# A Design and Implementation of Dual-Stack Aware Authentication System for Enterprise Captive Portal

Surasak Sanguanpong[†], Kasom Koht-Arsa[†,‡]
[†]Department of Computer Engineering; [‡]Engineering Computer Center,
Faculty of Engineering, Kasetsart University
Bangkok, Thailand
{Surasak.S, Kasom.K}@ku.ac.th

***Abstract*—In the IPv4/IPv6 dual-stack environment, enterprises critically need a captive portal based authentication system that can bind a user account to both IPv4 and IPv6 addresses, on the machine the user log-in, and release the binding when the user log-out. Aggravating users by requiring them to do multiple log-in, one per address, is out of the question. In this paper, we present a design and implementation of a dual-stack aware captive portal system for a large-scale network. We propose a mechanism for dual address (IPv4/IPv6) discovery and service bundle authentication of both addresses. The proposed system offers a total solution for dual-stack authentication in enterprise network to support widespread adoption of IPv6.**

***Keywords—captive portal; authentication; dual-stack; IPv6;***

## I. INTRODUCTION

Captive portal [1] is a technique that imposes users' authentication by presenting their credentials before gaining access to the network. A captive portal typically uses a web browser as an authentication device with displaying conditions of use or usage policy. In general, an authentication is performed based on unique IP address basis. Today's operating systems allow IPv4 and IPv6 addresses both operate on the same network interface under a dual-stack environment [2]. A question arises. What to choose for client's source address when a DNS resolution of a destination host yields both IPv4 and IPv6 addresses? Current operating systems by default prefer IPv6 source address over IPv4 [3]. However, the IPv4 address on the client's interface might be subsequently used when a DNS resolution of another destination host yields an IPv4 address. Thus, IPv4 and IPv6 addresses in the dual-stack client are usually not active at the same time.

The existent and independency of IPv4 and IPv6 addresses requires users to authenticate twice on an address by address basis. With this reason, most captive portals often deny IPv6-based request from clients to avoid multiple authentications, which can cause confusion to users. Thus, transparent and automatic binding of dual addresses to authenticate all at once becomes an alternative solution and can further encourage adoption of IPv6.

This paper describes a design and implementation of automatic IPv4 and IPv6 addresses binding for dual-stack networks. Our goal is to simplify user authentication and allow transparency of an authentication based on IP address basis. We propose a design and implementation of a dual-stack aware captive portal system for a large-scale network. The rest of paper is organized as follows: Section II reviews related work and known approaches for dual-stack authentication. Section III describes system components and functionalities. Section IV describes methods and techniques for dual address discovery and binding them for bundle authentication. Section V presents usage statistic and performance measurement results, and Section VI concludes the paper.

## II. RELATED WORK

User authentication is considerably important for legal purpose. IPv6 brings a new challenge for user authentication corresponding to client's IP address. The system proposed by Brno University's researchers [4] supported dual-stack lawful interception. IPv4 and IPv6 addresses are discovered by monitoring client's RADIUS authentication and DHCP traffic. This address monitoring approach might be a suitable platform for traffic tracking in lawful enforcement applications, but is considerably inappropriate for captive portal, where an IP address must be explicitly identified from an authentication process.

Several captive portals are properly designed for small and medium networks [5][6][7]. The Barcelona's Open Access Network Testbed [8] is one of large-scale captive portals, unfortunately it does not support dual-stack infrastructure.

The Opengate [6][7] is a captive portal supported dual-stack authentication, but it does not support IPv6-only client. Discovering of an IP address simply relies on the first encountered HTTP request, and it must be IPv4 packet. If the first HTTP request is IPv6 packet (by preference [3]), the firewall demands the client to renew a request with an IPv4 packet. This limitation introduces a delay of response in authentication and might be a potential source of poor user experiences. Moreover, one unit of Opengate can support only one of class C subnet. To operate on multiple subnets, it requires to deploy multiple Opengate units. Compared to the Opengate, our approach supports both single stack and dual-stack clients under a single system management and control. Moreover, there is no restriction on type of the first HTTP request, either IPv4 or IPv6 packet is allowed.

## III. System Design

In this section, we overview a general captive portal and propose a design of supplementary components in detail. Typical captive portal may consist of two main components, i.e., a gateway and an authenticator. The gateway is generally implemented using firewall to inhibit unauthenticated client's traffic, while HTTP and HTTPS request will be intercepted and redirected to the authenticator. A login server, serving as an authenticator, displays to the user a login page. After receiving a correct credential, the login server communicates with the firewall allowing traffic according to the client's IP address to pass through. Figure 1 shows a typical network connectivity diagram with Firewall and Login Server placement.

A firewall-based captive portal system does not only have to discard undesired packets, but also must divert clients' HTTP and HTTPS requests to authentication server and keep track of users' sessions. In our proposed system, these two functions are assigned to a stateless HTTP redirector and a session manager respectively.

### *A. Lightweight Stateless HTTP Redirector*

General purpose web server liked Apache is not designed to meet two crucial requirements for redirection service: (1) minimal processing delay, and (2) minimal resources consumption specifically on CPU and memory under high concurrent request. We propose a dual-stack *Lightweight Stateless HTTP Redirector* (LSHR) as a dedicated web server to perform redirection service inside Firewall. A compact design of LSHR provides rapid response with minimal resource consumption and allows Firewall to fully perform its primary functions without performance degradation.

The LSHR is implemented by a small daemon with embedded TCP/IP stack and HTTP server in the single user space. Figure 2 shows the standard HTTP server compared to LSHR. Client's TCP packets intercepted by LSHR are classified into five categories for serving redirection task.

Table *I* shows categories of TCP packets to be inspected. For each category of an incoming packet, LSHR replies with a corresponded packet to the client. As the first checkpoint, LSHR resists against SYN-flooding attacks created by Worms, Trojans, or malwares infected clients. Attacks mitigation can be accomplished through rapidly dropping the malicious request based on (1) rate limiting, and (2) a predefined list of common software update agents and non-browser based applications.

### *B. User Session Manager*

The *User Session Manager* (USM) keeps track of all active users sessions. IPv4 and IPv6 addresses from the same login session are stored as a single entry in the database. For every successful login, logout, and scheduled time-out, Login Server provides session information to be managed by USM and Firewall.

Figure 3 shows the information exchange between a client, Firewall, and Login Server based on LSHR and USM functions as previously described.

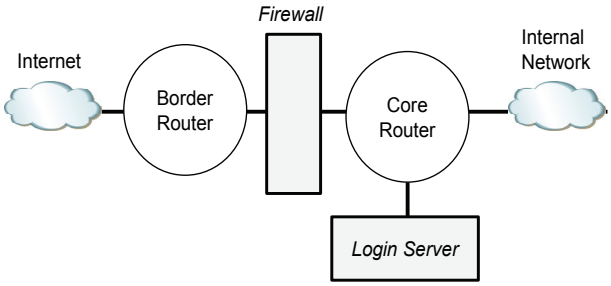


Fig. 1. Network diagram with Firewall and Login Server

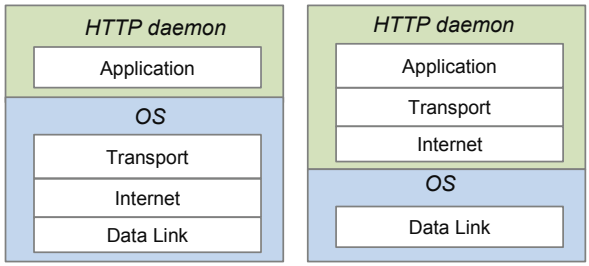


Fig. 2. A comparative of standard HTTP server and proposed LSHR

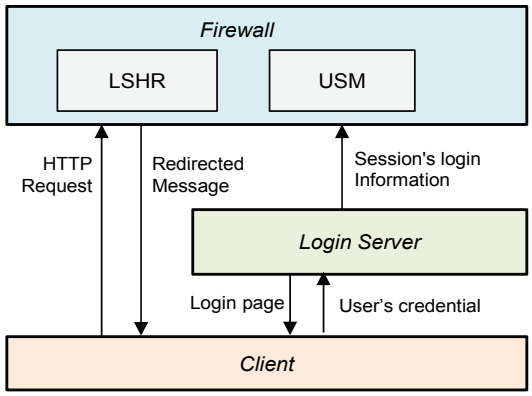


Fig. 3. Information exchange for authentication procedure

Table I. Five Categories of TCP Packets Inspected by LSHR

| Incoming packet | Reply packet |
|---|---|
| SYN | SYN + ACK |
| ACK without payload | (None) |
| ACK with HTTP request | ACK+ FIN + HTTP redirection |
| ACK with unknown payload | (None) |
| ACK + FIN | ACK |

## IV. Address Discovery and Binding Technique

We propose the *Dual Address Discovery* (DAD) as a mechanism to activate and bind together IPv4 and IPv6 client addresses. Briefly, Login Server presents a login page embedded with IPv4 and IPv6 host names to the client. If both addresses are available, two consecutive DNS resolution operation inherently activate them with two corresponded login requests (one for IPv4, and the other for IPv6) to Login Server. Login Server then associates them with a 48-bit hash code into 3-tuples of (*IPv4*, *IPv6*, *hash code*) for further authentication.

### A. DAD Address to hostname mapping

Figure 4 shows a sample of login. The IPv4 and IPv6 addresses on the page notify to the user that two addresses are going to be authenticated. Each address is generated by an image tag (`img`) where a source (`src`) of image is specified by a CGI (Common Gateway Interface) script tying to a specific host name of Login Server described in next paragraphs. The unique hash code attached with the script notifies Login Server to authenticate IPv4/IPv6 addresses simultaneously when receiving a correct user's credential.

To provide high availability and redundancy of login services, Login Server may be implemented using a cluster of servers. Each physical server can contain several instances of virtual machines (VM) for login services. Load balancing among them can be achieved by round-robin DNS [9] or by layer 4 switching. Each VM is assigned with three types of host names serving different purposes as follows:

*1) Type I: Dual-stack host names:* Login Server with dual-stack host name serves as the main login server from redirection by Firewall. The name is assigned with `loginX`, where `X`=1, 2, 3,…, *N* (*N* is the number of available VM). The login page with this host name contains two tags for address resolution as shown by HTML code in Fig. 4. The first tag is for IPv4 address (A DNS record mapping), and the second tag is for IPv6 address (AAAA DNS record mapping). The A and AAA record respectively are in fact the Type II and Type III host names. This type of host name allow the first HTTP request to be any type of packet as described in Section II.

*2) Type II: IPv4-only host names:* This type of host name (`loginX-v4`) is specifically used to obtain client's IPv4 address (for IPv4 stack).

*3) Type III: IPv6-only host names:* The last type of host name (`loginX-v6`) is specifically used to obtain client's IPv6 address (for IPv6 stack).

Figure 5 illustrates the relationship between three types of host names and how to bind client's address with each type. The `loginX` (Type I) is assigned to main login page, while `loginX-v4` (Type II) and `loginX-v6` (Type III) are used in embedded tags generated by `loginX`, separately serving each IP stack.

### B. DAD Sequence Diagram

Figure 6 shows DAD messages exchanged between the client, Login Server and Session Manager. There are three main parts as numbered in the figure as follows:

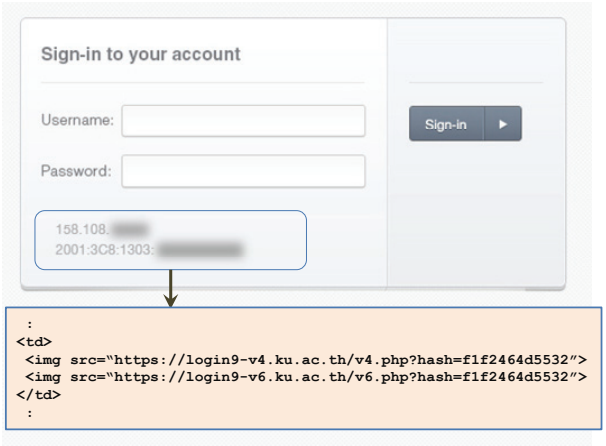


Fig. 4. Sample login page with embedded CGI generated images

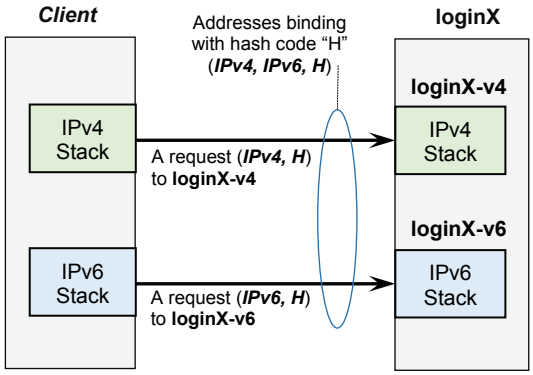


Fig. 5. Address binding with a mapping host names by each IP stack

*1) Address Acquisition:* The client, either with IPv4 or IPv6 address, sends a HTTP request (redirected from firewall) to the Login Server using Type I host name (`loginX`). The Login Server sends a login page reply back to the client. The login page contains two HTML tags: one for IPv4, the other for IPv6. Both addressed are attached with the same hash code for unique session identification.

*2) Address Binding:* When the client receives the login page, both HTML tags enforce the client to reopen two requests corresponding to IPv4 tag (Type II host name) and IPv6 tag (Type III host name) to Login Server again. The DNS resolutions from these requests yield client's IPv4 and IPv6 addresses, while the graphical images of the two addresses are sent back to inform the user. Subsequently, the system waits for user's credential submission.

*3) Address Authentication:* The credential with IPv4 and IPv6 addresses are together submitted to Login Server. Both are forwarded to USM and will be further processed by the firewall. Finally, the client with IPv4/IPv6 addresses are allowed to access network services.

## V. Measurements And Results

The proposed system is carried out in the Kasetsart University (KU) network. The collection of data presented in this paper started from June 6, 2012 (World IPv6 Day) to December 31, 2012. Our system handled over 88,000 authentication sessions per day for 100,000+ user accounts (66,000+ students and 34,000+ faculties and staffs in 4 campuses and 20 research stations). The recent measurement from international Internet exchange showed that the amount of IPv6 shared 7.97% of total university traffic [10].

To investigate the number of dual-stack ready devices, we classified users into three groups from log files and found that (1) 41% of users never login with IPv6. Users in this group might use operating systems that do not support IPv6, or they were in subareas, where IPv6 was not available, (2) 57% of users occasionally login with IPv6, and (3) 2% of users always login with IPv6. The results revealed that nearly 60% of total users had IPv6-enabled devices.

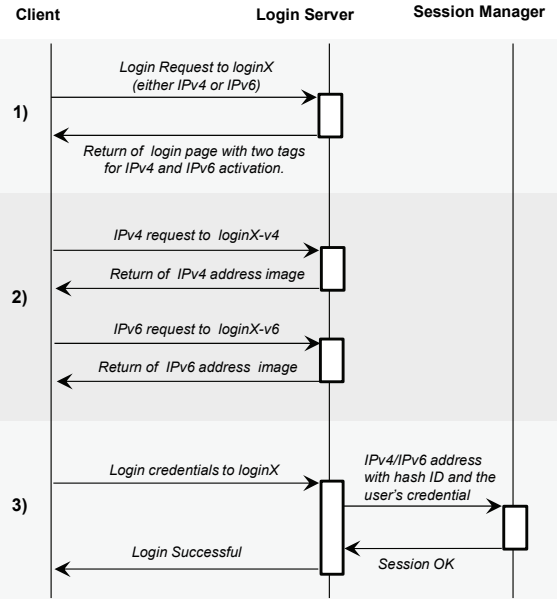


Fig. 6. Sequence diagram for Dual Address Discovery (DAD)

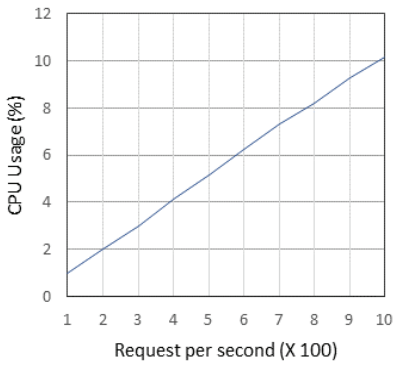


Fig. 7. Request rate and percentage of CPU usage of LSHR

We measure the performance of LSHR's redirection operation by simulating a number of HTTP requests from 100 to 1,000 requests per second. The result in Fig. 7 shows that CPU usage increases linearly with increased load. As an example, approximately 1% of CPU usage is consumed to handle 100 requests per second. This rate is about four times of the peak rate occurred in our network (25 requests per second and it consumes 0.25% of CPU usage). A comparative test shows that LSHR could deliver up to 10 times of performance gain over the standard Apache server.

## VI. Conclusion

We proposed a method for IPv4/IPv6 dual-stack address activation and their binding to simplify users' authentication in captive portal environment. The proposed stateless HTTP redirector and the addresses binding procedure show that the system consumes minimal CPU power and could handle enterprise-scale services. The system supports a 'per IPv4-IPv6 pair' basis, i.e., only one of IPv4 addresses and one of IPv6 addresses can be bound together, but it does not allow users to authenticate IPv4 address in the first place and later ask for binding with the IPv6 address or vice versa. Our proposed system offers a total solution for dual-stack authentication in a large-scale network and encourages adoption of IPv6.

## Acknowledgment

The authors would like to thank Dr. Supaporn Erjongmanee, Dr. Wannana Soontornnaruerangsee, and the anonymous reviewers for their constructive comments and suggestions that help improve the manuscript. Special thanks to KU's Computer Center for equipment support and computing resources.

## References

[1] K. Koht-arsa, A. Phonphoem, and S. Sanguanpong, "Architectural Design for Large-Scale Campus-wide Captive Portal," in 43rd IEEE International Carnahan Conference on Security Technology, 2009.

[2] E. Nordmark and R. Gilligan, "Basic Transition Mechanism for IPv6 Hosts and Routers," RFC 4213, 2005.

[3] D. Thaler, R. Draves, A. Matsumoto, and T. Chown, "Default Address Selection for Internet Protocol Version 6 (IPv6)," RFC 6724, 2012.

[4] L. Polčák, M. Grégr, M. Kajan, P. Matoušek, and V. Veselý, "Designing Lawful Interception in IPv6 Networks," in Security and Protection of Information Conference, 2011, pp. 114-126.

[5] M. Brunato, R. L. Cigno, and D. Severina, "Managing Wireless HotSpots: the Uni-Fy Approach" in MedHocNet, 2006,

[6] K. Watanabe, M. Otani, S. Tadaki, and Y. Watanabe, "Opengate on the Cloud," in 26th International Conference on Advanced Information Networking and Applications Workshops, 2012.

[7] M. Otani, K. Eguchi, H. Eto, K. Watanabe, S. Tadaki, and Y. Watanabe "Implementation of IPv6 functions for a network user authentication system opengate," in Proceedings of the 33rd annual ACM SIGUCCS fall conference, 2005, pp. 283-286.

[8] J. Barcelo, C. Macian, J. Infante, M. Oliver, and A. Sfairopoulou, "Barcelona's Open Access Network Testbed," in the 2nd International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities, 2006.

[9] V. Cardellini, M. Colajanni, and P.S. Yu, "Dynamic load balancing on Web-server systems," IEEE Internet Computing, Vol. 3, No. 3, May.-Jun. 1999, pp. 28-39.

[10] World IPv6 Launch, "Network operator measurements (Jul. 16, 2013)," [Online]. Available: http://www.worldipv6launch.org/measurements/. [Accessed: Aug. 17, 2013].