packetpaper.txt: An Accountable Network for Machines

PacketPaper LLC · White Paper · Version 1.1 · November 2026

Abstract

A growing share of internet traffic is sent by machines: search crawlers, AI agents, price monitors, uptime checkers, mail servers and bots of every kind. Websites cannot tell the good ones from the bad ones. Abusive traffic hides in cloud address ranges and on home internet connections that nobody answers for, so sites respond by blocking whole networks, and honest builders are blocked along with the abusers. This paper proposes packetpaper.txt, a simple, open way for networks to publish who is accountable for the automated traffic leaving their addresses, and for websites to publish which kinds of automated traffic they will accept. It also describes the PacketPaper Lab network, a new network in Chicago built to follow the proposal from its first day. Nothing described here is in production yet. Everything not yet running is labeled planned.

1. The problem

When a request reaches a website, the site learns three things it can trust: the IP address it came from, the time, and what was asked for. Everything else, including the user agent, is self-reported and easy to fake.

From the IP address alone a site can usually learn which network the address belongs to. It cannot learn who inside that network sent the traffic, what the traffic is for, or how to make it stop. WHOIS and RDAP records name the network operator, not its customers. Abuse contacts, when they exist, are slow, generic and often unanswered.

So site operators do the only thing that works at scale: they block. They block entire cloud providers, entire hosting companies, entire countries. They put challenge pages in front of everything. This has three costs:

- **Honest automation gets blocked.** A developer running a legitimate crawler, a monitoring service or an AI agent on a cloud server is treated the same as a credential-stuffing botnet on the same provider.
- **Abusers move.** Blocking a range does not stop bad actors. They rent addresses elsewhere, increasingly on home internet connections, and the cycle repeats.
- **Nobody is accountable.** There is no common, machine-readable way for a site to ask a network, "who is responsible for this traffic, and how do I report it?"

robots.txt, standardized in RFC 9309, tells well-behaved crawlers which paths to avoid. It says nothing about who is crawling, whether they can be trusted, or what happens when they misbehave. It depends entirely on the crawler choosing to obey.

2. Related work

Several efforts address parts of this problem. packetpaper.txt is designed to sit alongside them, not replace them.

Effort	What it does	What it leaves open
robots.txt (RFC 9309)	Site lists paths crawlers should avoid	Who is crawling; trust; enforcement
security.txt (RFC 9116)	Site publishes a security contact	Says nothing about automated traffic
Signed bot requests (HTTP Message Signatures, RFC 9421, and industry "Web Bot Auth" work)	A bot proves its identity cryptographically per request	Only for bots that adopt it; does not cover the network the traffic came from
AI usage preferences (IETF work in progress)	Site states whether content may be used for AI training, search and so on	Who is accountable for the traffic that collects it
WHOIS / RDAP	Names the operator of an address block	Customers and purposes inside the block

The gap is accountability at the network level: a standard way for the operator of an address block to say, for any address that sends automated traffic, who is responsible, what the traffic is for and how to get it stopped.

3. The proposal: packetpaper.txt

packetpaper.txt has two halves. Either can be used on its own, and they are most useful together.

3.1 Network side: the accountability record

A network that sends automated traffic publishes an accountability record for each address range that carries it. The record is plain text, served at a well-known location on the network's own domain and discoverable from the address itself.

```
# https://packetpaper.com/.well-known/packetpaper.txt
Version: 1
Network: PacketPaper LLC
Contact: noc@packetpaper.com
Abuse: https://packetpaper.com/report        # planned page
Policy: https://packetpaper.com/aup

Range: <IPv6 prefix>                         # one customer's own IPv6 range
Purpose: ai-agent
Customer: c-7f3a91        # stable pseudonymous customer ID
Verified: yes             # customer identity checked by the network
Robots: obey
Rate-Limit: 1/s per host
Report: https://packetpaper.com/report?c=c-7f3a91

Range: <IPv4 prefix>                         # shared transition gateway
Purpose: ai-agent
Customer: shared
Attribution: address, port and time

Range: <IPv6 prefix>
Purpose: crawler
Customer: packetpaper     # our own crawler
Verified: yes
Robots: obey
Info: https://packetpaper.com/bots/packetpaper
```

On a shared IPv4 address, many customers share the address, each on its own block of ports. Each connection is traced to one customer by address, port and time, so a report about a shared address needs all three.

A site that receives a request from an address can find the record without prior knowledge of the network, through DNS. The network publishes a TXT record under the reverse DNS name of the address or its covering prefix:

```
_packetpaper.<reverse name of the address>  TXT
  "v=pp1; url=https://packetpaper.com/.well-known/packetpaper.txt"
```

The record answers four questions a site actually has:

1. **Who is accountable?** The network, and a stable customer ID behind it. The customer's real identity stays private between the customer and the network, but the network vouches that it knows who they are.
2. **What is the traffic for?** A short list of purposes: crawler, ai-agent, monitor, mail, research-scan, proxy.
3. **Does it follow the rules?** Whether it obeys robots.txt, its rate limits, and a link to the network's acceptable use policy.
4. **How do I stop it?** A report link that goes straight to the network's abuse desk with the customer already named. The desk can suspend that one customer in a single step.

Records are signed, so a site can be sure the record came from the network that holds the addresses. The signing key is published at the network's domain, and the address ranges in the record are checked against routing data (RPKI route origins) so a network cannot claim addresses it does not hold.

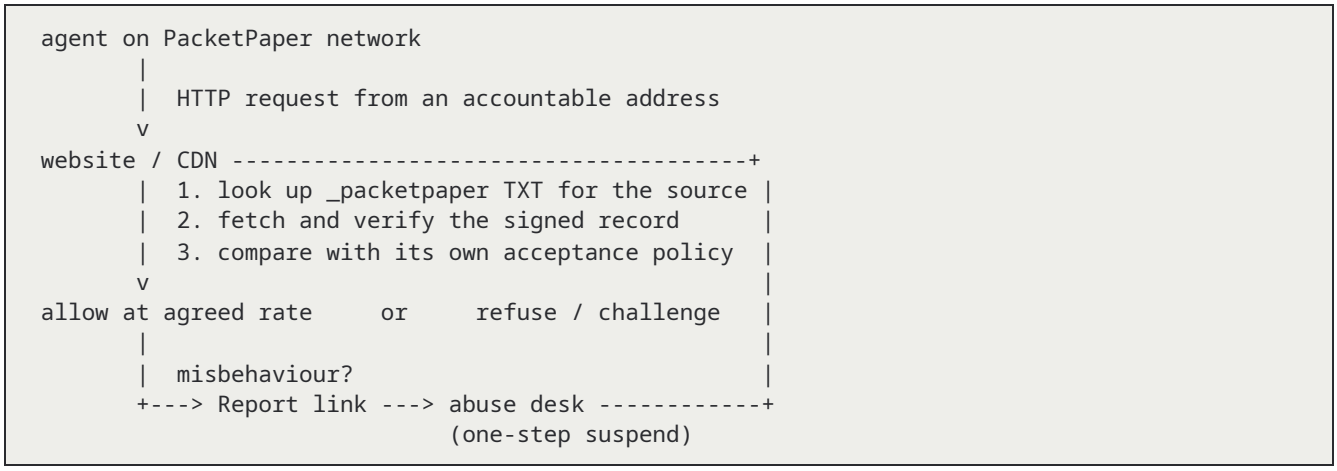
3.2 Site side: the acceptance policy

A website publishes its own packetpaper.txt stating what automated traffic it accepts and on what terms:

```
# https://example.com/.well-known/packetpaper.txt
Version: 1
Accept: crawler       when Verified=yes and Robots=obey
Accept: monitor       when Verified=yes
Accept: ai-agent      when Verified=yes   rate=5/s
Refuse: ai-agent      purpose=training
Refuse: unaccountable                     # no record found
Contact: webmaster@example.com
```

This lets a site say things robots.txt cannot: give accountable traffic a faster lane, refuse traffic that cannot say who sent it, and treat different purposes differently. Sites can apply the policy at their firewall, CDN or application, using the network-side records as input.

3.3 How it works end to end



3.4 Design principles

- **Simple.** Plain text, one well-known path, one DNS record. A small site can adopt it in minutes.
- **Privacy-preserving.** Customers are identified by pseudonymous IDs. The network knows who they are; the public does not.
- **Verifiable.** Signed records, checked against routing data.
- **Incremental.** Useful to the first site and the first network that adopt it. No central registry is required.
- **Complementary.** Works with robots.txt, signed bot requests and AI usage preferences rather than replacing them.

4. The PacketPaper Lab network

PacketPaper is building a network in Chicago that follows packetpaper.txt from its first day. It exists to prove the proposal works and to give honest builders a place to send automated traffic from that sites can trust.

4.1 Design

- **Every service gets IPv6 and IPv4.** Every customer and every service receives native IPv6 and an IPv4 address. IPv6 is the foundation of the network; IPv4 is provided alongside it for reachability.
- **Every address has a record.** No address on the network sends automated traffic without a published accountability record.
- **Segments by purpose.** Agents, our own crawlers, proxies, mail and sensors each run in their own address ranges with their own policy, so a problem in one cannot spill into another and so the purpose field in the record is always accurate. Monitoring comes later, on addresses of its own.
- **Routing hygiene.** Prefixes announced with RPKI route origins, and source address validation on every customer connection, following the MANRS actions for network operators.
- **Attribution.** Every connection leaving the network can be traced to one customer and one purpose. On shared IPv4, that takes the address, port and time.

4.2 Planned services

All of the following are planned. None is running today.

Accountable outbound traffic for AI agents. Customers run their agents on the network. Every one is identity-checked, publishes an accountability record, identifies itself in its user agent, obeys robots.txt and links to an info page explaining what it does and how to opt out. Our own crawlers follow the same rules. Crawling for customers comes later, on its own addresses.

Monitoring and uptime checks (later), on addresses of their own. Addresses for monitoring services, published as such, so sites can allow them without guesswork.

Mail. Sending addresses for the MinakiMail platform, on a range used for nothing else, with a published record, SPF, DKIM and DMARC enforced, and sending limits. Dedicated addresses for steady high-volume senders.

Accountable proxies. Proxy addresses for legitimate uses such as testing what a site looks like from Chicago, with every customer verified and every address published.

Scan and attack data. Sensors that listen on unused addresses and record what connects to them. On our own addresses they never answer. The few decoys that do answer run on separate rented addresses and can't send traffic anywhere else. None of them ever scans. The result is a feed showing which traffic on the internet refuses to identify itself, the other half of the accountability picture.

IPv6-first hosting. Servers and containers for small businesses and developers, on native IPv6, with IPv4 visitors reaching them through our shared front ends.

packetpaper.txt tooling. Open-source tools: a validator for network and site files, a lookup command, and plug-ins for common web servers and CDNs so sites can apply acceptance policies.

5. Abuse handling

An accountable network has to answer for its traffic. These are commitments for every service from the day it opens, not aspirations:

- **Reachable people.** abuse@ and noc@ are read and answered. Contacts will be published in WHOIS and PeeringDB once our registration is approved, and in security.txt and every packetpaper.txt record.
- **Verified customers.** Every customer of outbound, proxy and mail services is identity-checked before activation.
- **Blocked destinations.** Automated traffic to banking, login and checkout pages and to government systems is blocked by default.
- **Limits.** Per-customer rate limits on connections, requests and mail.
- **One-step suspension.** A report through a record's link goes straight to our abuse desk with that customer already named. The desk can suspend that one customer in a single step, within a minute, without affecting anyone else.
- **No evasion services.** The network does not sell block evasion, ban evasion, bulk unsolicited mail or a way to hide who sent traffic for abuse. Traffic that hides who sent it is exactly what this network exists to end.

6. Measurement

We will publish on a public status page:

- the number of accountability records published and the share of our traffic they cover (target: all of it);
- abuse reports received, median time to respond and the share resolved by suspension;
- the number of sites known to apply packetpaper.txt policies;
- from the sensors, the share of unsolicited traffic that carries no accountability record.

7. Relationship to MinakiLabs and TunnelNet

PacketPaper LLC is a subsidiary of MinakiLabs LLC. MinakiLabs also operates TunnelNet, which gives customers a public IP address in the city they choose, over encrypted tunnels. TunnelNet gives people an address; the PacketPaper Lab gives their machines an address the rest of the internet can trust. The two are separate networks with separate purposes, addresses and contacts. We state the relationship openly because sites, registries and peers should know who operates a network.

8. Roadmap

1. Publish this proposal and a draft specification for comment.
2. Bring up the PacketPaper Lab network in Chicago with RPKI and published contacts.

3. Publish our own network-side records and release the open-source validator and lookup tools.
4. Bring up sensors and publish the first report on unaccountable traffic.
5. Launch accountable outbound traffic for agents, with verification and one-step suspension.
6. Release web server and CDN plug-ins so sites can apply acceptance policies.
7. Submit the specification to a standards body once it has been used in practice.

9. Conclusion

The internet does not need more blocking. It needs a way for machines to say who sent them and for the people responsible to be reachable. packetpaper.txt is a small, open proposal to do that, and the PacketPaper Lab is a network built to prove it: every address accounted for, every purpose stated, every report answered.

Contacts

Network operations: noc@packetpaper.com · Abuse: abuse@packetpaper.com · Peering: peering@packetpaper.com · Comments on the proposal: noc@packetpaper.com

References

RFC 9309, Robots Exclusion Protocol · RFC 9116, security.txt · RFC 9421, HTTP Message Signatures · RFC 8615, Well-Known URIs · RFC 6811, BGP Prefix Origin Validation (RPKI) · RFC 9082/9083, RDAP · MANRS Actions for Network Operators · IETF AI Preferences (aipref) working group drafts