WIFI入侵检测防御应用系统

目录

[WIFI入侵检测防御应用系统 1](#_Toc475115239)

[1 系统简介 3](#_Toc475115240)

[1.1 术语定义 3](#_Toc475115241)

[2 数据库设计 4](#_Toc475115242)

[2.1 检测前端信息 4](#_Toc475115243)

[2.2 周边AP和终端信息 4](#_Toc475115244)

[2.3 录入AP和终端信息 6](#_Toc475115245)

[2.3.1 信任AP 6](#_Toc475115246)

[2.3.2 邻居AP 6](#_Toc475115247)

[2.3.3 访客AP 7](#_Toc475115248)

[2.3.4 自组网AP 7](#_Toc475115249)

[2.3.5 忽略MAC 8](#_Toc475115250)

[2.3.6 保护终端 8](#_Toc475115251)

[2.3.7 邻居终端 9](#_Toc475115252)

[2.3.8 访客终端 9](#_Toc475115253)

[2.3.9 自组网终端 9](#_Toc475115254)

[2.4 威胁事件信息 10](#_Toc475115255)

[2.4.1 威胁事件定义 10](#_Toc475115256)

[2.4.2 告警事件记录 12](#_Toc475115257)

[2.5 用户管理 13](#_Toc475115258)

[2.6 用户审计日志 14](#_Toc475115259)

[3 其它接口 15](#_Toc475115260)

[3.1 网络通知接口 15](#_Toc475115261)

[3.1.1 连接请求 16](#_Toc475115262)

[3.1.2 心跳请求 16](#_Toc475115263)

[3.1.3 心跳应答 16](#_Toc475115264)

[3.1.4 业务数据 17](#_Toc475115265)

[3.1.5 数据示例 20](#_Toc475115266)

[4 用户界面 22](#_Toc475115267)

[4.1 登陆 22](#_Toc475115268)

[4.2 检测前端设备管理 22](#_Toc475115269)

[4.3 预览周边WIFI环境 22](#_Toc475115270)

[4.3.1 预览AP信息 22](#_Toc475115271)

[4.3.2 展开AP预览终端 23](#_Toc475115272)

[4.3.3 将AP或终端加入名单 23](#_Toc475115273)

[4.3.4 周边AP安全报告 24](#_Toc475115274)

[4.3.5 下载生成的安全报告 25](#_Toc475115275)

[4.4 管理MAC名单 25](#_Toc475115276)

[4.5 事件告警 25](#_Toc475115277)

[4.5.1 查看系统支持检测的威胁事件 25](#_Toc475115278)

[4.5.2 显示实时告警信息 26](#_Toc475115279)

[4.5.3 历史告警事件查询 26](#_Toc475115280)

[4.6 用户管理 26](#_Toc475115281)

[4.7 审计日志查看 27](#_Toc475115282)

[4.8 系统升级 27](#_Toc475115283)

# 系统简介

应用系统是WIFI入侵检测防御系统的一部分。系统由下面三部分组成：

**应用系统** 1

**用户UI** 0…N

**控制中心** 1…N

**检测前端** 1…N

检测前端负责实现系统数据采集和功能实施。控制中心负责管理检测前端，并中转应用系统发送的指令，将一些需要及时交给用户查看和处理的信息转交给应用系统。本文接下来的需求和设计主要是对应用系统进行约束的。

## 术语定义

本文档将使用下面一些术语。

|  |  |
| --- | --- |
| 名称 | 定义 |
| AP | Access Point，WIFI接入点，可以理解为WIFI路由器 |
| SSID | AP的名称，可以理解为WIFI的名称，如i-hangzhou |
| 信任AP | 系统保护的AP，可以理解为自家的WIFI |
| 邻居AP | 在系统保护AP周边的AP，可以理解为邻居家WIFI |
| 访客AP | 开通给访客访问的AP，可以理解为专门开放给客人的WIFI |
| 自组网AP | 检测前端通过WIFI上网时使用的AP |
| 忽略MAC | 不进行检测的MAC地址 |
| 保护终端 | 可以连接信任AP的终端，如自家的手机，PAD，笔记本电脑 |

# 数据库设计

数据库中bigint(20)的时间值都是EPOCH时间。

## 检测前端信息

使用数据库表为device\_monitor，建表语句如下：

CREATE TABLE `device\_monitor` (

`device\_serial` varchar(16) NOT NULL COMMENT '设备序列号',

`install\_time` bigint(20) not null comment '安装时间',

`version` int(11) NOT NULL COMMENT '前端版本',

`name` varchar(64) NOT NULL COMMENT '名称',

`function` int(11) NOT NULL COMMENT '当前功能',

`online` int(11) NOT NULL COMMENT '在线标志，1：在线 0：下线',

`last\_time` bigint(20) NOT NULL COMMENT '最后活动时间',

`remarks` varchar(256) NOT NULL COMMENT '备注',

`latitude` varchar(16) NOT NULL DEFAULT '' COMMENT '纬度',

`longitude` varchar(16) NOT NULL DEFAULT '' COMMENT '经度',

`reserved1` int(11) NOT NULL DEFAULT '0' COMMENT '预留',

`reserved2` int(11) NOT NULL DEFAULT '0' COMMENT '预留',

PRIMARY KEY (`device\_serial`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

## 周边AP和终端信息

使用两张临时表来记录当前周边在线的AP和终端信息。其中表名为scan\_ap和scan\_station。AP或终端MAC地址在对应的检测前端出现的信息在临时表scan\_mac\_relation中。临时表的意思是每次控制中心如果关闭后重启，则数据库表会被清空。

AP信息表scan\_ap建表语句如下：

CREATE TABLE `scan\_ap` (

`mac` varchar(24) NOT NULL COMMENT 'MAC地址',

`identity` int(11) NOT NULL COMMENT '1:ap,3:bridge,5:bridge下的AP,6:bridge下未知,7:未知',

`channel` int(11) NOT NULL COMMENT '信道',

`signal` int(11) NOT NULL COMMENT '信号,单位dBm',

`privacy` int(11) NOT NULL COMMENT '保护方式 0x00:不加密, 0x01:WEP, 0x02:WPA, 0x03:WEP|WPA, 0x04:WPA2, 0x05:WEP|WPA2, 0x06:WPA|WPA2, 0x07:WEP|WPA|WPA2',

`encrypt\_way` int(11) not null comment '加密算法 0:不加密, 1:WEP40, 2:WEP104, 3:WEP40|WEP104, 4:TKIP, 8:CCMP, 12:TKIP|CCMP',

`auth\_way` int(11) not null comment '认证方式 0:不认证, 1:PSK, 2:RADIUS',

`wps` int(11) not null comment 'WPS状态，0:没有开启WPS,1:启用WPS',

`ssid` varchar(34) NOT NULL COMMENT 'SSID',

`broadcast\_ssid` int(11) not null comment '是否开启SSID广播，0:不开启,1:已开启',

`maker` varchar(256) NOT NULL COMMENT '硬件制造商',

PRIMARY KEY (`mac`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

终端信息表scan\_station建表语句如下：

CREATE TABLE `scan\_station` (

`mac` varchar(24) NOT NULL COMMENT 'MAC地址',

`identity` int(11) NOT NULL COMMENT '2:station,4:bridge下的station,6:bridge下未知,7:未知',

`channel` int(11) NOT NULL COMMENT '信道',

`signal` int(11) NOT NULL COMMENT '信号,单位dBm',

`maker` varchar(256) NOT NULL COMMENT '硬件制造商',

`mac\_ap` varchar(24) NOT NULL COMMENT '所属AP的MAC地址',

PRIMARY KEY (`mac`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

AP和终端在检测前端出现的信息表scan\_mac\_relation建表语句如下：

CREATE TABLE `scan\_mac\_relation` (

`mac` varchar(24) NOT NULL COMMENT 'MAC地址',

`device\_serial` varchar(16) NOT NULL COMMENT '出现前端的序列号',

PRIMARY KEY (`mac`,`device\_serial`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

## 录入AP和终端信息

### 信任AP

CREATE TABLE `ap\_trust` (

`mac` varchar(24) NOT NULL COMMENT 'mac地址',

`ssid` varchar(34) NOT NULL COMMENT '路由名',

`pwd` varchar(64) NOT NULL COMMENT '密码',

`maker` varchar(256) NOT NULL COMMENT '硬件制造商',

`remarks` varchar(128) NOT NULL COMMENT '备注',

`privacy` int(11) NOT NULL COMMENT '加密方式 0:不加密, 1:WEP, 2:WPA, 3:WEP|WPA, 4:WPA2, 5:WEP|WPA2, 6:WPA|WPA2, 7:WEP|WPA|WPA2',

`reserved1` int(11) NOT NULL DEFAULT '0' COMMENT '预留',

`reserved2` int(11) NOT NULL DEFAULT '0' COMMENT '预留',

`beacon\_size` int(11) NOT NULL COMMENT 'beacon数据长度',

`beacon` varchar(2312) DEFAULT NULL COMMENT 'beacon数据',

`probe\_response\_size` int(11) NOT NULL COMMENT 'probe\_response数据长度',

`probe\_response` varchar(2312) DEFAULT NULL COMMENT 'probe\_response数据',

PRIMARY KEY (`mac`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

### 邻居AP

CREATE TABLE `ap\_neighbor` (

`mac` varchar(24) NOT NULL COMMENT 'mac地址',

`ssid` varchar(34) NOT NULL COMMENT '路由名',

`pwd` varchar(64) NOT NULL COMMENT '密码',

`maker` varchar(256) NOT NULL COMMENT '硬件制造商',

`remarks` varchar(128) NOT NULL COMMENT '备注',

`privacy` int(11) NOT NULL COMMENT '加密方式 0:不加密, 1:WEP, 2:WPA, 3:WEP|WPA, 4:WPA2, 5:WEP|WPA2, 6:WPA|WPA2, 7:WEP|WPA|WPA2',

`reserved1` int(11) NOT NULL DEFAULT '0' COMMENT '预留',

`reserved2` int(11) NOT NULL DEFAULT '0' COMMENT '预留',

`beacon\_size` int(11) NOT NULL COMMENT 'beacon数据长度',

`beacon` varchar(2312) DEFAULT NULL COMMENT 'beacon数据',

`probe\_response\_size` int(11) NOT NULL COMMENT 'probe\_response数据长度',

`probe\_response` varchar(2312) DEFAULT NULL COMMENT 'probe\_response数据',

PRIMARY KEY (`mac`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

### 访客AP

CREATE TABLE `ap\_visitor` (

`mac` varchar(24) NOT NULL COMMENT 'mac地址',

`ssid` varchar(34) NOT NULL COMMENT '路由名',

`pwd` varchar(64) NOT NULL COMMENT '密码',

`maker` varchar(256) NOT NULL COMMENT '硬件制造商',

`remarks` varchar(128) NOT NULL COMMENT '备注',

`privacy` int(11) NOT NULL COMMENT '加密方式 0:不加密, 1:WEP, 2:WPA, 3:WEP|WPA, 4:WPA2, 5:WEP|WPA2, 6:WPA|WPA2, 7:WEP|WPA|WPA2',

`reserved1` int(11) NOT NULL DEFAULT '0' COMMENT '预留',

`reserved2` int(11) NOT NULL DEFAULT '0' COMMENT '预留',

`beacon\_size` int(11) NOT NULL COMMENT 'beacon数据长度',

`beacon` varchar(2312) DEFAULT NULL COMMENT 'beacon数据',

`probe\_response\_size` int(11) NOT NULL COMMENT 'probe\_response数据长度',

`probe\_response` varchar(2312) DEFAULT NULL COMMENT 'probe\_response数据',

PRIMARY KEY (`mac`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

### 自组网AP

CREATE TABLE `ap\_ad\_hoc` (

`mac` varchar(24) NOT NULL COMMENT 'mac地址',

`ssid` varchar(34) NOT NULL COMMENT '路由名',

`pwd` varchar(64) NOT NULL COMMENT '密码',

`maker` varchar(256) NOT NULL COMMENT '硬件制造商',

`remarks` varchar(128) NOT NULL COMMENT '备注',

`privacy` int(11) NOT NULL COMMENT '加密方式 0x00:不加密, 0x01:WEP, 0x02:WPA, 0x03:WEP|WPA, 0x04:WPA2, 0x05:WEP|WPA2, 0x06:WPA|WPA2, 0x07:WEP|WPA|WPA2',

`reserved1` int(11) NOT NULL DEFAULT '0' COMMENT '预留',

`reserved2` int(11) NOT NULL DEFAULT '0' COMMENT '预留',

`beacon\_size` int(11) NOT NULL COMMENT 'beacon数据长度',

`beacon` varchar(2312) DEFAULT NULL COMMENT 'beacon数据',

`probe\_response\_size` int(11) NOT NULL COMMENT 'probe\_response数据长度',

`probe\_response` varchar(2312) DEFAULT NULL COMMENT 'probe\_response数据',

PRIMARY KEY (`mac`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

### 忽略MAC

CREATE TABLE `exception\_mac` (

`mac` varchar(24) NOT NULL COMMENT '例外MAC,将不对该MAC进行告警',

`name` varchar(64) NOT NULL COMMENT '名称',

`remarks` varchar(128) NOT NULL COMMENT '备注',

`create\_time` bigint(20) NOT NULL COMMENT '记录创建时间',

`cnvalid\_time` bigint(20) NOT NULL COMMENT '记录失效时间',

`excp\_state` int(11) NOT NULL COMMENT '例外状态,0:失效,1:有效',

PRIMARY KEY (`mac`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

### 保护终端

CREATE TABLE `station\_trust` (

`mac` varchar(24) NOT NULL COMMENT 'mac地址',

`maker` varchar(256) NOT NULL COMMENT '硬件制造商',

`name` varchar(64) NOT NULL COMMENT '姓名',

`phone\_number` varchar(32) NOT NULL COMMENT '电话号码',

`id\_number` varchar(64) NOT NULL COMMENT '证件号',

`department` varchar(64) NOT NULL COMMENT '部门',

`remarks` varchar(128) NOT NULL COMMENT '备注',

`reserved1` int(11) NOT NULL DEFAULT '0' COMMENT '预留',

`reserved2` int(11) NOT NULL DEFAULT '0' COMMENT '预留',

PRIMARY KEY (`mac`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

### 邻居终端

CREATE TABLE `station\_neighbor` (

`mac` varchar(24) NOT NULL COMMENT 'mac地址',

`maker` varchar(256) NOT NULL COMMENT '硬件制造商',

`name` varchar(64) NOT NULL COMMENT '姓名',

`phone\_number` varchar(32) NOT NULL COMMENT '电话号码',

`id\_number` varchar(64) NOT NULL COMMENT '证件号',

`remarks` varchar(128) NOT NULL COMMENT '备注',

`reserved1` int(11) NOT NULL DEFAULT '0' COMMENT '预留',

`reserved2` int(11) NOT NULL DEFAULT '0' COMMENT '预留',

PRIMARY KEY (`mac`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

### 访客终端

CREATE TABLE `station\_visitor` (

`mac` varchar(24) NOT NULL COMMENT 'mac地址',

`maker` varchar(256) NOT NULL COMMENT '硬件制造商',

`name` varchar(64) NOT NULL COMMENT '姓名',

`phone\_number` varchar(32) NOT NULL COMMENT '电话号码',

`id\_number` varchar(64) NOT NULL COMMENT '证件号',

`remarks` varchar(128) NOT NULL COMMENT '备注',

`reserved1` int(11) NOT NULL DEFAULT '0' COMMENT '预留',

`reserved2` int(11) NOT NULL DEFAULT '0' COMMENT '预留',

PRIMARY KEY (`mac`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

### 自组网终端

CREATE TABLE `station\_ad\_hoc` (

`mac` varchar(24) NOT NULL COMMENT 'mac地址',

`maker` varchar(256) NOT NULL COMMENT '硬件制造商',

`name` varchar(64) NOT NULL COMMENT '姓名',

`remarks` varchar(128) NOT NULL COMMENT '备注',

`reserved1` int(11) NOT NULL DEFAULT '0' COMMENT '预留',

`reserved2` int(11) NOT NULL DEFAULT '0' COMMENT '预留',

PRIMARY KEY (`mac`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

## 威胁事件信息

### 威胁事件定义

分为六张表，表invade\_type定义攻击事件类型：

CREATE TABLE `invade\_type` (

`invade\_id` int(11) NOT NULL COMMENT '类型ID',

`name` varchar(64) NOT NULL COMMENT '类型名',

`flag` int(11) NOT NULL COMMENT '入侵类型标志,0:未激活,1:激活',

`des` varchar(256) NOT NULL COMMENT '类型描述',

`reserved` int(11) NOT NULL DEFAULT '0' COMMENT '预留',

PRIMARY KEY (`invade\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

表defense\_type定义建议的解决方式：

CREATE TABLE `defense\_type` (

`defense\_id` int(11) NOT NULL COMMENT '策略ID',

`name` varchar(64) NOT NULL COMMENT '策略名',

`when` int(11) NOT NULL COMMENT '策略有效时长',

`reserved` int(11) NOT NULL COMMENT '预留',

`des` varchar(256) NOT NULL COMMENT '策略描述',

PRIMARY KEY (`defense\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

表strategy定义入侵事件类型和解决方式之间的关联关系，注意，其关联关系为invade\_type与defense\_type一对多：

CREATE TABLE `strategy` (

`invade\_id` int(11) NOT NULL COMMENT '入侵类型ID',

`defense\_id` int(11) NOT NULL COMMENT '策略ID',

PRIMARY KEY (`invade\_id`,`defense\_id`),

KEY `invade\_id` (`invade\_id`),

CONSTRAINT `strategy\_ibfk\_1` FOREIGN KEY (`invade\_id`) REFERENCES `invade\_type` (`invade\_id`) ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT `strategy\_ibfk\_2` FOREIGN KEY (`defense\_id`) REFERENCES `defense\_type` (`defense\_id`) ON DELETE CASCADE ON UPDATE CASCADE

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

表alarm\_level定义了危害等级，建表语句如下：

CREATE TABLE `alarm\_level` (

`level\_id` int(11) NOT NULL COMMENT '级别ID',

`name` varchar(64) NOT NULL COMMENT '级别名',

`des` varchar(256) NOT NULL COMMENT '级别描述',

`flag` int(11) NOT NULL COMMENT '告警标志,0:不需要告警,1:需要告警',

`reserved` int(11) NOT NULL DEFAULT '0' COMMENT '预留',

PRIMARY KEY (`level\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

表relation\_alarm\_invade关联了攻击事件与告警级别的关系，其关系为alarm\_level对invade\_type一对多，relation\_alarm\_invade的建表语句如下：

CREATE TABLE `relation\_alarm\_invade` (

`level\_id` int(11) NOT NULL COMMENT '告警级别ID',

`invade\_id` int(11) NOT NULL COMMENT '入侵类型ID',

PRIMARY KEY (`level\_id`,`invade\_id`),

KEY `invade\_id` (`invade\_id`),

CONSTRAINT `relation\_alarm\_invade\_ibfk\_1` FOREIGN KEY (`level\_id`) REFERENCES `alarm\_level` (`level\_id`) ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT `relation\_alarm\_invade\_ibfk\_2` FOREIGN KEY (`invade\_id`) REFERENCES `invade\_type` (`invade\_id`) ON DELETE CASCADE ON UPDATE CASCADE

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

表strategy\_default定义入侵事件类型和解决方式之间的默认关联关系，在初始化时，它和strategy一样，但是这张表不能修改，用于重置入侵类型响应方式：

CREATE TABLE `strategy\_default` (

`invade\_id` int(11) NOT NULL COMMENT '入侵类型ID',

`defense\_id` int(11) NOT NULL COMMENT '策略ID',

PRIMARY KEY (`invade\_id`,`defense\_id`),

KEY `invade\_id` (`invade\_id`),

CONSTRAINT `strategy\_ibfk\_1` FOREIGN KEY (`invade\_id`) REFERENCES `invade\_type` (`invade\_id`) ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT `strategy\_ibfk\_2` FOREIGN KEY (`defense\_id`) REFERENCES `defense\_type` (`defense\_id`) ON DELETE CASCADE ON UPDATE CASCADE

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

### 告警事件记录

告警记录alarm\_record建表语句如下：

CREATE TABLE `alarm\_record` (

`id` int(11) NOT NULL AUTO\_INCREMENT COMMENT '记录ID',

`account\_id` int(11) NOT NULL COMMENT '告警账号ID',

`mac` varchar(24) NOT NULL COMMENT '告警MAC',

`mac\_invaded` varchar(24) NOT NULL COMMENT '被入侵设备的MAC',

`invade\_id` int NOT NULL COMMENT '对应的入侵类型',

`device\_serial` varchar(16) NOT NULL COMMENT '告警前端设备ID',

`create\_time` bigint(20) NOT NULL,

`alarm\_state` int(11) NOT NULL COMMENT '告警状态,0:未推送,1:已推送,2:推送失败',

`read\_state` int(11) NOT NULL COMMENT 'Web读状态,0:未读,1:已读',

`alarm\_msg` varchar(256) NOT NULL COMMENT '告警信息',

PRIMARY KEY (`id`),

KEY `account\_id` (`account\_id`),

CONSTRAINT `alarm\_record\_ibfk\_1` FOREIGN KEY (`account\_id`) REFERENCES `alarm\_account` (`account\_id`) ON DELETE CASCADE ON UPDATE CASCADE

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

对于已经阅读的告警信息，将其移入历史告警信息表history\_alarm\_record，其建表语句与普通告警事件记录一致，如下：

CREATE TABLE `history\_alarm\_record` (

`id` int(11) NOT NULL AUTO\_INCREMENT COMMENT '记录ID',

`account\_id` int(11) NOT NULL COMMENT '告警账号ID',

`mac` varchar(24) NOT NULL COMMENT '告警MAC',

`mac\_invaded` varchar(24) NOT NULL COMMENT '被入侵设备的MAC',

`invade\_id` int NOT NULL COMMENT '对应的入侵类型',

`device\_serial` varchar(16) NOT NULL COMMENT '告警前端设备ID',

`create\_time` bigint(20) NOT NULL,

`alarm\_state` int(11) NOT NULL COMMENT '告警状态,0:未推送,1:已推送,2:推送失败',

`read\_state` int(11) NOT NULL COMMENT 'Web读状态,0:未读,1:已读',

`alarm\_msg` varchar(256) NOT NULL COMMENT '告警信息',

PRIMARY KEY (`id`),

KEY `account\_id` (`account\_id`),

CONSTRAINT `history\_alarm\_record\_ibfk\_1` FOREIGN KEY (`account\_id`) REFERENCES `alarm\_account` (`account\_id`) ON DELETE CASCADE ON UPDATE CASCADE

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

告警消息需要与告警账号关联，关联表alarm\_account建表语句如下：

CREATE TABLE `alarm\_account` (

`account\_id` int(11) NOT NULL COMMENT '账号ID',

`user\_id` int(11) NOT NULL COMMENT '用户ID',

`account` varchar(64) NOT NULL COMMENT '账号',

`account\_type` int(11) NOT NULL COMMENT '账号类型,0:未知,1:手机号码,2:邮箱,3:用户(用户APP推送)',

PRIMARY KEY (`account\_id`),

KEY `user\_id` (`user\_id`),

CONSTRAINT `alarm\_account\_ibfk\_1` FOREIGN KEY (`user\_id`) REFERENCES `users` (`user\_id`) ON DELETE CASCADE ON UPDATE CASCADE

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

其中alarm\_account中的user\_id与[用户管理](#_用户管理)中的users表关联。需要对应的用户推送对应的消息。

## 用户管理

用户信息表users建表语句如下，其中分组信息只需要放在备注中即可。

CREATE TABLE `users` (

`user\_id` int(11) NOT NULL COMMENT '用户ID',

`user\_name` varchar(64) NOT NULL COMMENT '用户名',

`user\_pwd` varchar(64) NOT NULL COMMENT '用户密码',

`remarks` varchar(128) NOT NULL COMMENT '备注',

`create\_time` bigint(20) NOT NULL COMMENT '创建时间',

`invalid\_time` bigint(20) NOT NULL COMMENT '失效时间',

PRIMARY KEY (`user\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

权限信息表rights建表语句如下：

CREATE TABLE `rights` (

`right\_id` int(11) NOT NULL COMMENT '权限ID',

`right\_name` varchar(64) NOT NULL COMMENT '权限名',

`right\_describe` varchar(256) NOT NULL COMMENT '权限描述',

PRIMARY KEY (`right\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

用户与权限的关联为多对多关系，在表relation\_user\_right中：

CREATE TABLE `relation\_user\_right` (

`user\_id` int(11) NOT NULL COMMENT '用户ID',

`right\_id` int(11) NOT NULL COMMENT '权限ID',

PRIMARY KEY (`user\_id`,`right\_id`),

KEY `right\_id` (`right\_id`),

CONSTRAINT `relation\_user\_right\_ibfk\_1` FOREIGN KEY (`user\_id`) REFERENCES `users` (`user\_id`) ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT `relation\_user\_right\_ibfk\_2` FOREIGN KEY (`right\_id`) REFERENCES `rights` (`right\_id`) ON DELETE CASCADE ON UPDATE CASCADE

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

## 用户审计日志

用户审计日志类型建表语句如下：

CREATE TABLE `user\_log\_event\_type` (

`event\_type` int(11) NOT NULL COMMENT '事件类型',

`type\_name` varchar(64) not null comment '事件类型名称',

`type\_desc` varchar(256) not null comment '事件类型描述',

PRIMARY KEY (`event\_type`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

用户审计日志信息建表语句如下：

CREATE TABLE `user\_log` (

`id` int(11) NOT NULL COMMENT '审计日志ID',

`user\_id` int(11) NOT NULL COMMENT '用户ID',

`event\_time` bigint(20) NOT NULL COMMENT '事件时间',

`event\_type` int(11) NOT NULL COMMENT '事件类型',

`event\_result` int(11) NOT NULL COMMENT '操作结果，0成功，1失败',

`event\_verbose` varchar(512) not null comment '事件详细信息',

PRIMARY KEY (`id`),

CONSTRAINT `user\_log\_ibfk\_1` FOREIGN KEY (`user\_id`) REFERENCES `users` (`user\_id`) ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT `user\_log\_ibfk\_2` FOREIGN KEY (`event\_type`) REFERENCES `user\_log\_event\_type` (`event\_type`) ON DELETE CASCADE ON UPDATE CASCADE

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

# 其它接口

## 网络通知接口

控制中心和应用系统采用网络socket的方式进行通信，其中应用系统作为服务端，控制中心作为客户端，服务端监听端口可配置。每个控制中心与应用系统建立一条网络长连接通道。

网络通知采用字节流协议，本文接下来的描述中，数据传输字节序除特殊说明外都是小头序。每个网络收发单元数据如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| 协议版本2byte | | 压缩1byte | 加密1byte |
| 数据类型4byte | | | |
| 数据长度4byte | | | |
| 数据...... | | | |

协议版本：2字节固定值0x656C。

压缩：1字节，0:不压缩，1：压缩。当前版本不压缩。

加密：1字节，0:不加密，1:加密(L\_TEA)。既压缩又加密时，先压缩后加密。

数据类型：4字节，包含下面几个类型：

* 1：连接请求数据
* 2：心跳数据
* 3：心跳应答数据
* 4：其它业务数据

数据长度：4字节，用于决定后续紧跟的数据长度。

数据： 由数据长度字段决定具体长度，并在后面小节定义具体的数据内容。

### 连接请求

数据类型为1。在控制中心建立SOCKET完成后，必须发起连接建立请求，应用系统不需要应答。如果请求中的数据或控制中心序列号不合法，则应用系统要断开SOCKET连接。其数据方向为：控制中心-->应用系统。请求内容如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| 数据标识4byte | | | |
| 指令ID 4byte | | | |
| 控制中心序列号16byte | | | |

数据标识：4字节固定值0x6E636477。

指令ID：4字节随机值。

控制中心序列号：16字节，数据内容为长度15的字符串，第16字节为0。

### 心跳请求

数据类型为2，其数据方向为：控制中心-->应用系统。请求内容如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| 数据标识4byte | | | |
| 指令ID 4byte | | | |
| 控制中心序列号16byte | | | |

数据标识：4字节固定值0x71727462。

指令ID： 4字节随机值，值由心跳发起端随机生产，应答时需返回该值。

控制中心序列号：16字节，数据内容为长度15的字符串，第16字节为0。

### 心跳应答

数据类型为3，用于应答心跳请求。其数据方向为：应用系统-->控制中心。请求内容如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| 数据标识4byte | | | |
| 指令ID 4byte | | | |
| 控制中心序列号16byte | | | |

数据标识：4字节固定值0x71727462。

指令ID： 4字节值，与心跳请求的一致。

控制中心序列号：16字节，与心跳请求一致。

### 业务数据

数据类型为4，业务数据请求request头部如下：

该指令用于主动发起功能请求或提交数据请求，可以由控制中心发给检测前端，也可以由检测前端发给控制中心。结构如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| 指令标识 4byte | | | |
| 指令ID 4byte | | | |
| 控制中心序列号16 byte | | | |
| 功能号 2byte | | 操作号 2byte | |
| 预留 1 4byte | | | |
| 预留2 4byte | | | |
| 指令参数长度 4byte | | | |
| 指令参数...... | | | |

指令标识：占4个字节，默认值0x00000001。

指令ID：占4个字节，由系统决定。

控制中心序列号：占16个字节，由系统决定。值为长度15的字符串。

功能号：占2个字节，对应系统的业务功能。

操作号：占3个字节，对应系统业务功能下的处理分支。

预留1：占4个字节，暂时无用。

预留2：占4个字节，暂时无用。

指令参数长度：占4个字节，指令紧跟的指令参数长度。

指令参数：由指令参数长度字段决定具体长度。

业务数据响应用于应答request指令，返回request指令的执行结果。其头部结构如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| 指令标识 4byte | | | |
| 指令ID 4byte | | | |
| 控制中心序列号16 byte | | | |
| 功能号 2byte | | 操作号 2byte | |
| 状态码 4byte | | | |
| 预留1 4byte | | | |
| 预留2 4byte | | | |
| 指令参数长度 4byte | | | |
| 指令参数...... | | | |

指令标识：占4个字节，默认值0x00000002。

指令ID：占4个字节，使用request指令的指令ID。

控制中心序列号：占16个字节，由系统决定。值为长度15的字符串。

功能号：占2个字节，对应系统的业务功能。

操作号：占2个字节，对应系统业务功能下的处理分支。

状态码：占4个字节，为request指令的执行状态，0为成功，非0为失败。

预留1：占4个字节，暂时无用。

预留2：占4个字节，暂时无用。

指令参数长度：占4个字节，指令紧跟的指令参数长度。

指令参数：由指令参数长度字段决定具体长度的内容。

其中功能号2字节，包含下面一些类型具体类型：

|  |  |  |
| --- | --- | --- |
| 业务号 | 功能 | 请求方向 |
| 0x2001 | 检测前端状态变化通知 | 控制中心-->应用系统 |
| 0x2002 | 告警事件发生通知 | 控制中心-->应用系统 |
| 0x2101 | 录入AP和终端信息更新通知 | 应用系统-->控制中心 |
| 0x2102 | 修改威胁事件响应策略通知 | 应用系统-->控制中心 |
| 0x2103 | 检测前端升级通知 | 应用系统-->控制中心 |

指令参数具体如下。

#### 检测前端状态变化通知

请求发送方向为：控制中心-->应用系统，业务数据内容为18字节：

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| 状态变化的检测前端的序列号16byte | | | |
| 当前状态 2byte | |

状态变化的检测前端的序列号：为15字节字符串，第16字节为0。

当前状态：当前状态标志，1：在线 0：下线。

应答发送方向反向，没有业务数据内容。

#### 告警事件发生通知

请求发送方向为：控制中心-->应用系统，没有业务数据内容。

应答发送方向反向，没有业务数据内容。

#### 录入AP和终端信息更新通知

请求发送方向为：应用系统-->控制中心，没有业务数据内容。

应答发送方向反向，没有业务数据内容。

#### 修改威胁事件响应策略通知

请求发送方向为：应用系统-->控制中心，没有业务数据内容。

应答发送方向反向，没有业务数据内容。

#### 升级通知

请求发送方向为：应用系统-->控制中心，业务数据内容如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| 检测前端/控制中心升级ZIP文件的相对路径，以0结尾 | | | |

应答发送方向反向，没有业务数据内容。

该升级文件存储要注意不要覆盖以前的升级文件。升级结束后，由控制中心删除该文件。

### 数据示例

#### 连接请求示例

总体是36字节，其中网络单元头部为12字节，协议数据24字节，以Java 的 byte [] 为例，应用系统收到的36字节如下：

byte[] buf = {

0x6c, 0x65, // 网络单元--协议版本，小头序0x656c

0x00, // 网络单元--压缩：0，不压缩

0x00, // 网络单元--加密：0，不加密

0x01, 0x00, 0x00, 0x00, // 网络单元--数据类型：小头序0x00000001，也就是1--连接请求

0x18, 0x00, 0x00, 0x00, // 网络单元--数据长度：小头序0x00000018，也就是24字节

0x77, 0x64, 0x63, 0x6e, // 协议数据--数据标识：小头序0x6e636477

0x11, 0x22, 0x33, 0x44, // 协议数据--指令ID：4字节下头序 0x44332211

0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x38, 0x39, 0x30, 0x31, 0x32, 0x33, 0x34, 0x00 // 协议数据--控制中心序列号：转换为ASCII码字符串“012345678901234”

};

#### 升级通知请求

总体是63字节，其中网络单元头部为12字节，业务数据头部40字节，升级通知请求数据11字节，以Java 的 byte [] 为例，应用系统发送的63字节如下：

byte[] buf = {

0x6c, 0x65, // 网络单元--协议版本，小头序0x656c

0x00, // 网络单元--压缩：0，不压缩

0x00, // 网络单元--加密：0，不加密

0x01, 0x00, 0x00, 0x00, // 网络单元--数据类型：小头序0x00000001，也就是1--连接请求

0x33, 0x00, 0x00, 0x00, // 网络单元--数据长度：小头序0x00000033，也就是51字节

0x01, 0x00, 0x00, 0x00, // 业务数据头--请求指令标识：小头序0x00000001

0x03, 0x00, 0x00, 0x00, // 业务数据头--指令ID，随机值，小头序0x00000003

0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x38, 0x39, 0x30, 0x31, 0x32, 0x33, 0x34, 0x00, // 业务数据头--控制中心序列号：ASCII码字符串“012345678901234”

0x03, 0x21, // 业务数据头--功能号，小头序 0x2103

0x00, 0x00 // 业务数据头--操作号，目前无用，0x0000

0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, // 业务数据头--预留一、预留二

0x0B, 0x00, 0x00, 0x00, // 业务数据头--数据长度，小头序0x0000000B，也就是11

0x75, 0x70, 0x64, 0x61, 0x74, 0x65, 0x2E, 0x7A, 0x69, 0x70, 0x00 // 业务数据--11个字节相对路径“update.zip”

};

#### 升级通知应答

总体是56字节，其中网络单元头部为12字节，业务数据头部44字节，没有数据。以Java 的 byte [] 为例，应用系统发送的56字节如下：

byte[] buf = {

0x6c, 0x65, // 网络单元--协议版本，小头序0x656c

0x00, // 网络单元--压缩：0，不压缩

0x00, // 网络单元--加密：0，不加密

0x01, 0x00, 0x00, 0x00, // 网络单元--数据类型：小头序0x00000001，也就是1--连接请求

0x2c, 0x00, 0x00, 0x00, // 网络单元--数据长度：小头序0x0000000c，也就是44字节

0x02, 0x00, 0x00, 0x00, // 业务数据头--响应指令标识：小头序0x00000002

0x03, 0x00, 0x00, 0x00, // 业务数据头--指令ID，同请求，小头序0x00000003

0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x38, 0x39, 0x30, 0x31, 0x32, 0x33, 0x34, 0x00, // 业务数据头--控制中心序列号：ASCII码字符串“012345678901234”

0x03, 0x21, // 业务数据头--功能号，小头序 0x2103

0x00, 0x00 // 业务数据头--操作号，目前无用，0x0000

0x00, 0x00, 0x00, 0x00, //业务数据头--状态码，小头序 0x00000000，也就是0，成功

0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, // 业务数据头--预留一、预留二

0x00, 0x00, 0x00, 0x00 // 业务数据头--数据长度，小头序0x00000000，也就是0

};

# 用户界面

## 登陆

1. 登陆出错时只提示“用户名或密码错误”，不提示更多细节。
2. 登陆后如果一段时间不操作，则退出登陆，需要重新登陆。
3. 如果在一台电脑连续登陆出错超过一定次数，则锁定这台电脑当前在一段时间内不能登陆。
4. 如果一个用户连续登陆出错超过一定次数，则锁定该用户在一段时间内不能登陆。
5. 登陆成功后根据权限展示相应的功能块。

用户名密码及权限见数据库表[用户管理](#_用户管理)。用户登陆密码加密存储，加密方式为L\_TEA，数据库中存储加密后BASE64编码内容到数据库中。

## 检测前端设备管理

1. 登陆成功之后主页查看检测前端设备信息。系统可以展示检测前端设备的名称、当前版本、安装时间、当前在线状态、描述信息；离线的检测前端显示最后活动时间。信息从数据库表[device\_monitor](#_检测前端信息表device_monitor)中获取。
2. 可以选择单台设备修改备注信息。

检测前端的数量级为数百（目前不超过200），建议将检测前端设备状态在系统启动时读取到缓存，状态修改时，控制中心会通过[检测前端状态变化通知](#_检测前端状态变化通知)告知应用系统，然后应用系统更新缓存信息，并更新页面。

## 预览周边WIFI环境

### 预览AP信息

可以预览周边的WIFI环境。通过列表形式展示所有AP信息，展示内容包括：

* MAC地址
* 信道
* 信号强度
* 保护方式
* WPS状态
* 是否开启SSID广播
* 硬件制造商
* 有几个连接的终端：计算方式为，使用scan\_station表中mac\_ap等于scan\_ap中的mac字段，则终端连接在对应的AP上；如果mac\_ap为空，则终端没有连接任何AP。

对于没有连接AP的终端，展示终端总数。

展示的刷新频率目前暂定5秒。

我们其它系统的AP展示，仅做参考。



### 展开AP预览终端

可以展开终端数不为0的AP或未连接AP的终端，然后展示所有的终端信息。终端信息包括：MAC地址、硬件制造商、信号强度。

### 将AP或终端加入名单

对单个AP或终端进行选择，可以设置为信任AP、邻居AP、访客AP、自组网AP、忽略MAC、保护终端等。数据库表见[录入AP和终端信息](#_录入AP和终端信息)。

增加信任AP、邻居AP、访客AP、自组网AP时，填写的字段包括：

* MAC地址
* SSID
* 备注（可以为空）

增加忽略MAC时，填写的字段包括：

* MAC地址
* 名称（可以为空）
* 备注（可以为空）
* 失效时间

增加保护终端时，填写的字段包括：

* MAC地址
* 姓名（可以为空）
* 备注（可以为空）
* 电话（可以为空）
* 部门（可以为空）

增加邻居终端、访客终端时，填写的字段包括：

* MAC地址
* 姓名（可以为空）
* 备注（可以为空）
* 电话（可以为空）

增加自组网终端时，填写的字段包括：

* MAC地址
* 名称（可以为空）
* 备注（可以为空）

增加名单时，将数据加入数据库，然后通过[录入AP和终端信息更新通知](#_录入AP和终端信息更新通知)向控制中心通知。

**注意：任何一个MAC地址只能被加入唯一的名单。**

### 周边AP安全报告

系统支持对周边除忽略AP外的所有AP生成一份安全评估报告，报告内容包括：

AP总量，终端总量，信道分布， AP的安全性，含加密和保护方式、是否开启WPS等，针对不安全的AP，提供解决方法建议。数据库表使用[周边AP和终端信息](#_周边AP和终端信息)。

其中安全性分析方式：

* 对于privacy为0,1,3,5,7的AP，提示为有极大安全风险，建议开启WPA2加密；
* 对于privacy为2，encrypt\_way为4的WIFI，提示有较小的风险，建议设置为AES加密；
* 对于开启WPS的，提示有一定安全风险，建议关闭WPS；
* 对于使用其它AP，提示为安全AP。

### 下载生成的安全报告

系统支持生成周边安全报告的word文档，并提供下载。

## 管理MAC名单

1. 用户可以手动增加信任AP、邻居AP、访客AP、自组网AP、忽略MAC、保护终端等。
2. 用户可以修改这些数据。修改时，不能修改MAC地址。
3. 用户可以删除这些数据。
4. 需要提供批量操作的功能。

管理单条记录后，修改数据库，然后通过[录入AP和终端信息更新通知](#_录入AP和终端信息更新通知)向控制中心通知。

## 事件告警

### 查看系统支持检测的威胁事件

* 系统支持展示所有的威胁事件类型，展示的信息包括：详细的描述，分析，危害等级，建议的解决方案和对策，使用数据库表见[威胁事件定义](#_攻击事件定义)。
* 用户可以选择忽略或激活某类型威胁事件。这样将数据库中[威胁事件定义](#_攻击事件定义)中的内容设置为“不激活”或“激活”，然后通过[修改威胁事件响应策略通知](#_修改攻击事件响应策略通知)向控制中心发送通知。
* 用户可以修改单一威胁事件的响应策略。修改数据库，然后通过[修改威胁事件响应策略通知](#_修改攻击事件响应策略通知)向控制中心发送通知。注意：不激活的策略不支持修改响应策略。
* 用户可以恢复单一威胁事件或所有威胁事件的默认响应策略。修改数据库，然后通过[修改威胁事件响应策略通知](#_修改攻击事件响应策略通知)向控制中心发送通知。

### 显示实时告警信息

系统从控制中心接收[告警事件发生通知](#_告警事件发生通知)，然后从数据库[告警事件记录](#_攻击事件记录)去获取还没有读取的告警信息，然后推送给用户。用户读取后，将read\_state置为1。

告警信息需要显示：

* 事件发生时间
* 攻击类型
* 受攻击目标
* 危害等级
* 事件详细描述以及解决方案建议

其中对应的危害等级、解决方案建议都从[威胁事件定义](#_攻击事件定义)中获取。

### 历史告警事件查询

可以根据时间段、告警类型、告警MAC查询历史告警事件，并可以根据时间进行排序。显示的信息同[显示实时告警信息](#_显示实时告警信息)。

## 用户管理

需要实现两类用户的功能，用户分两个组。

普通用户权限包括：

* 登陆和退出系统
* 查看和备注检测前端
* 管理告警事件类型
* 管理MAC名单
* 接收告警信息
* 查询历史事件告警
* 修改个人密码

管理员还拥有几个额外的权限：

* 设置登陆超时时间，设置登陆失败超限次数，登陆超过限制次数后的锁定时间。
* 用户管理（新增、修改、删除、重置其他用户密码）
* 查看审计日志
* 升级设备

注意：不能修改默认管理员admin的任何信息。也不能重置默认管理员admin的密码。

## 审计日志查看

查看审计日志。用户的登录和注销，用户配置策略（设置单个AP或终端为特殊对象，生成和下载安全报告，设置忽略MAC或忽略事件），用户查看入侵检测结果，对用户进行增加、删除和属性修改的操作。

可以根据操作用户、操作类型、操作时间段进行过滤和排序。

日志信息形式为：事件发生的时间，事件的类型，操作者身份，操作结果。

审计日志类型和审计日志数据库表见[用户审计日志](#_用户审计日志)。

## 系统升级

* 系统支持让管理员升级前端设备和控制中心。由用户选择上传一个升级文件，升级文件为一个zip文件，解压后，zip中分为两个文件，其一为前端的升级文件，其二为更新信息readme.ini。其中readme.ini文件描述了当前升级包的升级信息，需要展示给用户，由其确认升级。然后应用服务将zip文件保存在一个路径，然后将该路径通过[升级通知](#_升级通知)发送给控制中心。
* 系统支持让管理员升级告警事件数据库。告警事件数据库为只增不减的项目，上传的更新包为L\_TEA加密的SQL文件，解密执行即可。