# XP安全防护功能列表

1、需求

1.1、管理中心：

1.1.1、安全管理员登陆

|  |  |  |  |
| --- | --- | --- | --- |
| 功能 |  |  |  |
| 客户端列表 |  |  | 列出所有登陆的客户端机器IP  点击后，进入每个单独客户端。默认显示第一个客户端信息。 |
| 客户端分组 |  | 组增加属性，鼠标悬停时候，会显示属性 | 曾管理中心可以把登陆的IP分成不同的组，要有添加删除组，添加IP到组的功能 |
| 策略下发 |  |  | 导出规则，下发到选定的IP |
| 事件总览 | 已保护主机多少天 |  |  |
| 拦截疑似攻击次数 |  |  |
| 处理账户变动次数 |  |  |
| 当前基本防护状态 |  | 防护|监控 |
| 当前增强防护状态 |  | 防护|监控 |
| 当前账户监视状态 |  | 开启|关闭 |
| 当前账户安全级别 |  | 高中低|自定义 |
| 状态图 |  | 简易的事件图形展示 |
| 安全防护 | 模式设置 |  | 防护|监控 |
| 系统目录及文件保护 |  | 开启|关闭 |
| 系统启动文件保护 |  | 开启|关闭 |
| 防止格式化 |  | 开启|关闭 |
| 防止关键进程被杀死 |  | 开启|关闭 |
| 防止篡改系统服务 |  | 开启|关闭 |
| 规则导出 |  |  |
| 增强防护 | 模式设置 |  | 防护|监控 |
| 防止服务被添加 |  | 开启|关闭 |
| 防止自动运行恶意程序 |  | 开启|关闭 |
| 防止添加开机自启动项 |  | 开启|关闭 |
| 防止磁盘被直接读写 |  | 开启|关闭 |
| 禁止创建.exe文件 |  | 开启|关闭 |
| 防止驱动被加载 |  | 开启|关闭 |
| 防止进程被注入 |  | 开启|关闭 |
| 规则导出 |  |  |
| 账户安全 | 状态 |  | 监控开启|关闭 |
| 账户策略设置 |  | 自定义|高中低 |
| 密码复杂度 |  | 开启|关闭 |
| 密码最小长度 |  | 字符个数 |
| 锁定时长 |  | 错误登陆间隔[已失效？] |
| 锁定次数 |  | 错误登陆次数 |
| 最短使用期限 |  | 密码使用指定时长后可修改(天) |
| 最长使用期限 |  | 使用多久后必须修改（天） |
| 强制密码历史 |  | 旧密码不重复个数[已失效？] |
| 事件查看 | 安全事件 |  |  |
| 账户事件 |  |  |
| 系统设置 | 账号设置 |  | 登陆密码修改 |
|  | 黑名单 |  | 添加|删除 |
|  | 白名单 |  | 添加|删除 |

1.1.2、审计管理员

|  |  |  |  |
| --- | --- | --- | --- |
| 系统设置 |  |  | 登陆密码 |
| 首页展示统计 |  |  | 单个IP |
| 操作日志 |  |  |  |
| 违规日志 |  |  | 安全事件|违规事件 |
| 生成报表 |  |  | 单独标签页，里面可选IP组或者多个IP生成报表 |

1.2、客户端（只有管理员账号）

1、管理中心界面没有客户端列表、客户端分组、策略下发。

2、客户端安装时候，要指定管理中心的IP，启动时候，连接到管理中心。

3、授权，提取客户机的硬件信息，生成一个码，然后我们单独有程序计算序列号，生成一个序列号，提供给安装机器。

2、软件设计

2.1、共用约定

2.1.1、编码

统一UTF8，给对方的数据必须是UTF8编码的。Http对外返回的数据为 UTF8编码后转成JSON。

* 1. 详细功能

2.2.1、卸载密码

与安全管理员登录密码一致。卸载时候，需要验证，错误情况下，不允许卸载。

2.2.2、授权

授权针对客户端，提取客户端机器码，用授权软件生成一个注册码，然后注册。

客户端未授权，也可以正常登录。但是登录后，首页会有未注册提示。点击可以进行注册。管理中心，打开单个的客户端，一样可以注册。

* 1. 接口

【注意】所有的URL的传入参数，和结果都是经过json编码的。

2.3.1、获取管理中心IP和端口

[注] 这个接口不需要验证UserTokey

URL：center/getaddr

参数：无

返回值：

{

"CenterIP" : "192.168.1.100",

"CenterPort" : 8080

}

2.3.2、客户端登录

[注]

登录后会返回一个UserTokey字符串，每次登录时候得到到UserTokey都会不同。在调用以后的接口时候，都需要用该字符串来做验证。如果验证失败，则返回错误，错误信息为：提示需要登录。

URL：login

参数：data = {

"User":"Admin",

"Password":"123456",

"CenterIP":"192.168.1.1",

"CenterPort":"8080"

}

返回值：

{

"Status":1,

"Errmsg":"登录成功",

"User":"admin",

"Usertype":2,

"UserTokey":"f5ab1c6a"

}

用户名和默认密码：

|  |  |  |  |
| --- | --- | --- | --- |
| 用户名 | 密码 | 类型(int) | 备注 |
| CenterAdmin | lzf:24324962@qq.com | 1 | 管理中心用户 |
| Admin | 123456 | 2 | 操作管理员 |
| Audit | 123456 | 3 | 审计管理员 |

2.3.3、系统设置-修改密码

URL：/sys/changepassword

参数1：UserTokey="f5ab1c6a"

参数2：data = {

"User":"Admin",

"OldPwd":"123456",

"NewPwd":"654321"

}

返回值：

{

"Status":1,

"Errmsg":"密码修改成功"

}

2.3.4、系统设置 - 白名单

**1、白名单添加**

URL：sys/addwhite

参数1：UserTokey="f5ab1c6a"

参数2：data={

"File":"C://test"

}

返回值：

{

"Status":1,

"Errmsg":"添加白名单成功"

}

**2、白名单删除**

URL：sys/delwhite

参数1：UserTokey="f5ab1c6a"

参数2：data={

"File":"C://test"

}

返回值：

{

"Status":1,

"Errmsg":"删除白名单成功"

}

**3、获取白名单数量**

URL：sys/totlewhite

参数1：UserTokey="f5ab1c6a"

返回值：

{

"Status":1,

"Errmsg":"查询白名单总数成功",

"TotCnt":10

}

**4、查询白名单内容**

URL：sys/querywhite

参数1：UserTokey="f5ab1c6a"

参数2：data={

"Start":0,

"Length":10

}

返回值：

{

"Status":1,

"Errmsg":"查询白名单成功",

"Files":["C:\\test","C:\\test\\dd"]

}

2.3.5、系统设置 – 黑名单

**1、黑名单添加**

URL：sys/addblack

参数1：UserTokey="f5ab1c6a"

参数2：data={

"File":"C://test"

}

返回值：

{

"Status":1,

"Errmsg":"添加黑名单成功"

}

**2、黑名单删除**

URL：sys/delblack

参数1：UserTokey="f5ab1c6a"

参数2：data={

"File":"C://test"

}

返回值：

{

"Status":1,

"Errmsg":"删除黑名单成功"

}

**3、获取黑名单数量**

URL：sys/totleblack

参数1：UserTokey="f5ab1c6a"

返回值：

{

"Status":1,

"Errmsg":"查询黑名单总数成功",

"TotCnt":10

}

**4、查询黑名单内容**

URL：sys/queryblack

参数1：UserTokey="f5ab1c6a"

参数2：data={

"Start":0,

"Length":10

}

返回值：

{

"Status":1,

"Errmsg":"查询黑名单成功",

"Files":["C:\\test","C:\\test\\dd"]

}

2.3.6、安全防护 – 基本防护

// 基本防护 - 设置 - 请求含义

**type** SafeBaseSetRequest **struct** {

    Mode int // 模式：0:监视模式 1:防护模式

    WinDir int // 系统文件及目录保护状态 0:关闭 1:开启

    WinStart int // 系统启动文件保护状态 0:关闭 1:开启

    WinFormat int // 防止格式化磁盘状态 0:关闭 1:开启

    WinProc int // 防止系统关键进程被杀死 0:关闭 1:开启

    WinService int // 防止篡改系统服务 0:关闭 1:开启

}

**1、设置**

URL：safe/baseset

参数1：UserTokey="f5ab1c6a"

参数2：data={

"Mode":0,

"WinDir":0,

"WinStart":0,

"WinFormat":0,

"WinProc":0,

"WinService":0

}

返回值：

{

"Status":1,

"Errmsg":"基本防护设置成功",

}

**2、获取**

URL：safe/baseget

参数1：UserTokey="f5ab1c6a"

返回值：

{

"Status":1,

"Errmsg":"获取基本防护设置成功",

"Mode":0,

"WinDir":0,

"WinStart":0,

"WinFormat":0,

"WinProc":0,

"WinService":0

}

**3、导出**

URL：safe/basesave

参数1：UserTokey="f5ab1c6a"

返回值：

{

"Status":1,

"Errmsg":"基本防护规则导出成功",

"Config":"### 配置文件\n\n[INFO]\nName = SafeBase\n\n[CONFIG]\nMode = \u0000\nWinDir = \u0000\nWinStart = \u0000\nWinFormat = \u0000\nWinProc = \u0000\nWinService = \u0000\n\n"

}

2.3.7、安全防护 – 增强防护

// 增强防护 - 设置 - 请求含义

**type** SafeHighSetRequest **struct** {

    Mode int // 模式：0:监视模式 1:防护模式

    AddService int // 防止服务被添加 0:关闭 1:开启

    AutoRun int // 防止自动运行恶意程序 0:关闭 1:开启

    AddStart int // 防止添加开机启动项 0:关闭 1:开启

    ReadWrite int // 防止磁盘直接读写 0:关闭 1:开启

    CreateExe int // 防止创建EXE文件 0:关闭 1:开启

    LoadSys int // 防止驱动被加载 0:关闭 1:开启

    ProcInject int // 防止进程被注入 0:关闭 1:开启

}

**1、设置**

URL：safe/highset

参数1：UserTokey="f5ab1c6a"

参数2：data={

"Mode":0,

"AddService":0,

"AutoRun":0,

"AddStart":0,

"ReadWrite":0,

"CreateExe":0,

"LoadSys":0,

"ProcInject":0

}

返回值：

{

"Status":1,

"Errmsg":"增强防护设置成功",

}

**2、获取**

URL：safe/highget

参数1：UserTokey="f5ab1c6a"

返回值：

{

"Status":1,

"Errmsg":"获取增强防护设置成功",

"Mode":0,

"AddService":0,

"AutoRun":0,

"AddStart":0,

"ReadWrite":0,

"CreateExe":0,

"LoadSys":0,

"ProcInject":0

}

**3、导出**

URL：safe/highsave

参数1：UserTokey="f5ab1c6a"

返回值：

{

"Status":1,

"Errmsg":"增强防护规则导出成功",

"Config":"### 配置文件\n\n[INFO]\nName = SafeHigh\n\n[CONFIG]\nMode = \u0000\nAddService = \u0000\nAutoRun = \u0000\nAddStart = \u0000\nReadWrite = \u0000\nCreateExe = \u0000\nLoadSys = \u0000\nProcInject = \u0000\n\n"

}

2.3.8、账户安全

// 账户安全 - 设置 - 请求含义

**type** AccountSetRequest **struct** {

    Mode int // 模式：0:关闭 1:开启

    SafeLev int // 账户策略设置 0:自定义 1:低级 2:中级 3:高级

    PwdComplex int // 密码复杂度 0:关闭 1:开启

    PwdMinLen int // 密码最小长度(字符个数)

    PwdUsedMin int // 最短使用期限(天)

    PwdUsedMax int // 最长使用期限(天)

    PwdOldNum int // 强制密码历史次数(次)

    AccountTimes int // 账户锁定次数(无效登录次数)

    AccountMinute int // 账户锁定时长(分钟)

}

**1、设置**

URL：account/set

参数1：UserTokey="f5ab1c6a"

参数2：data={

"Mode":0,

"SafeLev":0,

"PwdComplex":0,

"PwdMinLen":0,

"PwdUsedMin":0,

"PwdUsedMax":0,

"PwdOldNum":0,

"AccountTimes":0,

"AccountMinute":0

}

返回值：

{

"Status":1,

"Errmsg":"账户安全设置成功",

}

**2、获取**

URL：account/get

参数1：UserTokey="f5ab1c6a"

返回值：

{

"Status":1,

"Errmsg":"获取账户安全设置成功",

"Mode":0,

"SafeLev":0,

"PwdComplex":0,

"PwdMinLen":0,

"PwdUsedMin":0,

"PwdUsedMax":0,

"PwdOldNum":0,

"AccountTimes":0,

"AccountMinute":0

}

**3、导出**

URL：account/save

参数1：UserTokey="f5ab1c6a"

返回值：

{

"Status":1,

"Errmsg":"账户安全规则导出成功",

"Config":"### 配置文件\n\n[INFO]\nName = Account\n\n[CONFIG]\nMode = \u0000\nSafeLev = \u0000\nPwdComplex = \u0000\nPwdMinLen = \u0000\nPwdUsedMin = \u0000\nPwdUsedMax = \u0000\nPwdOldNum = \u0000\nAccountTimes = \u0000\nAccountMinute = \u0000\n\n"

}

2.3.9、日志获取 – 系统日志

**1、系统运行日志数量**

URL：log/systotle

参数1：UserTokey="f5ab1c6a"

返回值：

{

"Status":1,

"Errmsg":"查询:系统日志数量成功",

"Count":10

}

**2、系统日志记录**

URL：log/sysquery

参数1：UserTokey="f5ab1c6a"

参数2：data = {

    KeyWord string // 关键词

    TimeStart string // 起始时间

    TimeStop string // 结束时间

    Start int // 日志起始位置

    Length int // 日志条数

}

返回值：

{

"Status":1,

"Errmsg":"查询:系统日志数量成功",

     "LogArray "： [ // 日志数组

{

Uname string // 用户名

Op string // 操作

Info string // 内容

Result string // 结果

Time string // 时间

},

{

Uname string // 用户名

Op string // 操作

Info string // 内容

Result string // 结果

Time string // 时间

},

.......

}

2.3.9、日志获取 – 安全日志

**1、安全日志数量**

URL：log/eventtotle

参数1：UserTokey="f5ab1c6a"

返回值：

{

"Status":1,

"Errmsg":"查询:安全日志数量成功",

"Count":10

}

**2、安全日志记录**

URL：log/eventquery

参数1：UserTokey="f5ab1c6a"

参数2：data = {

    KeyWord string // 关键词

    TimeStart string // 起始时间

    TimeStop string // 结束时间

    Start int // 日志起始位置

    Length int // 日志条数

}

返回值：

{

"Status":1,

"Errmsg":"查询:系统日志数量成功",

     "LogArray "： [ // 日志数组

{

Module string // 模块名称

Mode string // 模式 (防护模式 | 监控模式)

User string // 用户名

Sub string // 主体进程

Obj string // 对象

Op string // 操作

Ret string // 操作结果

Time string // 时间

},

{

Module string // 模块名称

Mode string // 模式 (防护模式 | 监控模式)

User string // 用户名

Sub string // 主体进程

Obj string // 对象

Op string // 操作

Ret string // 操作结果

Time string // 时间

},

.......

}