Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: April 1, 2018

K. Larose
D. Dolson
Sandvine

September 28, 2017

CAPPORT Architecture
draft-ietf-capport-architecture-00

Abstract

This document aims to document consensus on the CAPPORT architecture. DHCP or Router Advertisements, ICMP, and an HTTP API are used to provide the solution. The role of Provisioning Domains (PvDs) is described.

Table of Contents

1. Introduction

In this document, "Captive Portal" is used to describe a network to which a device may be voluntarily attached, such that network access is limited until some requirements have been fulfilled. Typically a user is required to use a web browser to fulfil requirements imposed by the network operator, such as reading advertisements, accepting an acceptable-use policy, or providing some form of credentials.

Implementations generally require a web server, some method to allow/block traffic, and some method to alert the user. Common methods of alerting the user involve modifying HTTP or DNS traffic.

Problems with captive portal implementations have been described in [I-D.nottingham-capport-problem]. [If that document cannot be published, consider putting its best parts into an appendix of this document.]

This document standardizes an architecture for implementing captive portals that provides tools for addressing most of those problems. We are guided by these principles:

- Solutions SHOULD NOT require the forging of responses from DNS or HTTP servers, or any other protocol. In particular, solutions SHOULD NOT require man-in-the-middle proxy of TLS traffic.

- Solutions MUST operate at the layer of Internet Protocol (IP) or above, not being specific to any particular access technology such as Cable, WiFi or 3GPP.

- Solutions SHOULD allow a device to be alerted that it is in a captive network when attempting to use any application on the network. (Versus requiring a user to visit a clear-text HTTP site in order to receive a notification.)

- The state of captivity SHOULD be explicitly available to devices (in contrast to modification of DNS or HTTP, which is only indirectly machine-detectable by the client--by comparing responses to well-known queries with expected responses).

- The architecture MUST provide a path of incremental migration, acknowledging a huge variety of portals and end-user device implementations and software versions.

- The architecture SHOULD improve security by providing mechanisms for trust, allowing alerts from trusted network operators to be distinguished from attacks from untrusted agents.

A side-benefit of the architecture described in this document is that devices without user interfaces are able to identify parameters of captivity. (However, this document does not yet describe a mechanism for such devices to escape captivity.)

The architecture uses the following mechanisms:

- Network provisioning protocols provide end-user devices with a URI for the API that end-user devices query for information about what is required to escape captivity. DHCP, DHCPv6, and Router-Advertisement options for this purpose are available in [RFC7710]. Other protocols (such as RADIUS), Provisioning Domains [I-D.bruneau-intarea-provisioning-domains], or static configuration may also be used. A device MAY query this API at any time to determine whether the network is holding the device in a captive state.

- End-user devices are notified of captivity with ICMP/ICMP6 messages in response to traffic. This notification can work with any Internet protocol, not just clear-text HTTP. This notification does not carry the portal URI; rather it provides a notification to the User Equipment that it is in a captive state.

- Receipt of the ICMP/ICMP6 messages informs an end-user device that it is captive. In response, the device SHOULD query the provisioned API to obtain information about the network state. The device MAY take immediate action to satisfy the portal (according to its configuration/policy).

The architecture attempts to provide privacy, authentication, and safety mechanisms to the extent possible.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

1.2. Terminology

Captive Network: A network which limits communication of attached devices to restricted hosts until the user has satisfied Captive Portal Conditions, after which access is permitted to a wider set of hosts (typically the internet).

Captive Portal Conditions: site-specific requirements that a user or device must satisfy in order to gain access to the wider network.

Captive Portal Enforcement: The network equipment which enforces the traffic restriction and notifies the User Equipment it is in a captive state.

Captive Portal User Equipment: Also known as User Equipment. A device which has voluntarily joined a network for purposes of communicating beyond the constraints of the captive network.

Captive Portal Server: The web server providing a user interface for assisting the user in satisfying the conditions to escape captivity.

2. Components

2.1. User Equipment

The User Equipment is the device that a user desires to be attached to a network with full access to all hosts on the network (e.g., to have Internet access). The User Equipment communication is typically restricted by the Captive Portal Enforcement, described in Section 2.4, until site-specific requirements have been met.

At this time we consider only devices with web browsers, with web applications being the means of satisfying Captive Portal Conditions.

- An example interactive User Equipment is a smart phone.
- SHOULD support provisioning of the URI for the Captive Portal API (e.g., by DHCP)
- MAY distinguish Captive Portal API access per network interface, in the manner of Provisioning Domain Architecture [RFC7556].
- SHOULD have a mechanism for notifying the user of the Captive Portal
- SHOULD have a web browser so that the user may navigate the Captive Portal user interface.
- SHOULD be able to receive and validate the Captive Portal ICMP message types, and to access the Captive Portal API in response.
- MAY restrict application access to networks not granting full network access. E.g., a device connected to a mobile network may be connecting to a WiFi network; the operating system MAY avoid updating the default route until network access restrictions have been lifted (excepting access to the Captive Portal server). This has been termed "make before break".

None of the above requirements are mandatory because (a) we do not wish to say users or devices must seek access beyond the captive network, (b) the requirements may be fulfilled by manually visiting the captive portal web application, and (c) legacy devices must continue to be supported.

2.2. Provisioning Service

Here we discuss candidate mechanisms for provisioning the User Equipment with the URI of the API to query captive portal state and navigate the portal.

2.2.1. DHCP or Router Advertisements

A standard for providing a portal URI using DHCP or Router Advertisements is described in [RFC7710]. The CAPPORT architecture expects this URI to indicate the API described in Section 2.3.

Although it is not clear from RFC7710 what protocol should be executed at the specified URI, some readers might have assumed it to be an HTML page, and hence there might be User Equipment assuming a browser should open this URI. For backwards compatibility, it is RECOMMENDED that the server check the "Accept" field when serving the URI, and serve HTML pages for "text/html" and serve the API for "application/json". [REVISIT: are these details appropriate?]

2.2.2. Provisioning Domains

Although still a work in progress, [I-D.bruneau-intarea-provisioning-domains] proposes a mechanism for User Equipment to be provided with PvD Bootstrap Information containing the URI for a JSON file containing key-value pairs to be downloaded over HTTPS. This JSON file would fill the role of the Captive Portal API described in Section 2.3.

One key-value pair can be used to indicate the network has restricted access, requiring captive portal navigation by a user. E.g., key="captivePortal" and value=<URI of portal>. The key-value pair should provide a different result when access is not restricted. E.g., key="captivePortal" and value="".

This JSON file is extensible, allowing new key-value pairs to indicate such things as network access expiry time, URI for API access by IOT devices, etc.

The PvD server MUST support multiple (repeated) queries from each User Equipment, always returning the current captive portal information. The User Equipment is expected to make this query upon receiving (and validating) an ICMP Captive Portal message (see Section 2.5).

2.3. Captive Portal API Server

The purpose of a Captive Portal API is to permit a query of Captive Portal state without interrupting the user. This API thereby removes the need for a device to perform clear-text "canary" HTTP queries to check for response tampering.

The URI of this API will have been provisioned to the User Equipment. (Refer to Section 2.2).

This architecture expects the User Equipment to query the API when the User Equipment attaches to the network and multiple times thereafter. Therefore the API MUST support multiple repeated queries from the same User Equipment, returning current state of captivity for the equipment.

At minimum the API MUST provide: (1) the state of captivity and (2) a URI for a browser to present the portal application to the user. The API SHOULD provide evidence to the caller that it supports the present architecture.

When user equipment receives (and validates) ICMP Captive Portal alerts, the user equipment SHOULD query the API to check the state. The User Equipment SHOULD rate-limit these API queries in the event of ICMP flooding by an attacker. (See Section 6.)

The API MUST be extensible to support future use-cases by allowing extensible information elements. Suggestions include quota information, expiry time, method of providing credentials, security token for validating ICMP messages.

This document does not specify the details of the API.

The CAPPORT API SHOULD support TLS for privacy and server authentication.

2.4. Captive Portal Enforcement

The Captive Portal Enforcement component restricts network access to User Equipment according to site-specific policy. Typically User Equipment is permitted access to a small number of services and is denied general network access until it has performed some action.

The Captive Portal Enforcement component:

- Allows traffic through for allowed User Equipment.
- Blocks (discards) traffic and sends ICMP notifications for disallowed User Equipment.
- Permits disallowed User Equipment to access necessary APIs and web pages to fulfill requirements of exiting captivity.
- Updates allow/block rules per User Equipment in response to operations from the Captive Portal back-end.

As an upgrade path, captive portals MAY continue to support methods that work today, such as modification of port-80 HTTP responses to

redirect users to the portal. Various user-equipment vendors probe canary URLs to identify the state captivity [reference Mariko Kobayashi's survey]. While doing so, ICMP messages SHOULD also be sent, to activate work-flows in supporting devices. [TODO: give some thought to precise recommendations for backwards compatibility.]

2.5. ICMP/ICMP6

ICMP messages have been selected for indicating to a sender that packets could not be delivered for reason of a network policy (in particular, captive portal). ICMP is already used to indicate reasons that packets could not be delivered (network unreachable, port unreachable, packet too large, etc.).

A mechanism to trigger captive portal work-flows in the User Equipment is proposed in [I-D.wkumari-capport-icmp-unreach].

The Captive Portal Enforcement function is REQUIRED to send such ICMP messages when disallowed User Equipment attempts to send to the network. These ICMP messages MUST be rate-limited to a configurable rate.

The ICMP messages MUST NOT be sent to the Internet devices. The indications are only sent to the User Equipment.

The User Equipment operating system is NOT REQUIRED to deliver the impact of the ICMP message to the application that triggered it. The User Equipment might be able to satisfy the Captive Portal requirements quickly enough that existing transport connections are not impacted.

2.6. Component Diagram

The following diagram shows the communication between each component.

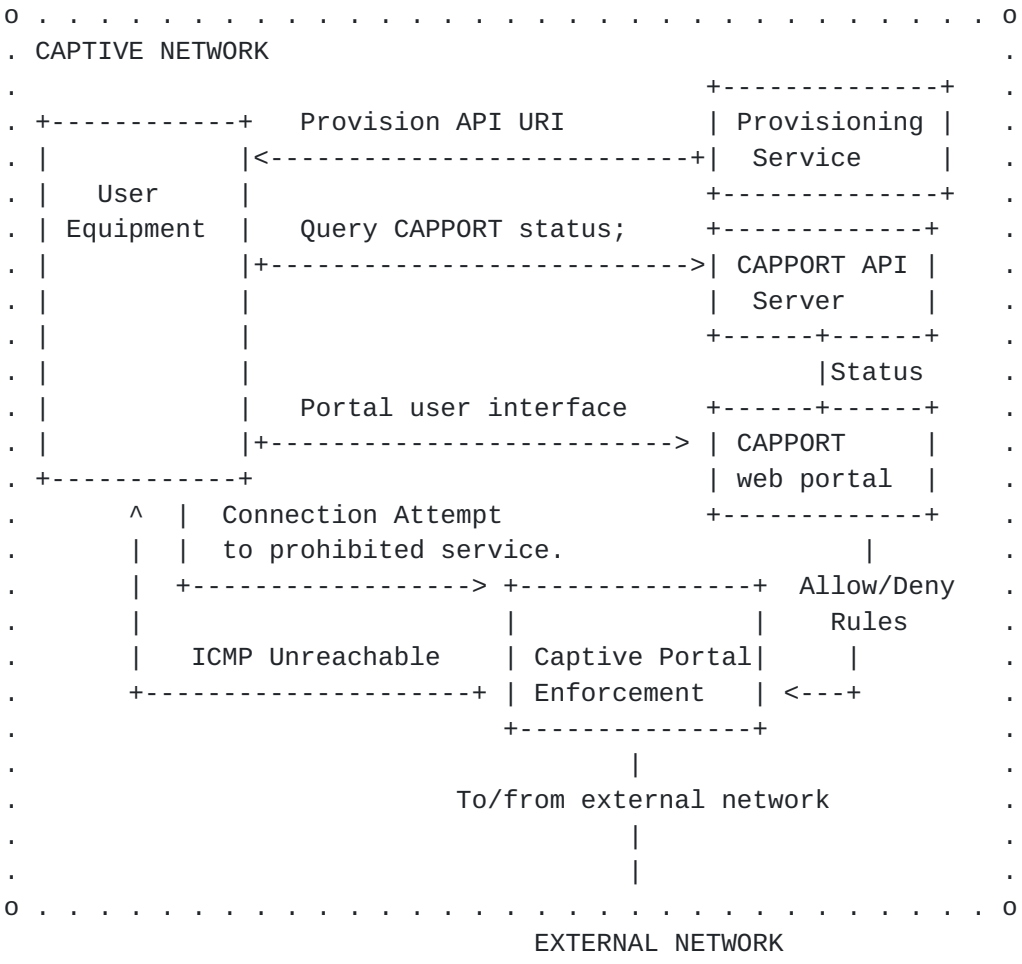

Figure 1: Captive Portal Architecture Component Diagram

In the diagram:

- During provisioning (e.g., DHCP), the User Equipment acquires the URI for the CAPPORT API.

- The User Equipment queries the API to learn of its state of captivity. If captive, the User Equipment presents the portal user interface to the user.

- The User Equipment attempts to communicate to the external network through the Captive Portal enforcement device.

- The Captive Portal Enforcement device either allows the User Equipment's packets to the external network, or responds with an ICMP message.

- The CAPPORT web portal server directs the Captive Portal Enforcement device to either allow or deny external network access for the User Equipment.

Although the provisioning, API, and web portal functions are shown as discrete blocks, they could of course be combined into a single element.

3. Solution Workflow

This section aims to improve understanding by describing a possible workflow of solutions adhering to the architecture.

3.1. Initial Connection

This section describes a possible work-flow when User Equipment initially joins a Captive Network.

1. The User Equipment joins the Captive Network by acquiring a DHCP lease, RA, or similar, acquiring provisioning information.

2. The User Equipment learns the URI for the Captive Portal API from the provisioning information (e.g., [RFC7710]).

3. The User Equipment accesses the CAPPORT API to receive parameters of the Captive Network, including web-portal URI. (This step replaces the clear-text query to a canary URL.)

4. If necessary, the User navigates the web portal to gain access to the external network.

5. The Captive Portal API server indicates to the Captive Portal Enforcement device that the User Equipment is allowed to access the external network.

6. The User Equipment attempts a connection outside the captive network

7. If the requirements have been satisfied, the access is permitted; otherwise the "Expired" behavior occurs.

8. The User Equipment accesses the network until conditions Expire.

3.2. Conditions Expire

This section describes a possible work-flow when conditions expire and the user visits the portal again (e.g., low quota, or time expiry).

1. Pre-condition: the Captive Portal Enforcement has been configured to detect an expiry condition, which has now occurred.

2. The User Equipment sends a packet to the outside network.

3. The Captive Portal Enforcement detects that the packet is from an expired User Equipment.

4. The Captive Portal Enforcement sends an ICMP message to the User Equipment indicating that it needs to refresh its access. [I-D.wkumari-capport-icmp-unreach].

5. The User Equipment verifies the ICMP message is plausible.

6. The User Equipment queries the Captive Portal API to refresh parameters and status of the Captive Network.

7. If necessary, the User once again navigates the web portal to gain access to the external network.

8. The Captive Portal API Server gives more quota (time, bytes, etc.) to the User Equipment by indicating to the Captive Portal Enforcement the new, extended quota.

9. The User Equipment accesses the external network.

4. Acknowledgements

The authors thank various individuals for their feedback on the mailing list and during the IETF98 hackathon: David Bird, Eric Kline, Alexis La Goulette, Alex Roscoe, Darshak Thakore, and Vincent van Dam.

5. IANA Considerations

This memo includes no request to IANA.

6. Security Considerations

6.1. Trusting the Network

When joining a network, some trust is placed in the network operator. This is usually considered to be a decision by a user on the basis of the reputation of an organization. However, once a user makes such a decision, protocols can support authenticating a network is operated by who claims to be operating it. The Provisioning Domain Architecture [RFC7556] provides some discussion on authenticating an operator.

Given that a user chooses to visit a Captive Portal URI, the URI location SHOULD be securely provided to the user's device. E.g., the DHCPv6 AUTH option can sign this information.

If a user decides to incorrectly trust an attacking network, they might be convinced to visit an attacking web page and unwittingly provide credentials to an attacker. Browsers can authenticate servers but cannot detect cleverly misspelled domains, for example.

6.2. Authenticated APIs

The solution described here assumes that when the User Equipment needs to trust the API server, server authentication will be utilized using TLS mechanisms.

6.3. Risk of Nuisance Captive Portal

It is possible for any user on the Internet to send ICMP packets in an attempt to cause the receiving equipment to go to the captive portal. This has been considered and addressed in the following ways:

The ICMP packet does not carry the URL, making this method safer than HTTP 3xx-redirect methods currently in use. The User Equipment does not have to use clear-text HTTP to solicit the URL of the portal.

Because the ICMP messages will carry embedded packets sent by the sender, the receiver of the ICMP message can validate that the transport header is plausibly one it sent (i.e., the transport-layer 5-tuple matches an open connection; there is no need to save every packet it sent). This validation requires an off-path attacker to guess the 5-tuple in order to affect a flow. It is trivial for an on-path attacker to send a plausible ICMP packet. (This is not a new ICMP attack.) The impact of getting a valid ICMP packet to the User Equipment is that it will visit the CAPPORT API to check the status. For this reason we recommend the User Equipment rate-limit these requests to the API.

We considered that the ICMP packet could carry a short secret token that would be known to the User Equipment and Captive Portal Enforcement device but would not be available to an attacker, even to an on-path attacker. Although possible to guess by brute force, the impact would be at worst a nuisance visit to the API. We suggest that a 32-bit token would be sufficient to deter nuisance attacks.

Even when redirected, the User Equipment securely authenticates with API servers.

6.4. User Options

The ICMP messaging informs the User Equipment that it is being held captive. There is no requirement that the User Equipment do something about this. Devices MAY permit users to disable automatic reaction to captive-portal indications for privacy reasons. However, there is the trade-off that the user doesn't get notified when network access is restricted. Hence, end-user devices MAY allow users to manually control captive portal interactions, possibly on the granularity of Provisioning Domains.

7. References

7.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC7556] Anipko, D., Ed., "Multiple Provisioning Domain Architecture", RFC 7556, DOI 10.17487/RFC7556, June 2015, <https://www.rfc-editor.org/info/rfc7556>.

[RFC7710] Kumari, W., Gudmundsson, O., Ebersman, P., and S. Sheng, "Captive-Portal Identification Using DHCP or Router Advertisements (RAs)", RFC 7710, DOI 10.17487/RFC7710, December 2015, <https://www.rfc-editor.org/info/rfc7710>.

7.2. Informative References

[I-D.bruneau-intarea-provisioning-domains] Pfister, P., Schinazi, D., Pauly, T., Vyncke, E., and B. Bruneau, "Discovering Provisioning Domain Names and Data", draft-bruneau-intarea-provisioning-domains-02 (work in progress), July 2017.

[I-D.nottingham-capport-problem] Nottingham, M., "Captive Portals Problem Statement", draft-nottingham-capport-problem-01 (work in progress), April 2016.

[I-D.wkumari-capport-icmp-unreach]
Bird, D. and W. Kumari, "Captive Portal ICMP Messages", draft-wkumari-capport-icmp-unreach-02 (work in progress), April 2017.

Authors' Addresses

Kyle Larose
Sandvine
408 Albert Street
Waterloo, ON N2L 3V3
Canada

Phone: +1 519 880 2400
Email: klarose@sandvine.com

David Dolson
Sandvine
408 Albert Street
Waterloo, ON N2L 3V3
Canada

Phone: +1 519 880 2400
Email: ddolson@sandvine.com