# Client Side

# Installer

## 1.1.1 Verify Installation procedure

Check if the machine already has a running process of the miners.

***Read a bit from a unique path (IsInstalledPath) in registry – if does not exist INSATLL***

If true (There is already a miner process installed & running):

* Don’t start installer

Else

* Start installer

## Show terms and conditions

## On Finish send install Ping Pixel & write bit to predefined path (***IsInstalledPath)*** in registry

## Extract all packages on the machine & Run MinerManager

## Add the minerManager to the boot sequence

# MinerManager

## Check if Configuration file exists and its timestamp

1. If there is no Configuration File (FIRST\_RUN\_STATE)

Register PC -> Parse Response

If response not Success loop every 5 min till getting success response

If response success

Save response configuration file to the local disk

1. On Start && Every 10 MIns:

For each miner in response check if dead:

{GetMiner() – parse the response

If response not Success loop every 5 min till getting success response

If response success:

Save MinerConfigurationFile

StartMiner()

}

# Miner (GPU / CPU)

1. Check if should run:

while (TMP of cpu/gpu > Max TMP)

Sleep (30 MINS)

While (RunOnly InActive && Active in last 10 Mins)

Sleep (10 MINS)

Connect to Pool using Pool info (HostName, Port, UserName, Password, PoolType, Algo)

While (Connect Not Success){

Sleep (10 Mins)

ConnectionTrialCounter ++;

If (ConnectionTrialCounter ==5 )

Exist;}

If (Connection to pool succesfull)

start mining();

count shares + time worked

Write to persistent storage and internal memory

Every updateTimeInmints{

send to server(time worked, shares created, log)

// Log If SEND\_LOG is true;

Parse response -> If response is success

Clear stats (time worked, shares created, log)

If (ASK\_FOR\_NEW\_MINER is true)

Exit;

}

Check each 60 seconds

check tmp if > threshold

Exit;

# Server Side

API:

1. register\_pc

Request:

json {

"pc": {

"mac": "",

"os\_ver": "",

"computer\_name": "",

"system\_type": "",

"ram\_size": "",

"ip\_address": "",

"is\_laptop": ""

};

"gpu": [

{

"number": "",

"params": {

"name": "",

"core\_amount": "",

}

},

{

"number": "",

"params": {

"name": "",

"core\_amount": "",

}

}

],

"cpu": {

"name": "",

"core\_amount": ""

}

*"campaign": {*

*"subID0": "",*

*"subID1": "",*

*"subID2": "",*

*"subID3": "",*

*"subID4": "",*

}

}

Response:

json {

"pc\_id": "",

"gpu": [

{

"number": "",

"pc\_gpu\_type\_id": ""

},

{

"number": "",

"pc\_gpu\_type\_id": ""

},

],

"cpu":

{

"pc\_cpu\_type\_id": ""

}

}

1. get\_miner

Request:

json {

"pc\_id": "",

"pc\_cpu\_type\_id": "", // is null if pc\_gpu\_type\_id is defined

"pc\_gpu\_type\_id": "" // is null if pc\_cpu\_type\_id is defined

}

Response:

json {

"max\_temp": "",

"max\_usage": "",

"run\_only\_inactive": "",

"pool\_info": {

"hostname": "",

"port": "",

"username": "",

"password": "",

"pool\_type": {

"name": "",

"algo": ""

}

}

"update\_interval\_mins": "",

"send\_log": ""

}

1. update\_miner\_stat

Request:

json {

"pc\_id": "",

"pc\_cpu\_type\_id": "", // is null if pc\_gpu\_type\_id is defined

"pc\_gpu\_type\_id": "", // is null if pc\_cpu\_type\_id is defined

"params": {

"shares": "",

"time": ""

},

*"campaign": {*

*"subID0": "",*

*"subID1": "",*

*"subID2": "",*

*"subID3": "",*

*"subID4": "",*

"log": ""

}

Response:

json {

"status": "", // 0 fail, 1 success

"ask\_for\_new\_miner": "" // 0 no, 1 yes

}

# DB SCHEMA

CREATE TABLE pcs (

    id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

    mac VARCHAR(20) NOT NULL,

    os\_ver VARCHAR(255) NOT NULL,

    computer\_name VARCHAR(255) NOT NULL,

    system\_type VARCHAR(255) NOT NULL,

    ram\_size INT NOT NULL,

    ip\_address VARCHAR(18) NOT NULL,

    is\_laptop BOOLEAN,

    UNIQUE(mac)

) ENGINE=InnoDB CHARSET=utf8;

CREATE TABLE gpu\_confs (

    id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

    name VARCHAR(255) NOT NULL,

    max\_temp INT NOT NULL,

    max\_cpu INT NOT NULL,

    run\_only\_inactive BOOLEAN NOT NULL

) ENGINE=InnoDB CHARSET=utf8;

CREATE TABLE cpu\_confs (

    id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

    name VARCHAR(255) NOT NULL,

    max\_temp INT NOT NULL,

    max\_gpu INT NOT NULL,

    run\_only\_inactive BOOLEAN NOT NULL

) ENGINE=InnoDB CHARSET=utf8;

CREATE TABLE cpu\_types (

    id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

    processor VARCHAR(255) NOT NULL,

    core\_amount INT NOT NULL,

    cpu\_conf\_id INT NOT NULL,

    FOREIGN KEY (cpu\_conf\_id) REFERENCES cpu\_confs(id),

    UNIQUE(processor)

) ENGINE=InnoDB CHARSET=utf8;

CREATE TABLE gpu\_types (

    id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

    name VARCHAR(255) NOT NULL,

    core\_amount INT NOT NULL,

    gpu\_conf\_id INT NOT NULL,

    FOREIGN KEY (gpu\_conf\_id) REFERENCES gpu\_confs(id),

    UNIQUE(name)

) ENGINE=InnoDB CHARSET=utf8;

CREATE TABLE pc\_cpu\_types (

    id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

    pc\_id INT NOT NULL,

    cpu\_type\_id INT NOT NULL,

    FOREIGN KEY (pc\_id) REFERENCES pcs(id),

    FOREIGN KEY (cpu\_type\_id) REFERENCES cpu\_typees(id),

    UNIQUE(pc\_id, cpu\_type\_id)

) ENGINE=InnoDB CHARSET=utf8;

CREATE TABLE pc\_gpu\_types (

    id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

    pc\_id INT NOT NULL,

    gpu\_type\_id INT NOT NULL,

    FOREIGN KEY (pc\_id) REFERENCES pcs(id),

    FOREIGN KEY (gpu\_type\_id) REFERENCES gpu\_typees(id),

    UNIQUE(pc\_id, gpu\_type\_id)

) ENGINE=InnoDB CHARSET=utf8;

CREATE TABLE pools (

    id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

    hostname VARCHAR(255) NOT NULL,

    port INT NOT NULL,

    username VARCHAR(255) NOT NULL,

    password VARCHAR(255) NOT NULL,

    pool\_type\_id INT NOT NULL,

    FOREIGN KEY (pool\_types) REFERENCES pool\_types(id)

) ENGINE=InnoDB CHARSET=utf8;

CREATE TABLE pool\_types (

    id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

    name VARCHAR(255) NOT NULL,

    algo VARCHAR(255) NOT NULL,

    UNIQUE(name)

) ENGINE=InnoDB CHARSET=utf8;

CREATE TABLE cpu\_miners (

    id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

    pc\_cpu\_type\_id INT NOT NULL,

    pool\_id INT NOT NULL,

    start\_datetime DATETIME NOT NULL, -- Timestamp of install

    end\_datetime DATETIME NOT NULL, -- Last ping

    total\_time INT NOT NULL, -- Total time worked

    shares INT NOT NULL, -- Total Shares gathered

    FOREIGN KEY (pc\_cpu\_type\_id) REFERENCES pc\_cpu\_types(id),

    FOREIGN KEY (pool\_id) REFERENCES pools(id),

    UNIQUE(pc\_cpu\_type\_id, pool\_id)

) ENGINE=InnoDB CHARSET=utf8;

CREATE TABLE gpu\_miners (

    id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

    pc\_gpu\_type\_id INT NOT NULL,

    pool\_id INT NOT NULL,

    start\_datetime DATETIME NOT NULL, -- Timestamp of install

    end\_datetime DATETIME NOT NULL, -- Last ping

    total\_time INT NOT NULL, -- Total time worked

    shares INT NOT NULL, -- Total Shares gathered

    FOREIGN KEY (pc\_gpu\_type\_id) REFERENCES pc\_gpu\_types(id),

    FOREIGN KEY (pool\_id) REFERENCES pools(id),

    UNIQUE(pc\_gpu\_type\_id, pool\_id)

) ENGINE=InnoDB CHARSET=utf8;

TO DO

1. Installer GUI
2. QA – For different CPU, GPU
3. Anti Virus Checks