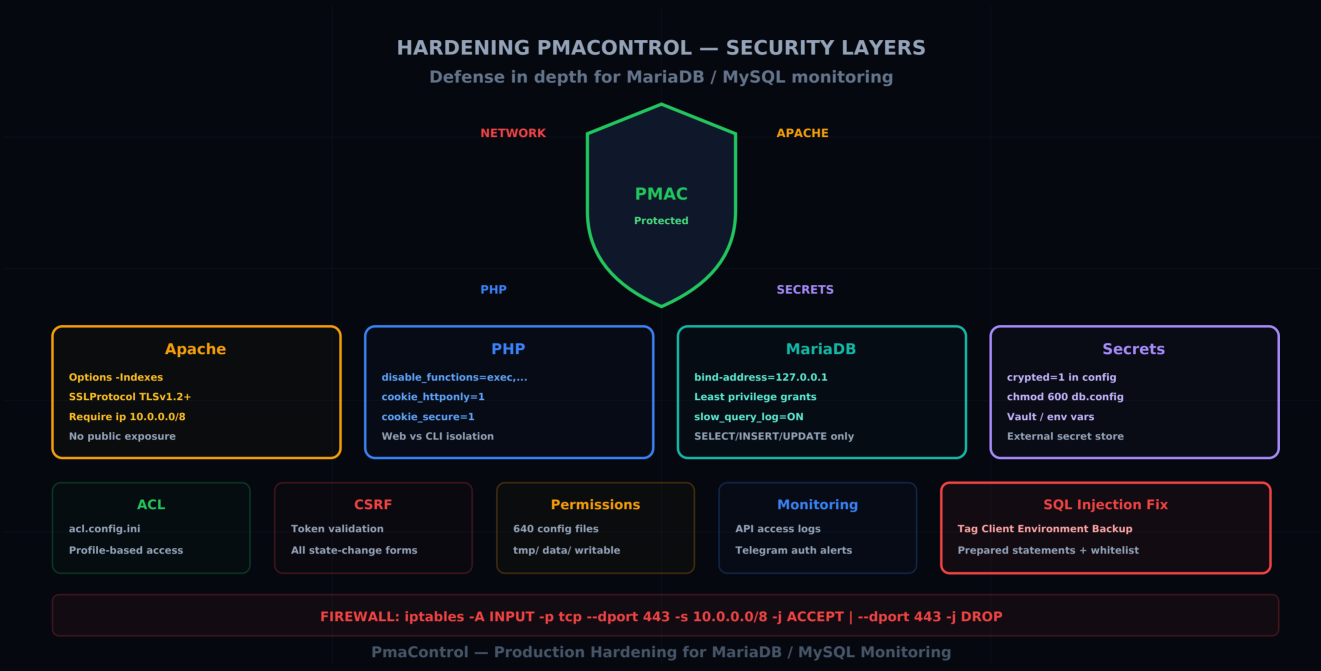
生产环境加固 PmaControl: 完整安全指南

Aurélien LEQUOY · 2026年4月13日

PMACONTROL SECURITY HARDENING APACHE PHP MARIADB



PmaControl 掌握着王国的钥匙

PmaControl 监控您的生产环境 MariaDB / MySQL 服务器。它存储连接凭据、SSH 密钥、性能指标和数据库结构。如果攻击者攻破了 PmaControl，他们可能获得对

本指南详细介绍了在将 PmaControl 投入生产之前应采取的加固措施。它来源于一次内部安全审计，涵盖了每一层：Apache、PHP、MariaDB、密钥、ACL、CSRF、文件权限和监控。

第 1 层：Apache

禁用目录列表

默认情况下，当不存在索引文件时，Apache 会显示目录内容。这是一种信息泄露：

```
<Directory /srv/www/pmacontrol>
    Options -Indexes
    AllowOverride All
    Require all granted
</Directory>
```

`-Indexes` 是不可协商的。没有它，攻击者可以浏览项目结构并找到配置文件、日志和数据转储。

强制 HTTPS

PmaControl 在 HTTP 请求中以明文传输凭据。没有 HTTPS，网络上的攻击者可以拦截它们：

```
<VirtualHost *:80>
    ServerName pmacontrol.internal.company.com
    Redirect permanent / https://pmacontrol.internal.company.com/
</VirtualHost>

<VirtualHost *:443>
    ServerName pmacontrol.internal.company.com
    SSLEngine On
    SSLCertificateFile /etc/ssl/certs/pmacontrol.pem
    SSLCertificateKeyFile /etc/ssl/private/pmacontrol.key

    # 仅使用现代 TLS
    SSLProtocol -all +TLSv1.2 +TLSv1.3
    SSLCipherSuite HIGH:!aNULL:!MD5:!3DES

    DocumentRoot /srv/www/pmacontrol
</VirtualHost>
```

限制为内部网络

PmaControl 绝不应该暴露在互联网上。将访问限制在内部网络：

```
<Location />
    Require ip 10.0.0.0/8
    Require ip 172.16.0.0/12
    Require ip 192.168.0.0/16
</Location>
```

或者更好的方案：将 PmaControl 放在 VPN 后面，完全不通过公共 Apache 暴露。

删除默认虚拟主机

Apache （`000-default.conf`）会响应对服务器 IP 的任何请求。删除它：

```
a2dissite 000-default.conf
systemctl reload apache2
```

安全头

添加 HTTP 安全头:

```
Header always set X-Content-Type-Options "nosniff"
Header always set X-Frame-Options "SAMEORIGIN"
Header always set X-XSS-Protection "1; mode=block"
Header always set Referrer-Policy "strict-origin-when-cross-origin"
Header always set Content-Security-Policy "default-src 'self'; script-src 'self' 'unsafe-
inline'; style-src 'self' 'unsafe-inline'"
```

第 2 层: PHP

禁用危险函数

PmaControl 在某些操作(SSH)采集中使用 `shell_exec()`。解决方案不是全局禁用它们，而是隔离需要它们的工作进程。

对于 Web 虚拟主机（界面）：

```
; php.ini 或 DocumentRoot 中的 .user.ini
disable_functions = exec,shell_exec,system,passthru,popen,proc_open
expose_php = Off
```

对于 CLI 工作进程（Aspirateur、Listener）：

```
; php-cli.ini — 这些工作进程需要 shell_exec
disable_functions =
```

这种隔离确保 Web 界面无法执行系统命令，即使攻击者发现了漏洞。

安全会话

```
session.cookie_httponly = 1
session.cookie_secure = 1
session.cookie_samesite = Strict
session.use_strict_mode = 1
session.name = PMACSESSID
```

`cookie_httponly` 防止 JavaScript 访问会话 Cookie，减少 XSS 攻击。`cookie_secure` 强制仅通过 HTTPS 发送 Cookie。`cookie_samesite = Strict` 防止基本的 CSRF 攻击。

限制上传和执行

```
upload_max_filesize = 2M
post_max_size = 8M
max_execution_time = 30
max_input_time = 60
memory_limit = 256M
```

PmaControl 不需要大文件上传。限制这些参数以减少攻击面。

隐藏 PHP 版本

```
expose_php = Off
```

这会从 HTTP 头中删除 `X-Powered-By: PHP/8.x` 头。

第 3 层: MariaDB

限制 PmaControl 用户权限

安装后，PmaControl 用户通常拥有所有权限。需要限制它们：

```
-- 撤销过多权限
REVOKE ALL PRIVILEGES ON *.* FROM 'pmacontrol'@'localhost';

-- 仅授予必需的权限
GRANT SELECT, INSERT, UPDATE, DELETE ON pmacontrol.* TO 'pmacontrol'@'localhost';
GRANT SELECT ON performance_schema.* TO 'pmacontrol'@'localhost';
GRANT REPLICATION CLIENT ON *.* TO 'pmacontrol'@'localhost';
GRANT PROCESS ON *.* TO 'pmacontrol'@'localhost';

FLUSH PRIVILEGES;
```

最小权限原则：PmaControl 只需要读取指标并写入自己的数据库。

绑定到 Localhost

PmaControl 数据库应仅监听本地接口：

```
[mysqld]
bind-address = 127.0.0.1
```

如果 PmaControl 和其数据库在同一台服务器上（典型配置），没有理由监听网络。

启用敏感查询日志

```
[mysqld]
general_log = OFF          # 在生产环境中过于冗长
slow_query_log = ON
long_query_time = 1
log_error = /var/log/mysql/error.log
```

慢查询日志有助于检测可能表明 SQL 注入被利用的异常查询。

第 4 层：密钥管理

加密凭据

PmaControl 将接凭 `config.ini.php` 中。该文件支持加密：

```
; configuration/db.config.ini.php
[default]
driver = mysql
host = 127.0.0.1
port = 3306
login = pmacontrol
password = "ENCRYPTED_VALUE_HERE"
database = pmacontrol
crypted = 1
```

`crypted=1` 标志告诉 PmaControl 在运行时解密密码。加密密钥与配置文件分开存储。

使用外部密钥存储

对于关键的生产部署，将密钥外部化：

- **Vault**（HashiCorp）：PmaControl 可以通过 API 读取密钥
- **AWS Secrets Manager**、**GCP Secret Manager**：用于云部署
- 环境变量：最基本的可行方案，优于明文存储

```
# 使用环境变量的示例
export PMAC_DB_PASSWORD="secret_value"
export PMAC_SSH_PASSPHRASE="ssh_secret"
```

保护配置文件

```
# 所有者: www-data (Apache 用户)
chown root:www-data /srv/www/pmacontrol/configuration/*.php

# 权限: 组可读, 其他用户无权限
chmod 640 /srv/www/pmacontrol/configuration/*.php

# 凭据文件应仅允许 www-data 读取
chmod 600 /srv/www/pmacontrol/configuration/db.config.ini.php
```

第 5 层: ACL (访问控制列表)

审查 acl.config.ini

PmaControl 拥有基于角色的 ACL 系统。`configuration/acl.config.ini` 文件定义了哪个角色可以访问哪个控制器。

```
; configuration/acl.config.ini
[admin]
* = allow

[dba]
Slave = allow
Server = allow
Dashboard = allow
Backup = deny
Config = deny

[readonly]
Slave = allow
Server = allow(show)
Dashboard = allow
* = deny
```

基本规则:

- `Config` 控 `Backup` `Install` `Api` 应仅对管理员开放
- 创建只读角色：用于需要查看但不需要修改的开发人员
- 定期审计：验证新添加的控制器已被 ACL 覆盖

保护关键端点

一些端点特别敏感：

```
[admin]
Install = allow       ; 安装/重装
Config = allow        ; 配置修改
Api = allow           ; 完整 REST API
Backup = allow        ; 备份访问（包含数据）

[dba]
Install = deny        ; 绝不允许非管理员访问
Config = deny
Api = allow(read)     ; 通过 API 只读访问
Backup = deny
```

第 6 层：**CSRF**（跨站请求伪造）

验证令牌存在

每个 PmaControl 表单都必须包含 CSRF 令牌：

```
<form method="POST" action="/slave/start/42/">
    <input type="hidden" name="csrf_token" value="<?= $csrf_token ?>">
    <button type="submit">Start Slave</button>
</form>
```

在服务器端，控制器必须验证令牌：

```
if ($_POST['csrf_token'] !== $_SESSION['csrf_token']) {
    throw new SecurityException('Invalid CSRF token');
}
```

需要优先保护的操作

修改状态的操作是最关键的：

- START/STOP SLAVE
- 跳过错误
- 服务器添加/删除
- 配置变更
- 用户创建/删除

没有 CSRF 保护，攻击者可以通过发送恶意链接，强制已登录的 DBA 停止生产服务器的复制。

第 7 层：文件权限

权限树

```
# 主目录：可读，不可写
chown -R root:www-data /srv/www/pmacontrol/
chmod -R 750 /srv/www/pmacontrol/

# 写入目录：www-data 为所有者
chown -R www-data:www-data /srv/www/pmacontrol/tmp/
chown -R www-data:www-data /srv/www/pmacontrol/data/

# PHP 文件：www-data 只读
find /srv/www/pmacontrol/App/ -name "*.php" -exec chmod 640 {} \;

# 配置：限制性权限
chmod 640 /srv/www/pmacontrol/configuration/*.php
chmod 600 /srv/www/pmacontrol/configuration/db.config.ini.php
```

`www-data` 可读取代码并写入 `tmp/` 和 `data/`。它不能修改源代码或配置。

第 8 层：安全监控

记录 **API** 访问日志

每个 REST API 调用都应记录以下信息：

- 时间戳
- 来源 IP
- 用户（令牌）

- 调用的端点
- 响应码

```
// 在 API 中间件中
$log = sprintf(
    "[%s] %s %s %s -> %d",
    date('Y-m-d H:i:s'),
    $_SERVER['REMOTE_ADDR'],
    $user->name,
    $_SERVER['REQUEST_URI'],
    http_response_code()
);
file_put_contents('/var/log/pmacontrol/api.log', $log . "\n", FILE_APPEND);
```

认证失败的 Telegram 告警

为每次登录失败配置 Telegram 告警:

```
if (!$auth->isValid()) {
    Telegram::send(
        "Auth failure on PmaControl\n" .
        "IP: " . $_SERVER['REMOTE_ADDR'] . "\n" .
        "User: " . $_POST['login'] . "\n" .
        "Time: " . date('Y-m-d H:i:s')
    );
}
```

同一 IP 在 5 分钟内 3 次失败应触发临时封锁。

监控配置文件

使用 `inotifywait` 或类似工具检测未授权的修改:

```
inotifywait -m -r /srv/www/pmacontrol/configuration/ -e modify,create,delete |
while read path action file; do
    echo "[$action] $path$file" >> /var/log/pmacontrol/config_changes.log
    # 发送 Telegram 告警
done
```

第 9 层: 网络

防火墙规则

```
# 仅允许来自内部网络的 HTTP/HTTPS
iptables -A INPUT -p tcp --dport 80  -s 10.0.0.0/8 -j ACCEPT
iptables -A INPUT -p tcp --dport 443 -s 10.0.0.0/8 -j ACCEPT
iptables -A INPUT -p tcp --dport 80  -j DROP
iptables -A INPUT -p tcp --dport 443 -j DROP

# 仅允许 localhost 访问 MySQL
iptables -A INPUT -p tcp --dport 3306 -s 127.0.0.1 -j ACCEPT
iptables -A INPUT -p tcp --dport 3306 -j DROP
```

禁止公开暴露

PmaControl 绝不应该从互联网访问。即使有身份验证，攻击面也太大：

- 存储了被监控服务器的凭据
- 存储了 SSH 密钥
- 界面允许在生产服务器上执行操作

如果需要远程访问，请使用 VPN（WireGuard、OpenVPN）或 SSH 隧道。

第 **10** 层：**SQL** 注入——修复

我们的内部审计在四个控制器中发现了 SQL 注入风险：

控制器	风险	修复措施
`Tag.php`	动态 WHERE 子句构建	预处理语句
`Client.php`	过滤器中的字符串拼接	预处理语句
`Environment.php`	ORDER BY 中的变量插值	列名白名单
`Backup.php`	LIKE 中未转义的参数	预处理语句

修复前（存在漏洞）：

```
// Tag.php — 存在漏洞
$sql = "SELECT * FROM tags WHERE name LIKE '%" . $_GET['search'] . "%'";
$results = $db->query($sql);
```

修复后（安全）：

```
// Tag.php — 安全
$sql = "SELECT * FROM tags WHERE name LIKE ?";
$results = $db->query($sql, ['%' . $_GET['search'] . '%']);
```

对于 ORDER BY 子句，白名单是唯一安全的解决方案：

```
$allowed_columns = ['name', 'created_at', 'id'];
$sort = in_array($_GET['sort'], $allowed_columns) ? $_GET['sort'] : 'name';
$sql = "SELECT * FROM tags ORDER BY " . $sort;
```

加固检查清单

在将 PmaControl 投入生产之前，验证每一项：

- [ ] Apache: 启`Indexes`
- [ ] Apache: 强制 HTTPS
- [ ] Apache: 限制访问为内部网络
- [ ] Apache: 删除默认虚拟主机
- [ ] PHP: 在 Web 虚拟主机上禁用危险函数
- [ ] PHP: `session.cookie_httponly = 1`
- [ ] PHP: `session.cookie_secure = 1`
- [ ] PHP `expose_php = Off`
- [ ] MariaDB: 使用最小权限用户
- [ ] MariaDB `bind-address = 127.0.0.1`
- [ ] 钥`ypted=1`）
- [ ] 配置文件: 权限 640
- [ ] ACL: 限制敏感控制器
- [ ] CSRF: 所有操作表单都有令牌
- [ ] 权`mp/` 和`data/` 是唯一可写目录
- [ ] 监控: API 访问日志
- [ ] 监控: 认证失败告警

- [ ] 网络：防火墙就位
- [ ] 网络：禁止公开暴露
- [ ] SQL：Tag、Client、Environment、Backup 中使用预处理语句

总结

保护 PmaControl 不是奢侈品——而是义务。该工具可以访问您的生产 MariaDB / MySQL 服务器，存储凭据，并且可以通过 SSH 执行命令。

加固是分层进行的：每一层（Apache、PHP、MariaDB、密钥、ACL、CSRF、权限、网络）都增加了一道屏障。如果一层被突破，其他层会减缓攻击者的速度。

好消息是：所有这些措施都是标准的，可以在一个工作日内完成。与数据库基础架构被攻破的风险相比，成本微乎其微。