm:ssTZD. The format of log entry's context and message depends on the event type and is discussed in more detail below.

Establishment of TCP connection

Establishment of TCP connection is registered in the main log as a one entry with following context and message:

- context := [kippo.core.honeypot.HoneyPotSSHFactory];
- message := New connection: *IP_source*:*port_source* (*IP_dest*:*port_dest*) [session: *session_id*], where:
  - *IP_source*: source IP address;
  - *port_source*: source TCP port;
  - *IP_dest*: target IP address;
  - *port_dest*: target TCP port;
  - *session_id*: number uniquely identifying TCP connection/SSH session in the main log (session identifier);

Example log entry informing about the establishment of TCP connection:

```
2013−01−08 14:18:25+0100 [kippo.core.honeypot.HoneyPotSSHFactory]
    New connection: 192.168.122.1:35533 (192.168.122.82:2222) [session: 0]
```

Termination of TCP connection/SSH session

Termination of TCP connection/SSH session is registered in main log as one or two entries with following context:

- context := [HoneyPotTransport,*session_id*,*IP_source*], where:
  - *session_id*: identifier of the terminated session ;
  - *IP_source*: source IP address of the terminated session;

Entry informing about the event of the termination of TCP connection/SSH session has the following message:

- message := connection lost;

Before above mentioned entry there may be an entry informing about the reason of termination of TCP connection/SSH session. The message of this entry depends on the kind of the reason and may be one of the following:

- message := Disconnecting with error, code 2
  reason: bad packet length *length_value*
- message := Got remote error, code 11
  reason: disconnected by user

Both above messages occupy two lines. The former one denotes the connection termination due to receiving packet with bad length *length_value*. The later one informs that the connection has been terminated by the client.

Example log entries informing about the termination of TCP connection/SSH session:

```
2013−01−08 15:03:22+0100 [HoneyPotTransport,0,192.168.122.1] Got
    remote error, code 11
reason: disconnected by user
2013−01−08 15:03:22+0100 [HoneyPotTransport,0,192.168.122.1]
    connection lost
```

Receiving of information about SSH client version

Receiving of information about SSH client version is registered in the main log as a one entry with following context and message:

- context := [HoneyPotTransport,*session_id*,*IP_source*], where:
  - session_id: identifier of the session involved with received information about SSH client version;
  - IP_source: source IP address of the session involved with received information about SSH client version;
- message := Remote SSH version: *version_string*, where:
  - *version_string*: string describing SSH client version;

Example log entry informing about SSH client version:

```
2013−01−08 14:18:26+0100 [HoneyPotTransport,0,192.168.122.1] Remote
    SSH version: SSH−2.0−OpenSSH_5.3
```

Establishment of encrypted connection

Establishment of encrypted connection is registered in the main log as five entries with following context:

- context := [HoneyPotTransport,*session_id*,*IP_source*], where:
  - *session_id*: identifier of the session for which the encrypted connection has been established;
  - *IP_source*: source IP address of the session for which the encrypted connection has been established;

Among these five log entries, two of them with following messages are significant:

- message := incoming: *cipher_spec*, where:
  - *cipher_spec*: encryption algorithm;
- message := starting service ssh-userauth;

The first message informs about the algorithm used for encrypting the connection. The second one states the fact of establishment of encrypted connection.

Example log entries informing about the establishment of encrypted connection:

```
2013−01−08 14:18:26+0100 [HoneyPotTransport,0,192.168.122.1] kex alg,
    key alg: diffie−hellman−group1−sha1 ssh−rsa
2013−01−08 14:18:26+0100 [HoneyPotTransport,0,192.168.122.1] outgoing:
     aes128−ctr hmac−md5 none
2013−01−08 14:18:26+0100 [HoneyPotTransport,0,192.168.122.1] incoming
    : aes128−ctr hmac−md5 none
2013−01−08 14:18:26+0100 [HoneyPotTransport,0,192.168.122.1] NEW
    KEYS
2013−01−08 14:18:26+0100 [HoneyPotTransport,0,192.168.122.1] starting
    service ssh−userauth
```

Client's authentication attempt

Client's authentication attempt is registered in the main log as several entries with following context:

- context := [SSHService ssh-userauth on HoneyPotTransport,*session_id*,*IP_source*], where:
  - *session_id*: identifier of the session for which the authentication is performed;
  - *IP_source*: source IP address of the session for which the authentication is performed;

Among these several log entries, the one with following messages is significant:

- message := login attempt [*username*/*password*] *result*, where:
  - username: name of the user;
  - password: password of the user;
  - result: result of the client's authentication attempt (succeeded or failed);

In the case of successful authentication attempt (result = succeeded) in the main log there are registered subsequent entries with following context:

- context := [SSHChannel session (0) on SSHService ssh-connection on HoneyPotTransport,*session_id*,*IP_source*], where:
  - *session_id*: identifier of the session for which the authentication was successful;
  - *IP_source*: source IP address of the session for which the authentication was successful;

These log entries inform about the size of terminal, the name of created file to store SSH session history and the values of some environment variables, respectively. Their messages are as follows:

- message := Terminal size: *row_num col_num*, where:
  - *row_num*: number of rows;
  - *col_num*: number of columns;
- message := Opening TTY log: *file_name*, where:
  - *file_name*: name of created file to store SSH session history;
- message := request_env: '\x00\x00\x00\nXMODIFIERS\x00\x00\x00\x08*var_value*', where:
  - *var_value*: value of XMODIFIERS variable ;
- message := request_env: '\x00\x00\x00\x04LANG\x00\x00\x00\n*var_value*', where:
  - *var_value*: value of LANG variable;

Example log entries informing about client's authentication attempt:

```
2013-01-08 14:33:27+0100 [SSHService ssh-userauth on
    HoneyPotTransport,0,192.168.122.1] login attempt [root/123456]
    succeeded
2013-01-08 14:33:27+0100 [SSHService ssh-userauth on
    HoneyPotTransport,0,192.168.122.1] root authenticated with keyboard
    -interactive
2013-01-08 14:33:27+0100 [SSHService ssh-userauth on
    HoneyPotTransport,0,192.168.122.1] starting service ssh-connection
2013-01-08 14:33:27+0100 [SSHService ssh-connection on
    HoneyPotTransport,0,192.168.122.1] got channel session request
2013-01-08 14:33:27+0100 [SSHChannel session (0) on SSHService ssh-
    connection on HoneyPotTransport,0,192.168.122.1] channel open
```

```
2013−01−08 14:33:27+0100 [SSHService ssh−connection on
    HoneyPotTransport,0,192.168.122.1] got global no−more−
    sessions@openssh.com request
2013−01−08 14:33:27+0100 [SSHChannel session (0) on SSHService ssh−
    connection on HoneyPotTransport,0,192.168.122.1] pty request: xterm
    (33, 129, 0, 0)
2013−01−08 14:33:27+0100 [SSHChannel session (0) on SSHService ssh−
    connection on HoneyPotTransport,0,192.168.122.1] Terminal size: 33
    129
2013−01−08 14:33:27+0100 [SSHChannel session (0) on SSHService ssh−
    connection on HoneyPotTransport,0,192.168.122.1] request_env: '\x00\
    x00\x00\nXMODIFIERS\x00\x00\x00\x08@im=none'
2013−01−08 14:33:27+0100 [SSHChannel session (0) on SSHService ssh−
    connection on HoneyPotTransport,0,192.168.122.1] request_env: '\x00\
    x00\x00\x04LANG\x00\x00\x00\npl_PL.utf8'
2013−01−08 14:33:27+0100 [SSHChannel session (0) on SSHService ssh−
    connection on HoneyPotTransport,0,192.168.122.1] getting shell
2013−01−08 14:33:27+0100 [SSHChannel session (0) on SSHService ssh−
    connection on HoneyPotTransport,0,192.168.122.1] Opening TTY log:
    /var/log/kippo/log/tty/20130108−143327−9152.log
2013−01−08 14:33:33+0100 [SSHChannel session (0) on SSHService ssh−
    connection on HoneyPotTransport,0,192.168.122.1] /etc/motd resolved
    into /etc/motd
```

Executing shell command

Executing shell command is registered in the main log as two entries with following context:

- context := [SSHChannel session (0) on SSHService ssh-connection on HoneyPotTransport,*session_id*,*IP_source*], where:
  - *session_id*: identifier of the session within the shell command is executed;
  - *IP_source*: source IP address of the session within the shell command is executed;

From these two entries the second one with following message is significant:

- message := *cmd_found* : *cmd_name*, where:

- *cmd_found*: information if command was recognized by Kippo (Command found means yes and Command not found means no);
- *cmd_name*: name of the shell command;

In the case of *wget* command the subsequent log entries inform about the URL of downloaded file, the name of the file where downloaded file is saved on the host machine and the time of starting and finishing the file download. These log entries have following context:

- context := [HTTPPageDownloader,client];

They contain the following messages:

- message := Updating realfile to *file_name*, where:
  - *file_name*: name of the file where downloaded file is saved on the host machine;
- message := Stopping factory <HTTPProgressDownloader: *url*>, where:
  - *url*: URL of downloaded file;

Example log entries informing about executing shell command:

```
2013-01-08 14:51:47+0100 [SSHChannel session (0) on SSHService ssh-
    connection on HoneyPotTransport,0,192.168.122.1] CMD: wget www.
    dna.caltech.edu/Papers/DNAorigami-nature.pdf
2013-01-08 14:51:47+0100 [SSHChannel session (0) on SSHService ssh-
    connection on HoneyPotTransport,0,192.168.122.1] Command found:
    wget www.dna.caltech.edu/Papers/DNAorigami-nature.pdf
2013-01-08 14:51:47+0100 [SSHChannel session (0) on SSHService ssh-
    connection on HoneyPotTransport,0,192.168.122.1] Starting factory <
    HTTPProgressDownloader: http://www.dna.caltech.edu/Papers/
    DNAorigami-nature.pdf>
2013-01-08 14:51:50+0100 [HTTPPageDownloader,client] Updating
    realfile to /var/log/kippo/dl/20130108145147
    _http___www_dna_caltech_edu_Papers_DNAorigami_nature_pdf
2013-01-08 14:51:50+0100 [HTTPPageDownloader,client] Stopping
    factory <HTTPProgressDownloader: http://www.dna.caltech.edu/
    Papers/DNAorigami-nature.pdf>
```

Inputting data for shell command

Inputting data for shell command is registered in the main log as a one entry with following context and message:

- context := [SSHChannel session (0) on SSHService ssh-connection on HoneyPotTransport,*session_id*,*IP_source*], where:
  - *session_id*: identifier of the session within the shell command is executed;
  - *IP_source*: source IP address of the session within the shell command is executed;
- message := INPUT (*cmd*): *input_value*, where:
  - *cmd*: name of the shell command;
  - *_value*: input data;

Example log entry informing about inputting data for shell command:

```
2013−01−08 14:51:47+0100 [SSHChannel session (0) on SSHService ssh−
    connection on HoneyPotTransport,0,192.168.122.1] INPUT (passwd):
    pp
```

6.2 MySQL database

Kippo enables storing a part of gathered data in the relational MySQL database. Fig. 1 presents Kippo's database diagram.

Seven tables are defined in Kippo's database:

- clients - table containing information about SSH client versions; this table has the following columns:
  - id: unique identifier;
  - version: SSH client version;
- sensors - table containing information about Kippo instances; this table has the following columns:
  - id: unique identifier;
  - ip: name identifying Kippo instance specified in the main configuration file (parameter sensor_name);

- sessions - table containing information about TCP connections/SSH sessions; this table has the following columns:
  - id: unique identifier;
  - starttime: time of the establishment of TCP connection/SSH session;
  - endtime: time of the termination of TCP connection/SSH session;
  - sensor: unique identifier of the Kippo instance handling TCP connection/SSH session;
  - ip: source IP address;
  - termsize: size of terminal;
  - client: unique identifier of the SSH client version;
- auth - table containing information about clients' authentication attempts; this table has the following columns:
  - id: unique identifier;
  - session: unique identifier of the session;
  - success: information if client's authentication attempt was successful (1) or failed (0);
  - username: name of the user;
  - password: password of the user;
  - timestamp: time of performing client's authentication attempt;
- input - table containing information about executed shell commands and data inputted for shell commands; this table has the following columns:
  - id: unique identifier;
  - session: unique identifier of the session;
  - timestamp: time of executing shell command/inputting data for shell command;
  - realm: NULL (in the case of shell command execution) or command name (in the case of inputting data for shell command);
  - success: NULL (in the case of inputting data for shell command) or information if command was recognized (1) or not recognized (0) by Kippo (in the case of executing shell command);

  - input: name of command/input data;

- downloads - table containing information about downloaded files; this table has the following columns:
  - id: unique identifier;
  - session: unique identifier of the session;
  - timestamp: time of starting the file download;
  - url: URL of the downloaded file;
  - outfile: name of the file where downloaded file was saved on the host machine;

- ttylog - table containing files with SSH sessions' histories; this table has the following columns:
  - id: unique identifier;
  - session: unique identifier of the session;
  - ttylog: content of the file with SSH session history;

The time registered in database is UTC. Database operations involved with events registered for TCP connections/SSH sessions are discussed below.

Establishment of TCP connection

When there is no information in database about the Kippo instance with which the TCP connection is established, the name of this Kippo instance is inserted into table sensors:

```
insert into sensors (id, ip) values ('3', 'kippo_hp');
```

Information about newly established connection is inserted into table sessions:

```
insert into sessions (id, starttime, endtime, sensor, ip, termsize, client)
values ('e28678b4599511e2bab10800277e980c', '2013-01-08 13:18:26',
    NULL, '3', '192.168.122.1', NULL, NULL);
```

Termination of TCP connection/SSH session

Information about the time of termination of TCP connection/SSH session is added to the session data comprised in appropriate row of table sessions:

```
update sessions set endtime = '2013−01−08 14:03:22' where id = '
    e28678b4599511e2bab10800277e980c';
```

In the case of the termination of SSH session, the content of the file with SSH session history is inserted into table ttylog:

```
insert into ttylog (id, session, ttylog) values ('4', '
    e28678b4599511e2bab10800277e980c', BLOB);
```

Receiving information about SSH client version

When there is no information in database about the received SSH client version it is inserted into table clients:

```
insert into clients (id, version) values ('2', 'SSH−2.0−OpenSSH_5.3');
```

Information about SSH client version is associated with the session described by appropriate row of table sessions:

```
update sessions set client = 3 where id = '
    e28678b4599511e2bab10800277e980c';
```

Client's authentication attempt

Information about client's authentication attempt is inserted into table auth:

```
insert into auth (id, session, success, username, password, timestamp)
values ('12', 'e28678b4599511e2bab10800277e980c', '1', 'root', '123456',
    '2013−01−08 13:33:27');
```

In the case of successful client's authentication attempt information about the size of terminal is added to the session data comprised in appropriate row of table sessions:

```
update sessions set termsize = '129x33' where id = '
    e28678b4599511e2bab10800277e980c';
```

Executing shell command

Information about executing shell command is inserted into table input:

```
insert into input (id, session, timestamp, realm, success, input)
values ('27', 'e28678b4599511e2bab10800277e980c', '2013−01−08 13:42:03',
     NULL, '1', 'pwd')
```

In the case of *wget* command information about downloaded file is inserted into table downloads:

```
insert downloads (id, session, timestamp, url, outfile)
values ('1', 'e28678b4599511e2bab10800277e980c', '2013−01−08 13:42:03', '
    http://cachefly.cachefly.net/100mb.test', '/var/log/kippo/dl
    /20130131032708_http___cachefly_cachefly_net_100mb_test' );
```

Inputting data for shell command

Information about inputting data for shell command is inserted into table input:

```
insert into input (id, session, timestamp, realm, success, input)
values ('27', 'e28678b4599511e2bab10800277e980c', '2013−01−08 13:42:03',
    'passwd', NULL, 'pp');
```

6.3 XMPP server

Kippo enables the sending of part of gathered data to the XMPP server. It uses six kinds of XMPP messages informing about different events:

- establishment of TCP connection;
- termination of TCP connection/SSH session;
- receiving SSH client version;
- failed client's authentication attempt;
- successful client's authentication attempt;
- executing shell command;

With each kind of XMPP message there may be associated separate MUC room, to which these messages will be sent according to XMPP protocol.

Example XMPP messages are included below.

Message informing about the establishment of TCP connection

```
<message from="kippo-events-createsession@conference.localhost/kippo
    -XDJQcVxo" type="groupchat" to="kkoltys@localhost/localhost">
<body>
<kippo xmlns="http://code.google.com/p/kippo/" type="createsession">
<session local_host="127.0.0.1" session="136371692
    cdb4d91b3eec6ff04618232" local_port="2222" remote_port="35533"
    remote_host="192.168.122.1"/>
</kippo>
</body>
</message>
```

Message informing about the termination of TCP connection/SSH session

```
<message from="kippo-events-connectionlost@conference.localhost/
    kippo-XDJQcVxo" type="groupchat" to="kkoltys@localhost/
    localhost">
<body>
<kippo xmlns="http://code.google.com/p/kippo/" type="connectionlost
    ">
<session session="136371692cdb4d91b3eec6ff04618232"/>
</kippo>
</body>
</message>
```

Message informing about receiving SSH client version

```
<message from="kippo-events-clientversion@conference.localhost/kippo
    -XDJQcVxo" type="groupchat" to="kkoltys@localhost/localhost">
<body>
<kippo xmlns="http://code.google.com/p/kippo/" type="clientversion">
<version session="136371692cdb4d91b3eec6ff04618232" version="SSH
    -2.0-OpenSSH_5.3"/>
</kippo>
</body>
</message>
```

Message informing about failed client's authentication attempt

```
<message from="kippo-events-loginfailed@conference.localhost/kippo-
    XDJQcVxo" type="groupchat" to="kkoltys@localhost/localhost">
<body>
<kippo xmlns="http://code.google.com/p/kippo/" type="loginfailed">
<credentials session="136371692cdb4d91b3eec6ff04618232" password
    ="123" username="root"/>
</kippo>
</body>
</message>
```

Message informing about successful client's authentication attempt

```
<message from="kippo-events-loginsucceded@conference.localhost/kippo
    -XDJQcVxo" type="groupchat" to="kkoltys@localhost/localhost">
<body>
<kippo xmlns="http://code.google.com/p/kippo/" type="loginsucceeded
    ">
<credentials session="136371692cdb4d91b3eec6ff04618232" password
    ="123456" username="root"/>
</kippo>
</body>
</message>
```

Message informing about executing shell command

```
<message from="kippo-events-command@conference.localhost/kippo-
    XDJQcVxo" type="groupchat" to="kkoltys@localhost/localhost">
<body>
<kippo xmlns="http://code.google.com/p/kippo/" type="command">
<command session="136371692cdb4d91b3eec6ff04618232" command="
    known">pwd</command>
</kippo>
</body>
</message>
```

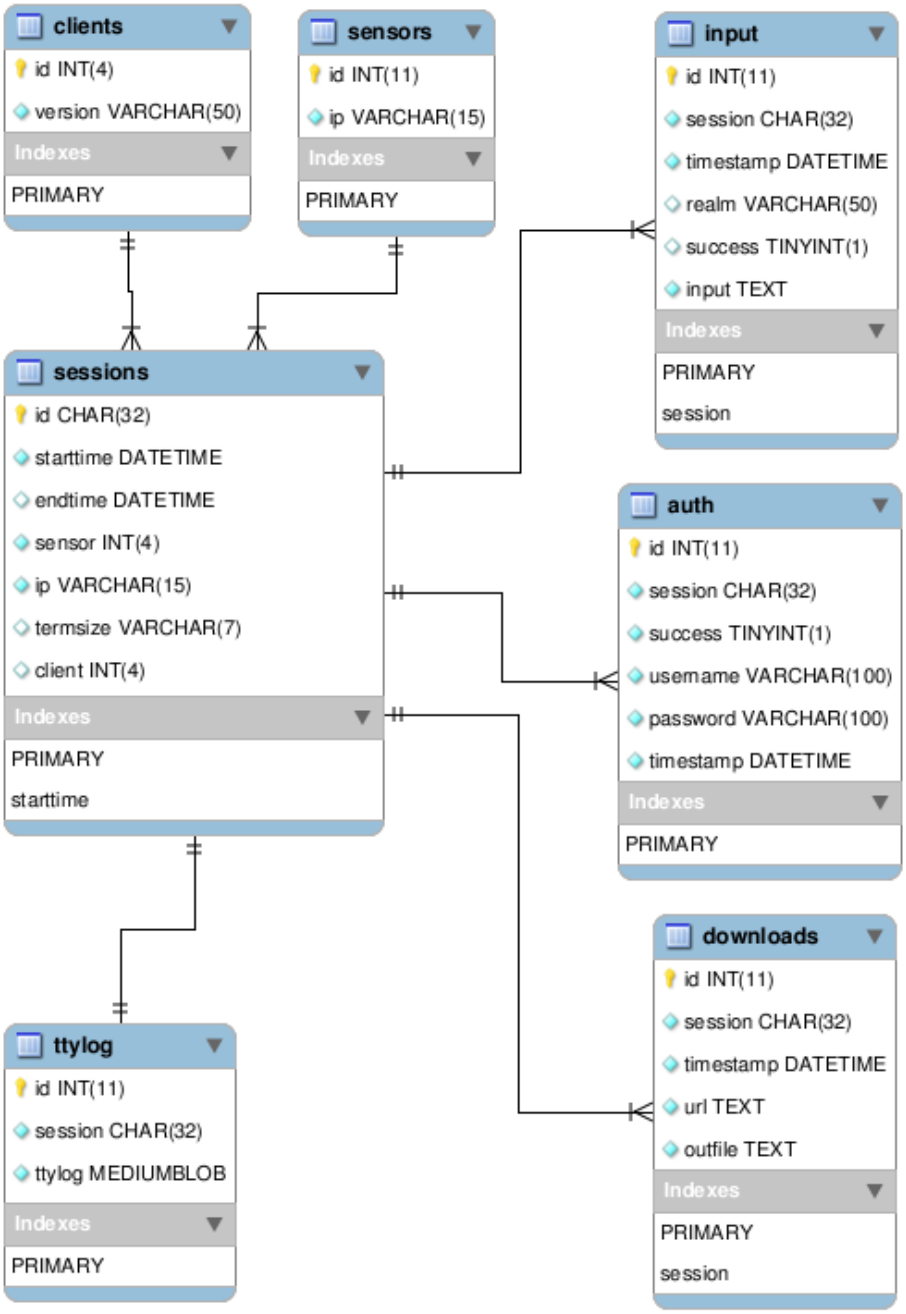

Figure 1: The diagram of Kippo's database