RevCode Client - > Establishing a secure connection to server over SSL (443)

RevCode Android Client will try to establish a connection and retrieve tasks from: <http://androidtest.wm01.to/tasks.php>

Note: Diffie-Hellman handshake authentication is required in order to establish a valid connection. See *Handshake* below.

**Handshake:**

In order to perform the Diffie-Hellman handshake, a reply (echo) from the Server containing a 4-bytes string: “AUTH” has to be made from tasks.php.

Client will then attempt a handshake with: <http://androidtest.wm01.to/auth.php> by posting the following variables:

mode=”keyauth”

data=”{BASE64(PUBLIC\_KEY|UID|dP|dG|dPub)}“

**Input packet structure:**

All received packets needs to be identified by their header followed by necessary parameters. The Client will split parameters split by “|”.

Incoming packet structure (string): “HEADER|TASK\_ID|TASK\_STATUS|Params[]”

**Output packet structure:**

All sent packets by the Android client are sent using HTTP Post.

All packets include the following variables:

* “t\_id” – Task ID
* “key” – Config Key
* “mode” – Task mode
* “uid” – Unique identifiable hardware ID
* “enc” – Encryption; 1=Encryption used, 0=Encryption not used
* “cmp” – Compression; 1=Compression used, 0=Compression not used

1. *If encryption or compression has been used, then Base64 must be used before decrypting or decompressing values.*
2. ***Decoding:*** *If a combination of encryption and compression has been used, then pseudo decoding would be: raw\_value =*

*decompress(decrypt(Base64Decode(encoded\_value)))*

**Important:** Task should be assumed to have failed whenever **“v1”** POST variable equals “0”.

**Performing tasks:**

* **Call logs:** Listing of all call logs
  + **Input:** 
    - Dataheader=**”**CALLS\_LOGS\_GET**”** (string)
    - Data type: POST
    - No parameters needed
  + Output:
    - Data header: mode=”calls\_logs\_get”
    - Data type: POST
    - Variables: **“v1”** – Contains JSon array -> Columns:
      * “contactName” - {Contact name (string)}
      * “phoneNumber” – {Phone number (string)}
      * “callType” – {Call type (string)}
      * “callDate” – {Call date (string)}
      * “callDuration” – {Call duration (string)}
* **SMS logs:** Listing of all SMS
  + **Input:** 
    - Dataheader=**”**SMS\_GET**”** (string)
    - Data type: POST
    - No parameters needed
  + **Output:** 
    - Data header: mode=”sms\_get”
    - Data type: POST
    - Variables: **“v1”** – Contains JSon array -> Columns:
      * “phonenumber” – {Phone number (string)}
      * “mailbox” – {Mailbox (string)}
      * “date” – {Date (string)}
      * “message” – {Message (string)}
* **Location:** Obtain current GPS position
  + **Input:** 
    - Dataheader=**“**LOCATION\_GET**”** (string)
    - Data type: POST
    - No parameters needed
  + **Output:** 
    - Data header: mode=”location\_get”
    - Data type: POST
    - Variables: POST with series of variables
      * “v1” – {Latitude - Get the latitude, in degrees (double)}
      * “v2” – {Longitude - Get the longitude, in degrees (double)}
      * “v3” – {Time - Return the UTC time of this fix, in milliseconds since January 1, 1970 (long)}
      * “v4” – {Accuracy - Get the estimated horizontal accuracy of this location, radial, in meters (float)}
      * “v5” – {Speed - Get the speed if it is available, in meters/second over ground (float)}
      * “v6” – {Bearing - Get the bearing, in degrees (float)}
    - Notes: Client will wait until Android system GPS has been enabled in order to obtain a GPS position.
* **URL opener**: Open a URL in default browser
  + **Input:** 
    - Dataheader=“URL\_OPEN” (string)
    - Data type: POST
    - No parameters needed
  + **Output:** 
    - Data header: mode=”url\_open”
    - Data type: POST
    - Variables: POST with single variable
      * “v1” – {Task status – 1=Success, 0=Failed (long)}
* **Browser bookmarks**: Obtain browser bookmarks
  + **Input:** 
    - Dataheader=“BROWSER\_BOOKMARKS\_GET” (string)
    - Data type: POST
    - No parameters needed
  + **Output:** 
    - Data header: mode=”browser\_bookmarks”
    - Data type: POST
    - Variables: Single variable
      * “v1” – {Row(s) of browser bookmarks data, split by “|” (string)}
* **Device information**: Obtain device information
  + **Input:** 
    - Dataheader=“DEVICE\_INFO” (string)
    - Data type: POST
    - No parameters needed
  + **Output:** 
    - Data header: mode=”device\_info”
    - Data type: POST
    - Variables: **“v1”** - Contains JSon array -> Columns:
      * “imei” – {IMEI number (string)}
      * ”networkOperator” – {Returns the numeric name (MCC+MNC) of current registered operator. (string)}
      * ”networkCountry” – {Returns the ISO country code equivalent of the current registered operator's MCC (Mobile Country Code) (string)}
      * ”networkOperatorName” – { Returns the alphabetic name of current registered operator (string)}
      * ”batteryLevel” – {Battery level including percent-symbol (string)}
      * ”phoneNumber” – { Returns the phone number string for line 1, for example, the MSISDN for a GSM phone (string)}
      * ”radioType” – {Radio type ranging from GPRS to 4G (string)}
      * ”conType” – {Returns a human-readable name describing the type of the network, for example "WIFI" or "MOBILE" (string)}
      * ”deviceName” – {Returns manufacture followed by device model (string)}
      * ”wifiSSID” – {WiFi SSID – Only if connected (string)}
      * ”wifiIP” – {WiFi IP – Only if connected (string)}
      * ”appVersion” – {Application version (string)}
      * ”androidSDKVersion” – {Android SDK Version (string)}
      * ”androidReleaseVersion” – {Android release version (string)}
      * ”deviceModel” – {Device model (string)}
      * ”deviceBrand” – {Device brand (string)}
      * ”deviceProduct” – {Device product (string)}
      * ”deviceBuildID” – {Device Build ID (string)}
      * ”deviceHost” – {Device host (string)}
      * ”deviceSerial” – {Device serial number (string)}
      * ”deviceFingerprint” – {Device fingerprint data – Only if present (string)}
      * ”deviceHardware” – {Device hardware data (string)}
      * ”deviceType” – {Device type (string)}
      * ”deviceUser” – {Device user (string)}
      * ”deviceBootloader” – {Device bootloader data (string)}
      * ”hasRootAccess” – {Returns true/false, depending on whether or not device has Root access (string)}
      * ”hddSizeTotal” – {Total size of internal storage in bytes (double)}
      * ”hddSizeFree” – {Free space of internal storage in bytes (double)}
      * ”SDMounted” – {Returns true/false, depending on whether or not device has a mounted SD card (string)}
      * ”SDSizeFree” – {Free space of SD Card stored in bytes (double)}
      * ”SDSizeTotal” – {Total space of SD card in bytes (double)}
      * ”RAMTotal” – {Total space of RAM in bytes (double)}
      * ”RAMFree” – {Free space of RAM in bytes (double)}
      * ”CPUInfo” – {Information about CPU (string)}
* **Contacts information**: Obtain stored contacts and associated data
  + **Input:** 
    - Data header=“CONTACTS\_GET” (string)
    - Data type: POST
    - No parameters needed
  + **Output:** 
    - Data header: mode=”contacts\_get”
    - Data type: POST
    - Variables: **“v1”** - Contains JSon array -> Columns:
      * “name” – {Name (string)}
      * “phonenumber[1..n]” – {Phone number(s) (string)}
      * “email[1..n]” – {Email(s) (string)}
* **Sound recording**: Records from microphone for n given seconds and sends back binary of recording data
  + **Input:** 
    - Data header=“SOUND\_RECORD” (string)
    - Data type: POST
    - Parameters:
      * DURATION\_SECONDS (long)
  + **Output:** 
    - Data header: mode=”sound\_record”
    - Data type: POST
    - Variables: Single variable containing binary of a 3GPP media file
      * “data” – {Base64 encoded binary data. Note: See ***Output packet structure*, *section 2***, if decoding is needed - (string)}
* **Camera snapshot**: Captures a snapshot from front or back camera
  + **Input:** 
    - Data header=“CAM\_SHOT” (string)
    - Data type: POST
    - Parameters:
      * CAMERA\_INDEX – {Index starting from the back-facing camera on the device from 0 to N of camera devices - (long)}
  + **Output:** 
    - Data header: mode=”cam\_shot”
    - Data type: POST
    - Variables: Single variable containing binary of a JPG image.
      * “data” – {Base64 encoded binary data. Note: See ***Output packet structure*, *section 2***, if decoding is needed - (string)}
* **Gallery**: Obtains thumbnails of all stored images in Gallery
  + **Input:** 
    - Data header=“GALLERY\_GET” (string)
    - Data type: POST
    - Parameters:
      * ARR\_MD5 [OPTIONAL] – {Array of MD5 checksums, split by vertical tab, “\t”. This parameter is an optional caching mechanism. Images having a matched checksum will be excluded – (string[])}
  + **Output:** 
    - Data header: mode=”gallery\_get”
    - Data type: POST
    - Variables: **“data”** - Containing JSon array -> Columns:
      * “name” – {File name - (string)}
      * “f\_size” – {Full image size - (long)}
      * “t\_size” – {Thumbnail size - (long)}
      * “exif\_model” – {Exif model - (string)}
      * “exif\_make” – {Exit make - (string)}
      * “date” – {Date of creation - (string)}
      * “md5” – {MD5 checksum, used - (string)}
      * “data” – {Base64 encoded thumbnail binary data - (string)}
* **Installed applications**: Obtain information about installed apps
  + **Input:** 
    - Data header=“INSTALLED\_APPS\_GET” (string)
    - Data type: POST
    - No parameters needed
  + **Output:** 
    - Data header: mode=”installed\_apps\_get”
    - Data type: POST
    - Variables: **“v1”** - Contains JSon array -> Columns:
      * “appName” – {Name (string)}
      * “appPackageName” – {Package name (string)}
      * “appPermissions” – {Permissions (string)}
      * “appProcName” – {Process name (string)}
      * “appVersion” – {Version (string)}
      * “appInfo” – {Additional information (string)}
      * “appIconData” – {Base64 encoded icon binary data - (string)}
* **WiFi networks**: Listing of all available WiFi networks
  + **Input:** 
    - Data header=“ WIFI\_GET” (string)
    - Data type: POST
    - No parameters needed
  + **Output:** 
    - Data header: mode=”wifi\_get”
    - Data type: POST
    - Variables: **“v1”** - Contains JSon array -> Columns:
      * “ssid” – {SSID (string)}
      * “bssid” – {MAC address of the wireless access point (string)}
      * “frequency” – {Frequency in GHZ (string)}
      * “rssi” – {Received signal strength indicator in dBm (string)}
      * “capabilities” – {WiFi capabilities (string)}
* **Clipboard**: Obtain clipboard data
  + **Input:** 
    - Data header=“CLIPBOARD\_GET” (string)
    - Data type: POST
    - No parameters needed
  + **Output:** 
    - Data header: mode=”clipboard\_get”
    - Data type: POST
    - Variables: **“v1”** - Contains JSon array -> Columns:
      * “html” – {HTML formatted data (string)}
      * “text” – {Raw data (string)}