GUIDE

The Internet Runs on Names: Research Explainer for Builders

The Internet's foundational protocols, like IP, were designed around numerical addresses. But for decades, engineers have been building layers on top that abstract away those numbers. This paper, "The Internet Runs on Names," makes a compelling case that this abstraction is now complete: **DNS names have become the true operational primitive of the Internet, eclipsing IP addresses in importance for how services are identified, reached, balanced, and trusted.**

This isn't just an academic observation; it has profound implications for how we design, build, secure, and operate networked systems today.

The Problem: The IP-Centric Illusion

For a long time, the mental model of the Internet was IP-centric. You wanted to reach a server? You needed its IP address. DNS was seen as a mere directory service, translating human-readable names into machine-readable IPs. This view, while historically accurate for the early Internet, no longer reflects reality.

Modern internet services are rarely accessed directly via IP. Consider:

- **Virtualization and Cloud:** Servers are ephemeral, their IPs change constantly.
- **CDNs and Edge Computing:** A single name can resolve to hundreds or thousands of different IPs globally, chosen dynamically for performance or availability.
- **Load Balancers:** A single IP often fronts many backend servers, making the IP an identifier for a group rather than a single service instance.
- **Security:** TLS certificates bind trust to names, not IPs.

The problem the paper identifies is that continuing to think of the Internet as primarily IP-addressable leads to a disconnect between our mental models and the actual operational reality. This can lead to suboptimal designs, security vulnerabilities, and difficulties in troubleshooting.

The Core Idea: Names as the Internet's Operational Primitive

The paper's core idea is that **DNS names have become the primary basis for service identity, reachability, load balancing, and trust on the Internet.** IP addresses, while still necessary at the packet level, are now merely transient, dynamic, and often opaque pointers behind the name.

Let's break down these four pillars:

Service Identity: Who Are You?

- **Traditional View:** An IP address uniquely identified a server or network interface.
- **Name-Centric View:** A DNS name (`api.example.com`) identifies a service or logical entity, not a specific machine. This service might be a single container, a cluster of servers, a function-as-a-service endpoint, or a global CDN. The underlying IPs can change without the service's identity changing.
- **Practical Impact:** When you configure a client to connect to `my-service.internal`, you're identifying the service by its name, not by the IP address that `my-service.internal` resolves to at any given moment.

Reachability: How Do I Get There?

- **Traditional View:** Routing tables direct packets to specific IP addresses.
- **Name-Centric View:** DNS resolution is the first and most critical step in establishing reachability. The DNS system, through various record types (A, AAAA, CNAME, SRV), directs clients to the correct IP(s) for a given name, potentially factoring in client location, server load, and policy.
- **Practical Impact:** Geo-distributed services use DNS to direct users to the nearest data center. Cloud auto-scaling groups update DNS records to reflect new instances. Service meshes use DNS to discover and route to internal services.

Load Balancing: Where Should I Go?

- **Traditional View:** Dedicated hardware load balancers distribute traffic among a set of known IPs.

- **Name-Centric View:** DNS itself acts as a massive, distributed load balancer. Techniques like DNS Round Robin, weighted DNS records, and advanced GSLB (Global Server Load Balancing) services distribute client requests across multiple IP addresses associated with a single name. This happens before the client even attempts to connect.
- **Practical Impact:** CDNs use DNS to direct traffic to optimal edge nodes. Cloud providers use DNS to balance requests across regions or availability zones. This "Layer 4.5" load balancing is often the first point of traffic distribution.

Trust: Can I Believe You?

- **Traditional View:** Trust was often implicitly tied to network location or IP ranges.
- **Name-Centric View:** Trust on the modern Internet is overwhelmingly anchored to names via TLS certificates. When you visit `<https://www.bank.com>`, your browser verifies that the certificate presented by the server is valid for `www.bank.com` (the name), not for the specific IP address it connected to. DNSSEC adds another layer of trust by cryptographically signing DNS records themselves, ensuring the integrity of the name-to-IP mapping.
- **Practical Impact:** Certificate management, automated certificate issuance (e.g., Let's Encrypt), and DNSSEC deployment are critical security concerns that revolve entirely around names.

How it Differs from the Traditional IP-Centric View

The paper fundamentally shifts the perspective from "DNS is a helper for IP" to "IP is a helper for DNS."

Feature	Traditional IP-Centric View	Name-Centric View (Paper's Argument)
Primary Identifier	IP Address	DNS Name
Service Identity	A server is its IP address.	A service is its DNS name; IPs are dynamic endpoints.
Reachability	Based on IP routing tables.	Initiated by DNS resolution, which guides routing.
Load Balancing	Dedicated hardware/ software at a specific IP.	Distributed via DNS (GSLB, Round Robin) before connection.
Trust Anchor	Network location, implicit trust.	TLS certificates bound to names; DNSSEC validates name-to-IP mappings.
IP Address Role	Stable, primary identifier.	Transient, dynamic, often opaque underlay for a name.
DNS Role	Address lookup service.	Control plane for identity, routing, balancing, and trust.

This isn't to say IP addresses are obsolete; they are still the fundamental addressing scheme for packets. However, the operational model for applications and services has moved up the stack, making names the primary interface for interaction.

Practical Implications for Builders

This conceptual shift has concrete consequences for how engineers should approach system design and operations:

For Application Developers

- **Always use names:** Hardcoding IP addresses is almost always a bad idea, even for internal services. Rely on DNS or service discovery mechanisms.
- **Understand DNS resolution paths:** Be aware that a name might resolve differently based on client location, network, or time of day. This is a feature, not a bug.

- **Embrace service meshes:** Tools like Istio, Linkerd, or Consul abstract away network details, relying heavily on names for service discovery, routing, and policy enforcement.

For Network Engineers

- **DNS is paramount:** DNS infrastructure (resolvers, authoritative servers) is now a critical control plane, not just a utility. Its reliability, performance, and security are non-negotiable.
- **GSLB and traffic steering:** Leverage advanced DNS features for global traffic management, disaster recovery, and blue/green deployments.
- **DNSSEC is crucial:** For critical services, DNSSEC provides cryptographic assurance of DNS record integrity, preventing DNS spoofing attacks.

For Security Engineers

- **Certificate management is key:** Ensure robust processes for issuing, renewing, and revoking TLS certificates, as they are the primary anchor of trust for names.
- **DANE (DNS-based Authentication of Named Entities):** Explore DANE as an alternative or complement to traditional CAs for binding trust to names via DNSSEC.
- **DNS as an attack vector:** Understand how DNS can be exploited (e.g., cache poisoning, DDoS against authoritative servers) and implement defenses.

For DevOps and SRE Teams

- **Automate DNS updates:** Integrate DNS record management into your CI/CD pipelines and auto-scaling logic.
- **Monitor DNS:** Treat DNS resolution latency and availability as critical metrics for service health.
- **Troubleshooting:** When diagnosing connectivity issues, always start with DNS resolution. Is the name resolving to the expected IP? Is it resolving at all?

Limitations and Open Questions

The paper primarily presents a conceptual framework and observational evidence. It doesn't propose new protocols or technologies but rather reinterprets the role of existing ones.

- **Quantification:** While the paper makes a strong qualitative argument, it doesn't provide extensive quantitative data on how much traffic or how many services rely solely on names versus direct IP access. Such data could further solidify the argument.
- **Edge Cases:** There are still scenarios where direct IP access is necessary (e.g., bootstrapping, specific network management tasks, legacy systems). The paper acknowledges these but doesn't delve deeply into the implications of this hybrid reality.
- **Future of IP:** What does this mean for IPv6 adoption? While IPv6 offers more addresses, the paper's thesis suggests that the address itself is less important than the name, potentially dampening the urgency for IPv6 solely based on address scarcity.
- **Decentralized Naming:** The paper focuses on the existing DNS hierarchy. How might emerging decentralized naming systems (e.g., blockchain-based names) fit into or challenge this name-centric view?

Should Builders Care?

Absolutely, yes.

This paper articulates a reality that many experienced engineers intuitively understand but might not explicitly frame. By recognizing DNS names as the primary operational primitive:

1. **You build more resilient systems:** Designing services around names allows for dynamic IP changes, seamless failovers, and global traffic management without client-side configuration changes.
2. **You build more secure systems:** Understanding that trust is anchored to names (via TLS) and that DNS itself can be secured (via DNSSEC) is fundamental to modern security posture.
3. **You troubleshoot more effectively:** When connectivity fails, the first question should always be "Is the name resolving correctly?" before diving into IP-level routing or firewall rules.

4. **You leverage cloud-native patterns better:** Cloud environments thrive on ephemeral resources and abstraction. A name-centric mindset aligns perfectly with service discovery, load balancing, and auto-scaling in the cloud.

Ignoring this shift means operating with an outdated mental model, potentially leading to brittle systems, security gaps, and inefficient operations. Embrace the name-centric Internet; your systems will thank you.

References

- The Internet Runs on Names (arXiv:2605.15646) - https://arxiv.org/abs/2605.15646

Transparency Note: This explainer was generated based on the provided abstract and common understanding of modern internet architecture, as the full paper content was not accessible for direct analysis. The claims and implications are derived from the core thesis presented in the abstract.