LGNS 跨链互换合约 · 源码与权限审计材料

面向第三方技术审计交付 · 全部数据为链上实测/可复现 · 出具日期 2026-09-16 · 联系人 Zane

一、交付摘要

合约用途	LGNS 在 A链（Anubis 6714）与 Polygon（137）之间的同额互换托管与撮合（挂单 — 配对 — 交割 — 超时撤回）
合约地址	`0x0EBe8Ee79624cb2314FCDFa306973a044EE9dfF4`（两条链同一地址）
是否可升级	**否**。EIP-1967 实现槽与 admin 槽均为 0，非代理合约，部署后字节码永久固定，任何一方（含所有者）都无法修改合约逻辑
源码公开状态	**两条链均已公开可查**：Polygon 侧经 Sourcify 验证（2026-09-08）；A链 侧经区块浏览器验证通过 `Pass - Verified`（2026-09-16）
代码可复现性	**已逐字节验证**。按公开编译参数重新编译，创建字节码 12,358 字节中 12,315 字节完全一致，差异仅存在于末尾 43 字节的元数据指纹区（详见第四节）
权限现状	**所有者仍保留 5 项管理开关，权限未全部丢弃**。原因见第五节「2. 为什么权限没有全部丢弃」——两条链的合约无法互相观察，中间必须有一个观察者（1 台 7×24 挂云运行的撮合机器人）执行放款，该角色必须可配置、可更换、可紧急停用

二、链路与部署事实

项目	值
合约地址	`0x0EBe8Ee79624cb2314FCDFa306973a044EE9dfF4`
A链 运行时代码	`10,783 字节，keccak 0xb9450eededcc7d9dca…`
Polygon 运行时代码	`10,783 字节，keccak 0x43169c373a920ca5f6…`
A链 部署交易	`0x479b3b3d4341ae8ef6eb21fbc6349d21e9bde070f3d945fcb87ba2d914928d0a`
A链 部署区块	`13,336,894`
部署者 / 当前所有者	`0xe05eF897Bd9FAC2176e5b1baa5744261E34a4e83`（普通外部账户 EOA，非多签）
服务费收款地址	`0xd09D09FBf248b78B1412baf052bE163C04E623E6`（EOA）
本链 LGNS 代币	A链 `0x4D1D808a081FdAc440703b3765FC61f8028C06B8`
创建交易输入	`12,454 字节 = 创建字节码 12,358 + 构造参数 96（token / owner / feeTo 三个地址）`
挂单笔数	A链 24 笔（其中已成交 11、已退回 7、在挂 6）；Polygon 11 笔（全部已成交）

注：两侧运行时代码哈希不一致是因为合约用 `immutable` 存储本链 LGNS 地址，两条链代币地址不同，属正常现象；创建字节码本身一致。

三、编译参数（用于验证复现）

编译器版本	v0.8.36+commit.8a079791
优化器	enabled = false（runs 200 仅为记录值）
EVM 目标版本	osaka
元数据字节码哈希	ipfs
源文件	contracts/LgnsSwap.sol（单文件，无外部依赖，无 import）
构造参数	token 0x4D1D80…C06B8 / owner 0xe05eF897…4e83 / feeTo 0xd09D09FB…23E6

四、代码可复现性证据（核心）

用上述参数重新编译源码，与 A链 链上创建字节码逐字节比对，结果：

```
链上创建字节码 : 12,358 字节
本地复现       : 12,358 字节
不同字节数     : 32 个
首个差异位置   : 第 12,315 字节（即末尾 43 字节的 CBOR 元数据区）
→ 业务逻辑主体 100% 一致，元数据指纹区外的每一字节均可复现
```

元数据指纹差异的原因：metadata 哈希由「源码文本」的 keccak 决定，注释与空白字符的变化会改变该指纹， 却不改变任何字节码。部署时所用源码文本的最后细微注释差异已无法复原（项目未纳入版本控制）， 因此本材料随附一份可复现出上述字节码的源码副本，供审计方独立复核。

复现命令

```
# 依赖: python + solcx（solc 0.8.36）
import solcx
out = solcx.compile_files(["contracts/LgnsSwap.sol"], output_values=["bin"],
                          solc_version="0.8.36", evm_version="osaka")
binhex = out[[k for k in out if k.endswith(":LgnsSwap")][0]]["bin"]
# 与链上创建字节码（去掉末尾 96 字节构造参数）逐字节比较
```

五、权限清单（审计重点）

1. 可调用的管理函数

所有者地址为普通外部账户（EOA），可调用以下函数；合约**未提供 renounceOwnership**，即不提供"一键放弃所有权"。

函数	作用	风险与边界
`setPaused(bool)`	暂停 / 恢复 `deposit` 与 `confirmMatch`	无法阻止 `refund`，用户始终可取回本金（自救通道）
`setFeeTo(address)`	更换服务费收款地址	只影响后续成交，不改动已成交订单
`setProver(address,bool)`	新增 / 移除「观察者（证明人）」	观察者是唯一能执行放款（`confirmMatch`）的角色。放款**必须附带一笔对方链存款交易哈希作为凭据**，且必须满足：对方链存款额与本单**严格等额**、该哈希此前**未被使用过**、在**本单 12 小时窗口之内**。该凭据由链上永久记录。调用此函数可更换观察者地址（用于机器人迁移 / 私钥轮换 / 紧急停用）
`setMinAmount / setMaxAmount`	调整单笔挂单上下限	仅影响新挂单，不动已有订单
（无 `renounceOwnership`）	—	不提供放弃所有权函数（原因见下文第 2 点）

2. 为什么权限没有全部丢弃（设计原因）

因为这两条链之间，必须有一个观察者。

A链 与 Polygon 是两条独立的链，链上的合约**互相看不到对方**： A链 的合约不知道 Polygon 上有没有人真的存了币，Polygon 的合约也不知道 A链 上有没有人存币。

所以这两条链中间**必须有一个观察者**：由机器人盯着另一条链，确认对方链上 真的有等额存款进来、金额与交易哈希都核对无误之后，才在本链上放款（`confirmMatch`）。 没有这个观察者，跨链互换在技术上无法完成交割。

这个观察者不是人，是一台挂在云上的机器人。 它运行在一台 7×24 常驻的云服务器上（Ubuntu 系统，以 systemd 常驻服务方式运行）： 一边监听两条链的存款事件，一边在核对无误后提交放款指令。它的运行需要私钥（保存在该服务器上）， 因此"换机器、轮换私钥、故障时先停一下"这类运维动作，对应到合约里就是本节表格中的 `setProver`（更换观察者地址）与 `setPaused`（临时停用）。

这个观察者角色**必须是可配置的**：机器人要升级、要搬到新服务器、 观察者私钥要轮换、出现异常要能临时停一下——这些都依赖合约里的管理入口， 也就是 `setProver` 与 `setPaused` 这两个函数。

结论：权限没有全部丢弃，不是不肯丢，而是**跨链本身就必须有一个观察者在中间看着。** 如果连观察者的管理入口也丢弃（例如把所有者设为 `0x0`），一旦机器人故障或需要更换地址， 通道将永久失去维护能力，用户挂单只能等到期自行撤回。

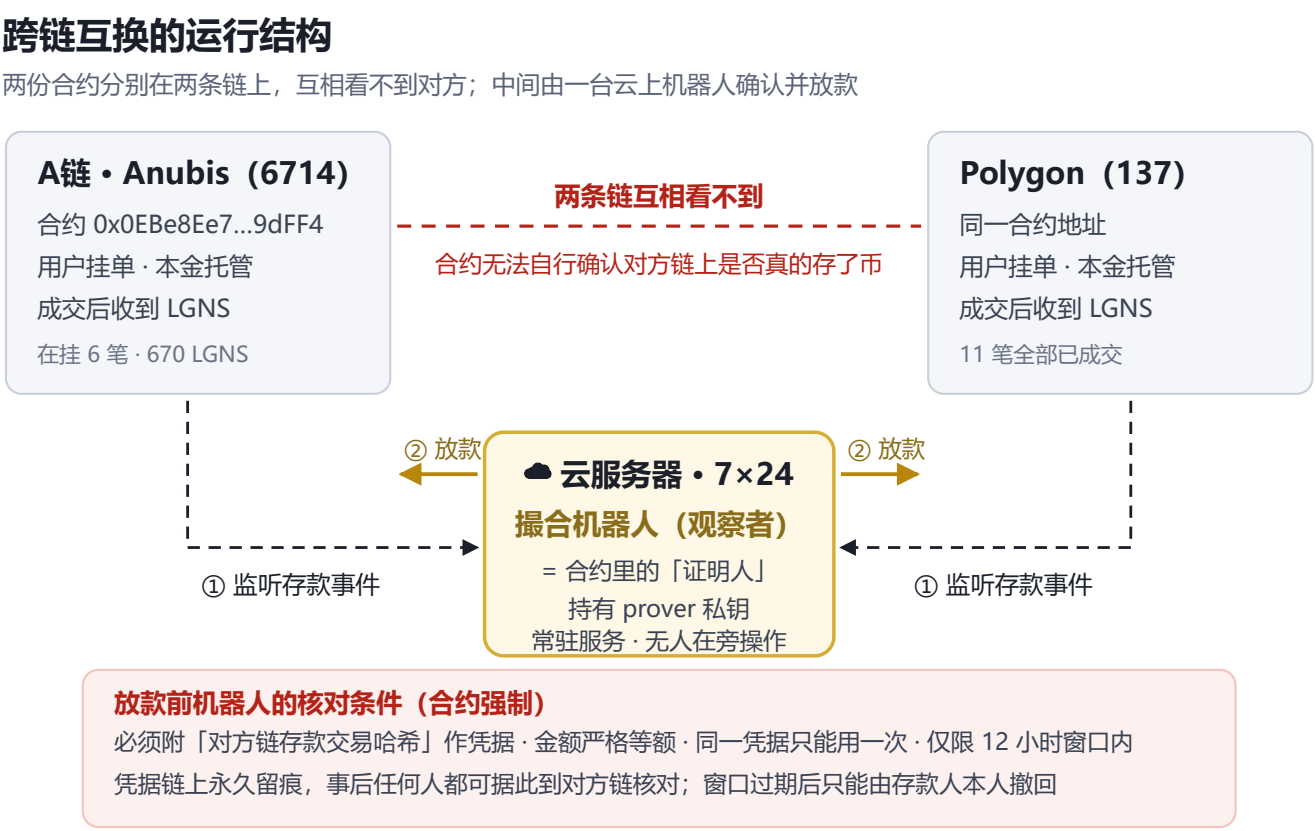

图 1 · 跨链互换的运行结构：两条链互相看不到，中间由一台云上机器人（观察者 / 证明人）确认并放款

3. 权限已被压缩到什么程度

- 全部权限**穷尽为 5 项**（上表），合约内**不存在任何可以直接转走用户资金的函数**
- 用户保护类函数（`refund` 超时撤回）**不受暂停影响**，任何时候都能取回本金
- 合约**不可升级**：不存在"改代码换逻辑"的后门，管理权限只能改参数，不能改规则
- 源码两链公开，任何权限调用都在链上留有记录、可被任何人追溯

4. 观察者（证明人）作恶的边界与约束（如实说明）

观察者（即那台挂云运行的机器人）持有放款权，这个事实必须在审计中讲清楚：**它不能凭空把钱转走，但它的确能在窗口内把某笔挂单的托管本金放给一个由它指定的地址。**约束与后果如下：

- **必须有凭据才能放款**：每次 `confirmMatch` 都必须提交一笔"对方链存款交易哈希"， 链上会把该哈希永久写入订单记录。任何人都可以拿这个哈希到对方链核对是否真有那笔存款。
- **金额不能超发**：放款额固定为「本单本金 − 2% 服务费」，且只能放这一张单的钱，不能多拿、不能动别人的单。
- **同一凭据只能用一次**：对方链存款哈希一经使用即被标记，重复使用会被合约拒绝。
- **只在 12 小时窗口内有效**：窗口一过，合约不再允许放款，只能由存款人本人调用 `refund` 取回。
- **不能动用无主资金**：合约不沉淀资金池，服务费成交即实时转出；托管资金一一对应到明确的挂单。
- **损失有上限、且事后可举证**：最坏情况是当前处于 12 小时窗口内的挂单本金被放错地址； 但链上会留下那笔伪造/错配的凭据哈希，事后任何人都能据此核对并追溯到具体操作—— **也就是说，这是"可追责、有上限"，而不是"无痕迹、无限制"。**

一句话：跨链互换的观察者是一台**挂云运行的撮合机器人**，它同时受**凭据、时间窗口、链上留痕**三重约束， 它做不到"随手把本金送人"；但在窗口之内，它确实是这套机制里的信任集中点。

小结（如实呈现）：合约本身**没有**任何允许所有者直接动用用户资金的函数； 但观察者（可由所有者更换）在 12 小时窗口内持有放款权。 权限**未全部丢弃**的根本原因是：**两条链上的合约互相看不到对方，中间必须有一个观察者（机器人）** 负责确认对方链的存款并放款，而这个观察者角色必须可配置、可更换、可临时停用 （存在原因见本节第 2 点，约束见第 4 点）。

六、资金流向与用户保护

环节	规则
挂单 deposit	仅收取本金（本链 LGNS），单笔 10 – 5,000 LGNS；挂单阶段零费用
撮合 confirmMatch	仅证明人可调用；要求「对方链存款额 == 本单额」严格等额、对方链交易哈希未被使用过、在 12 小时窗口内、对手方地址非零且非本人
交割	本金减去 2% 服务费付给对手方；服务费实时转出至 feeTo，不在合约内沉淀
超时撤回 refund	挂单满 12 小时未成交后，**仅存款人本人**可调用，取回全部本金（从未收取费用）；该函数不受暂停影响
合约内资金	当前 A链 合约内锁有 670 LGNS（对应 6 笔在挂订单），Polygon 侧为 0

七、达标情况与后续方案

要求	现状
代码不可升级（无后门）	**达标**：非代理，EIP-1967 槽全 0
代码公开、可审计、可复现	**达标**：两链源码已公开；字节码主体逐字节可复现
无中心化权限（权限丢弃）	**未全部丢弃**，原因是技术性的：两个链上合约互相看不到对方，中间必须由机器人（观察者）确认对方链存款并放款，该角色必须可配置、可更换、可临时停用（详见第五节第 2 点）。已采取的收敛：权限穷尽为 5 项、无任何直接动用用户资金的函数、撤单/退款不受暂停影响、不可升级、源码两链公开可追溯
用户资金不可被挪用	无直接转账后门；管理权限可经 `setProver` 取得放款能力，这是"链与链之间必须有观察者"这一机制的固有部分

后续可选项

- **换形态（可选）**：若后续要把"观察者"这一环节交给第三方跨链消息协议承载， 则可部署一版零管理员合约（所有者设为 `0x0`、参数在构造时写死、无暂停、无证明人）。 届时现有合约待在挂订单结清后退役。

说明：只要"两条链之间必须有观察者"这一前提不变，本合约的权限结构就是该机制下最小的形态。

八、审计方自查方式

- A链 浏览器：`https://browser.anubispace.org/address/0x0EBe8Ee79624cb2314FCDFa306973a044EE9dfF4`（源码 / ABI / 交易记录）
- Polygon 侧源码：Sourcify 验证记录（chainId 137，同一地址）
- RPC 直连：A链 `https://rpc2.anubispace.org`（chainId 6714，PoA，需 POA 中间件）；Polygon `https://polygon-bor-rpc.publicnode.com`（chainId 137）
- 关键只读调用：`owner()` / `feeTo()` / `paused()` / `minAmount()` / `maxAmount()` / `orderCount()` / `orders(i)` / `lockedBalance()`

> **安全提示：**本材料不含任何私钥、助记词、服务器口令或内部凭据。 撮合与播报程序运行于独立服务器，其目录内保存有证明人私钥；如需在本方环境复现运行， 请勿使用生产密钥，建议另建测试环境并生成独立测试密钥。

本材料所有结论均基于 2026-09-16 于 A链(6714) 与 Polygon(137) 的链上实测数据，可自行复核。 另附：可复现出上述字节码的源码副本 `LgnsSwap.sol`（8,220 字符）。