WifiDog Protocol Specifications

Alexandre Carmel-Veilleux
<acv@miniguru.ca>

Document Version 0.1
October 8th, 2006

This page intentionally left blank.

1 Table of Contents

2 Revision History

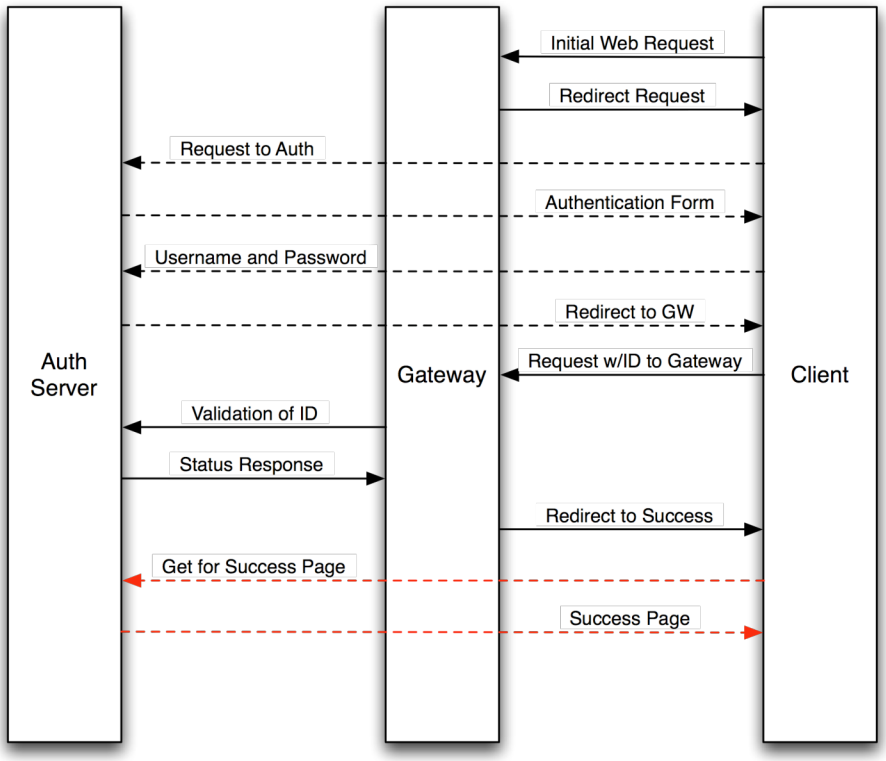
Version 0.1 – October 8th, 2006 – Alexandre Carmel-Veilleux:

- Initial version.

3 Introduction

3.1 Purpose

It has been nearly three years since Île Sans Fil has started work on the WifiDog captive portal. The initial version was hacked together with no real idea of how the Gateway and the Auth Server would talk to each other. Soon after work had started, a bunch of the ISF volunteers met at Daniel Drouet's house and one of the by-products of this meeting was the base of our protocol. Here it is:



For the last few years this along with a few lines of text was the only documentation we had. This diagram served us well, being published in Linux Journal (http://www.linuxjournal.com/article/8352, Figure 1).

However, this diagram does not tell the whole story. Some parts of the protocol are left out (the Ping interface) and the exact syntax of the different requests is only available to those who wish to read the code.

This document will detail the exact protocol as it currently stands. Attempts will be made to keep it up to date. Along with it, a number of tools to test the protocol are to be produced. These will make debugging easier by simulating quickly either an Auth Server or a Gateway. They will also further illustrate this documentation with a reference implementation.

3.2 Protocol Overview

There are three components involved in the WifiDog protocol. Two of which are part of the WifiDog distribution: the Gateway and the Auth Server. The final component is the client's Web Browser.

The basic idea is that when a new user connects to the wireless network, she will open her Web Browser and try to access a Web Site. The Gateway will capture this connection and instead redirect her to the Auth Server. The Auth Server will somehow authenticate the user and issue her a Token. The Auth Server will then redirect the user back to the Gateway with her new Token. The Gateway will independently validate the Token with the Auth Server and if it is good, send the user to either a Welcome Page or the Web Site she was trying to access in the first place.

This has been how the protocol was explained until now. Missing are three more scenarios.

The first scenario is a connected user. Every few minutes, the gateway iterates through all the connected users, make sure they are still active, retrieve statistics about them and then re-authenticates their Token with the Auth Server. If the user is no longer active, her connection is closed and the Auth Server is informed and provided with final usage statistics. The Auth Server can also instruct the Gateway to terminate a user's connection.

The second scenario is a new user. When a new user has created an account but not yet validated her email address, she has a limited "grace" period. The whole transaction plays out like the default scenario but a different code is returned during the Token validation so the user is issued different access then a fully validated user.

The third scenario is the Ping mechanism. The Gateway pings the Auth Server periodically with some health information. The Auth Server logs the health information and returns a Pong message. Failure to connect with the Auth Server causes the Gateway to mark it as bad.

These three scenarios are known to anyone who's ever administered an installation of WifiDog but the exact mechanism is not documented unless you dig in the code. The next sections will detail the exact protocol syntax used.

3.3 Conventions

Throughout this document, the following conventions will be used:

`http://auth.ilesansfil.org/wifidog/` will be used as the default location of the Auth Server.

`240.0.0.1` will be used as the default IP for the Gateway (it is an IANA reserved IP and not allocated to anyone).

4 Ping Mechanism

Periodically – as set by the CheckInterval directive in wifidog.conf – the Gateway checks the Auth Server to make sure it is still alive. This is accomplished by issuing a Ping Request using the following syntax:

GET Request:

http://auth.ilesansfil.org/wifidog/ping/?gw_id=**<Gateway ID>**&sys_uptime=**<Gateway Uptime>**&sys_memfree=**<Gateway Free Memory>**&sys_load=**<Gateway Load Average>**&wifidog_uptime=**<WifiDog Uptime>**

Where:

gw_id	Gateway ID, text string.
sys_uptime	Gateway Uptime, seconds in UNIX epoch format.
sys_memfree	Free memory, KiloBytes free as unsigned integer.
sys_load	Load average for the last minute, UNIX format.
wifidog_uptime	Uptime of WifiDog software, same format as for sys_uptime.

Example:

http://auth.ilesansfil.org/wifidog/ping/?gw_id=TESTGWID&sys_uptime=637&sys_memfree=3728&sys_load=0.08&wifidog_uptime=621

Translation:

The Gateway TESTGWID has been running for 10 minutes and 37 seconds with 3728 KB of free memory, a load average of 0.08 for the last minute and the WifiDog software itself has been running for 10 minutes and 21 seconds.

Response:

Pong

The exact response from the server does not matter so long as it contains the word above. The current gateway implementation only checks for the presence of that word in the response.