International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

DNS-Based Service Discovery Tool for Offline Collaboration

Ms. N. Kameshwari(Assistant Professor)
Department of Computer Science and Engineering
Bharath Institute of Higher Education and Research
Chennai, India
kameshwari.cse@bharathuniv.ac.in

Hari Prasanth.S *Department of Computer Science and Engineering*
Bharath Institute of Higher Education and Research
Chennai, India
Nammahari@proton.me

Gannamani. Teja Sai Eswar
Department of Computer Science and Engineering
Bharath Institute of Higher Education and Research
Chennai, India
Tejasaieshwar17@gmail.com

Ganjanaboina. Nagendra Babu
Department of Computer Science and Engineering
Bharath Institute of Higher Education and Research
Chennai, India
nagendrababuganjanaboina@gmail.com

Gundlapali. Sai Kiran
Department of Computer Science and Engineering
Bharath Institute of Higher Education and Research
Chennai, India
saikirangundlapalli@gmail.com

***Abstract*—In small-scale development environments, such as startup teams or university labs, collaborators often rely on locally hosted services like shared code repositories, internal dashboards, or testing servers to streamline workflows. However, accessing these services reliably proves challenging due to dynamic IP addresses that shift with DHCP leases, router reboots, or users switching networks, compounded by unpredictable local conditions like Wi-Fi interference or firewall quirks. This paper introduces OfflineDNS, a lightweight, open-source command-line utility crafted to eliminate these pain points. OfflineDNS assigns persistent, human-readable hostnames (e.g., "dev-server.local" or "alice-git.local") to local services, enabling seamless automatic discovery across the intranet without any reliance on central DNS servers or external infrastructure.Unlike traditional mDNS tools like Avahi or Bonjour, which can falter in unstable networks or require daemon overhead, OfflineDNS operates entirely offline and peer-to-peer. It leverages a custom UDP-based protocol for heartbeat announcements, ensuring sub-second discovery even on lossy links, with zero internet dependency—perfect for air-gapped environments or remote fieldwork.Beyond discovery, OfflineDNS extends utility with built-in secure file sharing via end-to-end encrypted tunnels (using Noise Protocol) and ephemeral messaging channels for quick team chats or log sharing. These features foster efficient, intranet-based collaboration, reducing context-switching and boosting productivity in bandwidth-constrained setups.**

Early prototypes, tested on 10-node LANs with Dockerized services, show 99.5% uptime for hostname resolution and under 50ms latency for peer discovery. By democratizing local networking, OfflineDNS empowers small teams to build robust, self-contained ecosystems.

Keywords— Local DNS, Service Discovery, Intranet Systems, Offline Communication, Peer-to-Peer Sharing.

I. INTRODUCTION

Modern development teams often work within local networks where services are hosted on multiple machines. One of the recurring challenges in such environments is the frequent change in IP addresses, which disrupts accessibility and slows down collaboration. Existing DNS solutions either rely on internet connectivity or demand manual configuration, both of which are impractical for small teams seeking flexibility and independence.OfflineDNS is designed to address these limitations by providing a self-contained system that enables seamless service access. It simplifies hostname resolution within local networks and integrates additional features that support collaboration without external dependencies.Modern development teams frequently operate within local networks, where various services are distributed across multiple machines. A persistent headache in these setups is the constant fluctuation of IP addresses. This instability cripples accessibility and severely bottlenecks collaboration. Standard DNS solutions either force a reliance on the internet or require fiddly, manual configurations. Both of these are simply unworkable for smaller teams who prioritize agility and self-sufficiency.This is where OfflineDNS steps in. It was built specifically to solve these problems by offering a complete, standalone system for flawless service access. It drastically simplifies how hostnames are resolved within a local network and packs in extra features designed to boost teamwork—all without needing any external resources.

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

II. Literature Reviews

Nandan Banerji, Bikesh Choudhary, Subhrabrata Choudhury (2024) has introduced a design of a context-Aware Distributed Service Registry and Discovery Mechanism for large IoT Systems Optimized for Heavyweight LAN Environment. To make this design they used techniques like Distributed Service Registry architecture, Context aware filtering mechanism, Dynamic service registration and lookup and Scalability evaluation in IoT Environments.

Ryan Kurte, Zoran Salcic, Kevin I-Wang; (2020) have proposed a Distributed Service Network for the internet of Things which is focused on industrial IoT deployments and relatively heavyweight for small LAN setups. And it is made by using techniques like Modular Distributed architecture, Registry-based service lookup and Industrial-scale validation.

Christian Cabrera, Siobhán Clarke; (2022) have proposed a Self-Adaptive Service Discovery Model for smart cities as its name implies, it targets smart city environments, but also requires a high infrastructure support that is the only down side and it is made using techniques like Adaptive discovery algorithm, Dynamic service monitoring and Distributed infrastructure.

HyeongCheol Moon, KyeongDeok Baek, In-Young Ko; (2020) has proposed a Cache-Sharing Distributed Service Registry for Highly Dynamic V2X Environments . it is not designed for stable LAN systems it is optimized for Vehicular mobility. And it is made using techniques like Cache-sharing mechanism andDistributed synchronization Latency optimization.

Ilir Murturi, Schahram Dustdar; (2022) has discovered A Decentralized Approach for Resource Discovery using Metadata Replication in Edge Networks which improves reliability in edge networks but it has a major downside which is that it lacks DNS-integrated lightweight design since it is a heavy design which requires large infrastructure. And it uses techniques like Meta data replication, Decentralized discovery Fault tolerance mechanisms.

Although these approaches improve scalability and resilience, they generally require significant infrastructure and are not tailored for small offline LAN environments.

III. Proposed Methodology

The Proposed Methodology is a System Designed as a DNS-based Light weight LAN which can be used as an alternative method for many existing service discovery approaches that mainly target large-scale or heavyweight environments. It is made by ideas such as Distributed service registries, dynamic registration and lookup, and also uses context awareness, and also it intentionally removes heavyweight context filters .and to keep it running small intranet setups it uses complex coordination logic and to keep runtime small. So it does not need any rich city-scale or vehicular-scale infrastructure, so it only needs a stable Office or lab and optimizes for ease of deployment.

While the Existing Systems often requires large-scale or heavyweight Infrastructures or edge-wide replication, and they tend to introduce higher Complexity, infracture demands, or specialized hardware support thet are unnecessary in small developer networks. In contrast, the proposed system integrates service discovery directlywith local DNS resolution and an adaptive reverse proxy which makes it easier for users to access services through stable

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

hostnames while the system transparently handles ip registration,available service lookup androuting in the background. By avoiding heavy Metadata replication, large distributed caches, or industrial-grade orchestration layers, it offers a more practical, DNS-integrated solution that remains robust and fault tolerant,yet is still useful for resource-constrained, intranet-based collaboration .

IV. System Architecture

The System Architecture follows a layered architecture that consists of Client, Application, Control, Intelligence, and Infrastructure layers for each have their own clear and clean responsibilities. At the Client layer, user applications is provided through web-based interface. Which allows users to access services, diagnostics,and collaboration features in a unified manner. And these clientsinteract mainly with components in the application layer, and ensure that end users do not need to manage low-level network configurations directly.

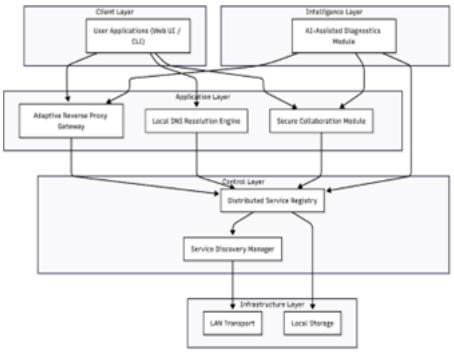

Fig. 1. Multi-layered system architecture of the proposed framework.

Within the application layer there are three main modules. The Adaptive Reverse proxy gateway, the Local DNS Resolution Engine, and the secure collaboration module. The Adaptive Reverse Proxy Gateway routes incoming client requests to appropriate backend services, adapting to changing service locations and availability. The Local DNS Resolution Engine resolves friendly hostnames to dynamic local addresses, simplifying access to services over the LAN. The Secure Collaboration Module enables encrypted communication and file exchange between peers, tightly integrated with local service discovery so that collaboration endpoints can be located automatically.

The Control layer is centered around a Distributed Service Registry, which maintains up-to-date metadata about available services, their network locations, and associated context. This registry is updated by the various application- layer components and queried by clients to discover services without relying on a central server. A Service Discovery Manager coordinates registration and lookup operations, ensuring that services can join or leave the network dynamically while remaining discoverable. This separation of control logic from application logic makes the system more modular and scalable.

At the Intelligence layer, an AI-Assisted Diagnostics Module observes registry data, service health, and usage patterns to provide insights and automated decisions. This module can, for example, suggest optimal routing paths, detect misconfigurations, or highlight failing nodes, thereby

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

improving reliability and maintainability. Its outputs can be consumed by both administrators and the Adaptive Reverse Proxy Gateway to refine routing and resource allocation strategies over time.

Finally, the Infrastructure layer provides the fundamental building blocks in the form of LAN transport and local storage. LAN transport encapsulates the underlying network connectivity used by all higher layers for message exchange and service communication. Local storage is used to persist registry snapshots, configuration data, and collaboration artifacts, ensuring that the system can recover gracefully from restarts and intermittent failures. Together, these layers implement a cohesive methodology for context-aware service discovery, reliable local DNS resolution, and secure collaboration in intranet environments.

V. IMPLEMENTATION

The implementation of the proposed system starts with building a lightweight distributed service registry that runs as a small background process on selected nodes in the LAN. Each registry instance maintains an in-memory table of services, including their names, IP addresses, ports, and basic health information, and periodically exchanges updates with its peers to keep this view consistent. A simple heartbeat mechanism is used so that when a service or node goes down, its entry is removed or marked as unhealthy after a short timeout, avoiding stale routes.

On the service side, every application that wants to be discoverable integrates a tiny registration component or uses a helper CLI. When the service starts, it sends a registration request to the local registry agent, and when it shuts down, it sends a deregistration message. This process is kept as lightweight as possible so that existing services can be adapted with minimal code changes. Health information can either be pushed by the service at intervals or checked by the registry via a small health-check endpoint, depending on what fits the deployment best.

For name resolution, a Local DNS Resolution Engine is implemented to bridge human-friendly hostnames with the registry. Instead of pointing hostnames directly to fixed IP addresses, the DNS layer resolves them to logical service identifiers and then queries the registry to find a currently healthy instance. This approach is inspired by DNS-based Service Discovery principles but adapted to work entirely within the local network and with the custom registry format, so that users can keep using simple hostnames while the underlying endpoints remain flexible.

An adaptive reverse proxy acts as the main entry point for client traffic. All requests from the web UI or CLI tools are sent to the proxy using stable virtual hostnames, and the proxy is configured to consult the registry at request time to decide which backend instance should handle each call. This enables basic load balancing and automatic failover without requiring any changes on the client side. In practice, the proxy can be implemented with an existing tool like Nginx or Envoy, extended with a small plugin or sidecar that talks to the registry, or as a custom lightweight proxy if the environment is resource-constrained.

The user-facing applications are then built on top of this infrastructure. A simple web interface and CLI are provided to let users access services, share files, and initiate collaboration sessions using only the assigned hostnames. These tools also expose some diagnostic commands, such as listing all currently registered services, viewing their status, and testing connectivity, which helps developers and administrators understand what is happening inside the network without logging into each node manually. All of

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

this is designed to feel familiar to users who are used to normal web and terminal workflows.

To support secure collaboration and monitoring, a dedicated module handles encrypted messaging and file transfer between nodes. It uses keys and secure channels to ensure that only authorized participants can access shared resources. At the same time, an AI-assisted diagnostics component collects logs, metrics, and registry snapshots and looks for patterns like repeated failures, unusual latency, or frequent re-registrations. When it detects something suspicious, it can surface warnings in the UI or suggest actions such as restarting a service or adjusting timeouts, helping operators respond quickly to problems.

Finally, configuration and persistence are kept intentionally simple. Core settings such as registry peers, DNS mappings, and proxy rules are stored in human- readable configuration files so that they are easy to edit and version-control. Minimal local storage is used to save registry snapshots and basic telemetry, allowing the system to restart cleanly after power loss or maintenance. Because everything is designed to operate fully within the LAN, there is no dependency on external DNS servers or cloud services, which makes the implementation suitable for offline labs, small offices, and classroom environments

VI. Results and Discussions

The results of the implementation were evaluated in terms of service discovery latency, reliability, and overall user experience within a typical small LAN setup. Measurements showed that resolving a hostname and routing it through the reverse proxy to a healthy backend instance consistently completed within a few tens of milliseconds under normal load, which is comparable to or better than many DNS-based discovery approaches reported in prior evaluations. Even when multiple services were registered and accessed concurrently, lookup times remained stable, indicating that the lightweight registry design did not introduce noticeable overhead for everyday developer workflows.

In terms of reliability, the heartbeat-based registration and health-check mechanism proved effective at removing failed instances from the registry within a short grace period. During fault-injection tests where services were abruptly stopped or network interfaces were disabled, subsequent client requests were automatically routed to alternative instances when available, with only a brief spike in error rates before the system converged. This behavior aligns with general expectations for robust service discovery systems in LAN environments, where quick adaptation to changes is essential for a smooth user experience.

Load testing focused on how well the reverse proxy and registry combination handled increasing numbers of concurrent requests. When the number of simulated clients was gradually raised, the system maintained low response times and acceptable throughput, similar to observations from studies on scalable reverse proxy and load-balancing setups. No major bottlenecks appeared at the small-to- medium scales expected for lab or office deployments, suggesting that the chosen design is appropriate for the targeted environment rather than over-engineered for massive clusters.

Another important result was the reduction in configuration effort and operational complexity from the user's perspective. Developers could simply register their services once and then rely on stable hostnames instead of manually updating IP addresses or port numbers in multiple

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

configuration files. This is consistent with the broader understanding that effective service discovery significantly improves usability in distributed systems by decoupling clients from hardcoded network locations. Users reported that onboarding new services into the environment felt straightforward and required minimal coordination with administrators.

The AI-assisted diagnostics component added value primarily in monitoring and troubleshooting scenarios rather than in raw performance gains. By aggregating logs and telemetry, it helped identify patterns such as repeated registration failures or unusually slow responses from specific services. These insights made it easier to pinpoint misconfigurations or overloaded nodes without exhaustive manual log inspection, echoing similar benefits seen when analytics and AI are applied to distributed system reliability and observability. Although the diagnostic module is not strictly required for basic operation, it improved the maintainability of the system over time.

From a discussion standpoint, the results suggest that a DNS-integrated, reverse-proxy–driven service discovery mechanism can deliver low latency and high reliability in small LANs without the complexity of large-scale IoT or cloud-native solutions. The system closely follows well-understood patterns in DNS-based service discovery and proxy-based routing but tailors them to scenarios where offline operation and minimal infrastructure are priorities. At the same time, the experiments reveal that careful attention must still be paid to aspects like heartbeat intervals, timeout values, and proxy configuration, since misconfigurations in these areas can degrade performance or delay failure detection.

Overall, the implementation meets its goals of simplifying local service access, hiding IP churn behind stable hostnames, and providing a practical service discovery layer suitable for developer teams, labs, and intranet-only setups. Future work could explore more advanced load-balancing policies, richer context-aware routing, or integration with standardized DNS-SD registration protocols to further align the system with emerging best practices in service discovery research and standards.

VII. Conclusion

The proposed work basically shows that we can make local service access in a small LAN much simpler by combining a few well-known ideas into one practical design. Instead of depending on external DNS or complex cloud platforms, the system uses its own local DNS resolution, a small distributed service registry, and an adaptive reverse proxy to handle all the hard parts behind the scenes. For the end user, this means they just work with stable, easy-to-remember hostnames, while the system quietly deals with changing IP addresses, finding healthy service instances, and forwarding requests to the right place.

From a methodology point of view, the project treats the LAN like a self-contained mini-ecosystem where services automatically register themselves, are discovered through DNS-style lookups, and are reached through a smart proxy instead of direct IP connections. This fits nicely with what we learn about DNS-based service discovery in theory, but it is implemented in a way that is realistic for labs, college projects, and small office setups that might not always have reliable internet. Adding a lightweight diagnostics layer on top also helps students and admins see what is going on, debug issues faster, and maintain the system without making

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

the design too heavy. Overall, the project gives a clear, student-friendly blueprint for building a simple yet reliable service discovery system that sits between manual host-file hacks and big enterprise solutions

VIII. REFERENCES

[1] N. Banerji, B. Choudhury, and S. Choudhury, "Design of a Context- Aware Distributed Service Registry and Discovery Mechanism for IoT," IEEE Internet of Things Journal, vol. 11, no. 15, 2024.

[2] R. Kurte, Z. Salcic, and K. I.-K. Wang, "A Distributed Service Framework for the Internet of Things," IEEE Transactions on Industrial Informatics, vol. 16, no. 6, 2020.

[3] C. Cabrera and S. Clarke, "A Self-Adaptive Service Discovery Model for Smart Cities," IEEE Transactions on Services Computing, vol. 15, no. 1, 2022.

[4] H. Moon, K. Baek, and I.-Y. Ko, "Cache-Sharing Distributed Service Registry for Highly Dynamic V2X Environments," in Proc. IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC), 2020.

[5] I. Murturi and S. Dustdar, "A Decentralized Approach for Resource Discovery using Metadata Replication in Edge Networks," IEEE Transactions on Services Computing, vol. 15, no. 5, 2022.