Independent Submission
Request for Comments: 6108
Category: Informational
ISSN: 2070-1721

C. Chung
A. Kasyanov
J. Livingood
N. Mody
Comcast
B. Van Lieu
Unaffiliated

February 2011

# Comcast's Web Notification System Design

## Abstract

The objective of this document is to describe a method of providing critical end-user notifications to web browsers, which has been deployed by Comcast, an Internet Service Provider (ISP). Such a notification system is being used to provide near-immediate notifications to customers, such as to warn them that their traffic exhibits patterns that are indicative of malware or virus infection. There are other proprietary systems that can perform such notifications, but those systems utilize Deep Packet Inspection (DPI) technology. In contrast to DPI, this document describes a system that does not rely upon DPI, and is instead based in open IETF standards and open source applications.

## Status of This Memo

## Copyright Notice

## Table of Contents

## 1. Introduction

Internet Service Providers (ISPs) have a need for a system that is capable of communicating with customers in a nearly immediate manner, to convey critical service notices such as warnings concerning likely malware infection. Given the prevalence of the web browser as the predominant client software in use by Internet users, the web browser is an ideal vehicle for providing these notifications. This document describes a system that has been deployed by Comcast, a broadband ISP, to provide near-immediate notifications to web browsers.

In the course of evaluating potential solutions, the authors discovered that the large majority of commercially available systems utilized Deep Packet Inspection (DPI) technology. While a DPI-based system would certainly work, Comcast and other ISPs are trying to avoid widespread deployment and use of DPI, and are searching for alternatives. Thus, Comcast desired to use a system that is based on open standards and non-proprietary software, and that did not require the use of DPI. While the system described herein is specific to the Data-Over-Cable Service Interface Specifications (DOCSIS, [CableLabs_DOCSIS]) networks used by most cable-based broadband ISPs, concepts described in this document can generally be applied to many different types of networks should those ISPs be interested in alternatives to DPI.

## 2. High-Level Design of the System

The web notification system design is based on the use of the Internet Content Adaptation Protocol (ICAP) [RFC3507]. The design uses open source applications, which are the Squid web proxy, GreasySpoon ICAP server, and Apache Tomcat. ICAP, an existing IETF protocol, allows for message transformation or adaptation. An ICAP client passes a HyperText Transport Protocol (HTTP, [RFC2616]) response to an ICAP server for content adaption. The ICAP server in turn responds back to the client with the HTTP response containing the notification message by using the "respmod" method defined in Section 3.2 of [RFC3507].

## 3. Design Requirements

This section describes all of the key requirements taken into consideration by Comcast for the design of this system. This information is provided in order to convey important design choices that were made in order to avoid the use of DPI, among other things. An "Additional Background" paragraph is included with each requirement to provide additional information, context, or other useful explanation.

### 3.1. General Requirements

R3.1.1. Must Only Be Used for Critical Service Notifications
Additional Background: The system must only provide critical notifications, rather than trivial notifications. An example of a critical, non-trivial notification, which is also the primary motivation of this system, is to advise the user that their computer is infected with malware, that their security is at severe risk and/or has already been compromised, and that it is recommended that they take immediate, corrective action NOW.

R3.1.2. Must Use TCP Port 80
Additional Background: The system must provide notifications via TCP port 80, the well-known port for HTTP traffic. Since the large majority of customers use a web browser as their primary application, this was deemed the best method to provide them with an immediate, critical notification.

R3.1.3. Must Support Block Listing
Additional Background: While unlikely, it is possible that the HyperText Markup Language (HTML, [RFC2854]) or JavaScript [RFC4329] used for notifications may cause problems while accessing a particular website. Therefore, such a system must be capable of using a block list of website Uniform Resource Identifiers (URIs, [RFC3986]) or Fully Qualified Domain Names (FQDNs, Section 5.1 of [RFC1035]) that conflict with the system, so that the system does not provide notifications in these cases, in order to minimize any errors or unexpected results. Also, while extensive development and testing has been performed to ensure that this system does not behave in unexpected ways, and standard ICAP (which has been in use for many years) is utilized, it is critical that if it does behave in such a way, there must be a method to rapidly exempt specific URIs or FQDNs.

R3.1.4. Must Not Cause Problems with Instant Messaging (IM) Clients Using TCP Port 80
Additional Background: Some IM clients use TCP port 80 in their communications, often as an alternate port when standard, well-known ports do not work. Other IM clients may in fact use TCP port 80 by default, in some cases even being based in a web browser. Therefore, this system must not conflict with or cause unexpected results for IM clients (or any other client types) that use TCP port 80.

R3.1.5. Must Handle Pre-Existing Active TCP Sessions Gracefully
Additional Background: Since the web notification system may temporarily re-route TCP port 80 traffic in order to provide a critical notification, previously established TCP port 80 sessions must not be disrupted while being routed to the proxy layer. Also, since the critical web notification occurs at a well-defined point in time, it is logical to conclude that an end user may well have an active TCP port 80 session in progress before the notification is sent, and which is still active at the time of the notification. It is therefore important that any such connections must not be reset, and that they instead must be handled gracefully.

R3.1.6. Must Not Use TCP Resets
Additional Background: The use of TCP resets has been widely criticized, both in the Internet community generally and in [RFC3360]. In Comcast’s recent history, for example, the company was criticized for using TCP resets in the course of operating a DPI-based network management system. As such, TCP resets as a function of the system must not be used.

R3.1.7. Must Be Non-Disruptive
Additional Background: The web notification system must not disrupt the end-user experience, for example by causing significant client errors.

R3.1.8. User Notification Acknowledgement Must Stop Further Immediate Notifications
Additional Background: Once a user acknowledges a critical notification, the notification should immediately stop. Otherwise, the user may believe the system is stuck in an error state and may not believe that the critical notification is valid. In addition, it is quite possible that the user will be annoyed that the system did not react to his acknowledgement.

R3.1.9. Non-Modification of Content Should Be Maintained
Additional Background: The system should not significantly alter the content of the HTTP response from any website the user is accessing.

R3.1.10. Must Handle Unexpected Content Gracefully
Additional Background: Sometimes, developers and/or implementers of software systems assume that a narrow range of inputs to a system will occur, all of which have been thought of beforehand by the designers. The authors

believe this is a poor assumption to make in the design and implementation of a system and, in contrast, that unexpected or even malformed inputs should be assumed. As a result, the system must gracefully and transparently handle traffic that is unexpected, even though there will be cases when the system cannot provide a critical web notification as a result of this. Thus, widely varying content should be expected, and all such unexpected traffic must be handled by the system without generating user-perceived errors or unexpected results.

R3.1.11. Web Content Must Not Be Cached
Additional Background: Maintaining the privacy of users is important. As such, content flowing through or incidentally observed by the system must not be cached.

R3.1.12. Advertising Replacement or Insertion Must Not Be Performed Under ANY Circumstances
Additional Background: The system must not be used to replace any advertising provided by a website, or to insert advertising into websites. This therefore includes cases where a web page already has space for advertising, as well as cases where a web page does not have any advertising. This is a critical area of concern for end users, privacy advocates, and other members of the Internet community. Therefore, it must be made abundantly clear that this system will not be used for such purposes.

## 3.2. Web Proxy Requirements

R3.2.1. Open Source Software Must Be Used
Additional Background: The system must use an open source web proxy server. (As noted in Section 2 and Section 4.1, Squid has been chosen.) While it is possible to use any web proxy, the use of open source enables others to easily access openly available documentation for the software, among the other benefits commonly attributed to the use of open source software.

R3.2.2. ICAP Client Should Be Integrated
Additional Background: The web proxy server should have an integrated ICAP client, which simplifies the design and implementation of the system.

R3.2.3. Access Control Must Be Implemented
Additional Background: Access to the proxy must be limited exclusively to the IP addresses of users for which notifications are intended, and only for limited periods of time. Furthermore, since a Session Management Broker (SMB) is utilized, as described in Section 4.1 below, then the proxy must restrict access only to the address of the SMB.

### 3.3. ICAP Server Requirements

R3.3.1. Must Provide ICAP Response Support
Additional Background: The system must support response adaptation, in accordance with [RFC3507]. An ICAP client passes a HyperText Transport Protocol (HTTP, [RFC2616]) response to an ICAP server for content adaption. The ICAP server in turn responds back to the client with the HTTP response containing the notification message by using the "respmod" method defined in Section 3.2 of [RFC3507].

R3.3.2. Must Provide Consistency of Critical Notifications
Additional Background: The system must be able to consistently provide a specific notification. For example, if a critical alert to notify a user that they are infected with malware is desired, then that notification should consistently look the same for all users and not vary.

R3.3.3. Must Support Multiple Notification Types
Additional Background: While the initial and sole critical notification sent by the system is intended to alert users of a malware infection, malware is a rapidly and continuously evolving threat. As a result of this reality, the system must be able to evolve to provide different types of critical notifications. For example, if malware begins to diverge into several different categories with substantially different implications for end users, then it may become desirable to provide a notification that has been narrowly tailored to each category of malware.

R3.3.4. Must Support Notification to Multiple Users Simultaneously
Additional Background: The system must be able to simultaneously serve notifications to different users. For example, if 100 users have been infected with malware and critically need to be notified about this security problem, then the system must be capable of providing the notification to several users at a time, or all of the users at the same time, rather than to just one user at a time.

### 3.4. Messaging Service Requirements

R3.4.1. A Messaging Service Must Be Used
Additional Background: The Messaging Service, as described in Section 4.1 below, caches the notifications for each specific user. Thus, the notification messages are cached by the system and do not have to be retrieved each time a notification is needed. As a result, the system can be more easily scaled to provide notification to multiple users simultaneously, as noted in an earlier requirement ("Must Support Notification to Multiple Users Simultaneously").

R3.4.2. Must Process Acknowledgements on a Timely Basis
Additional Background: The Messaging Service must quickly process notification acknowledgements by end users, as noted in an earlier requirement ("User Notification Acknowledgement Must Stop Further Immediate Notifications").

R3.4.3. Must Ensure Notification Targeting Accuracy
Additional Background: The Messaging Service must ensure that notifications are presented to the intended users. For example, if the system intends to provide a critical notification to User A and User B, but not User C, then User C must not be sent a notification.

R3.4.4. Should Keep Notification Records for Customer Support Purposes
Additional Background: The Messaging Service should maintain some type of record that a notification has been sent to a user, in case that user inquires with customer support personnel. For example, when a user is presented with the critical notification advising them of a malware infection, that user may choose to call Comcast’s Customer Security Assurance team, in the customer service organization. As a result, a Customer Security Assurance representative should be able to confirm that the user did in fact receive a notification concerning malware infection in the course of providing assistance to the end user in remediating the malware infection.

## 4. Implementation Details

This section defines and documents the various core functional components of the system, as they are implemented. These components are then shown in a diagram to describe how the various components are linked and relate to one another.

### 4.1. Functional Components Described, as Implemented

This section accurately and transparently describes the software (S) packages used by the system described herein, as well as all of the details of how the system functions. The authors acknowledge that there may be multiple alternative software choices for each component; the purpose of this section is to describe those selections that have been made and deployed.

S4.1.1. Web Proxy: The system uses Squid Proxy, an open source web proxy application in wide use, which supports an integrated ICAP client.

S4.1.2. ICAP Server: The system uses GreasySpoon, an open source application. The ICAP server retrieves the notifications from the Messaging Service cache when content adaption is needed.

S4.1.3. Customer Database: The Customer Database holds the relevant information that the system needs to provide a critical notification to a given user. The database may also hold the status of which users were notified and which users are pending notification.

S4.1.4. Messaging Service: The system uses Apache Tomcat, an open source application. This is a process engine that retrieves specific web notification messages from a catalog of possible notifications. While only one notification is currently used, concerning malware infection, as noted in Section 3.3 the system may eventually need to provide multiple notifications (the specific requirement is "Must Support Multiple Notification Types"). When a notification for a specific user is not in the cache, the process retrieves this information from the Customer Database and populates the cache for a specific period of time.

S4.1.5. Session Management Broker (SMB): A Load Balancer (LB) with a customized layer 7 inspection policy is used to differentiate between HTTP and non-HTTP traffic on TCP port 80, in order to meet the requirements documented in Section 3 above. The system uses a LB from A10 Networks. The SMB functions as a full stateful TCP proxy with the ability to forward packets from existing TCP sessions that do not exist in the internal session table (to meet the specific requirement "Must Handle Pre-Existing Active TCP Sessions Gracefully"). New HTTP sessions are load balanced to the web proxy layer either transparently or using source Network Address Translation (NAT [RFC3022]) from the SMB.

Non-HTTP traffic for established TCP sessions not in the SMB session table is simply forwarded to the destination transparently via the TCP proxy layer (again, to meet the specific requirement "Must Handle Pre-Existing Active TCP Sessions Gracefully").

### 4.2. Functional Diagram, as Implemented

```
   +--------+        +------------+        +----------+
   |  ICAP  | <----> | Messaging  | <----> | Customer |
   | Server |        |  Service   |        | Database |
   +--------+        +------------+        +----------+
     ^
     |                +----------+
     |                |          |
     |      +-------> | Internet | <-------+
     |      |         |          |         |
     |      |         +----------+         |
     |      |              ^               |
     v      v              |               |
   +----------+            v               v
   |+--------+|        +-------+       +--------+
   ||  ICAP  || <----> |  SMB  | <---> | Access |
   || Client ||        +-------+       | Router |
   |+--------+|                        +--------+
   || SQUID  ||                            ^
   || Proxy  ||                            |
   |+--------+|                            v
   +----------+                          +----------+
                                         |  CMTS*   |
                                         +----------+
                                            ^
                                            |
                                            v
                                       +------+
                                       |  PC  |
                                       +------+

    * A Cable Modem Termination System (CMTS)
      is an access network element.
```

Figure 1: Web Notification System - Functional Components

## 5. High-Level Communication Flow, as Implemented

In Section 4, the functional components of the system were described, and then shown in relation to one another in Figure 1 above. This section describes the high-level communication (C) flow of a transaction in the system, in order to explain the general way that the functions work together in action. This will be further explained in much more detail in later sections of this document.

C5.1. Setup of Differentiated Services (Diffserv): Using Diffserv [RFC2474] [RFC2475] [RFC2597] [RFC3140] [RFC3246] [RFC3260] [RFC4594], set a policy to direct TCP port 80 traffic to the web notification system's web proxy.

C5.2. Session Management: TCP port 80 packets are routed to a Session Management Broker (SMB) that distinguishes between HTTP or non-HTTP traffic and between new and existing sessions. HTTP packets are forwarded to the web proxy by the SMB. Non-HTTP packets such as instant messaging (IM) traffic are forwarded to a TCP proxy layer for routing to their destination, or the SMB operates as a full TCP proxy and forwards the non-HTTP packets to the destination. Pre-established TCP sessions on port 80 are identified by the SMB and forwarded with no impact.

C5.3. Web Proxy Forwards Request: The web proxy forwards the HTTP request on to the destination site, a web server, as a web proxy normally would do.

C5.4. On Response, Send Message to ICAP Server: When the HTTP response is received from the destination server, the web proxy sends a message to the ICAP server for the web notification.

C5.5. Messaging Service: The Messaging Service should respond with appropriate notification content or null response if no notification is cached.

C5.6. ICAP Server Responds: The ICAP server responds and furnishes the appropriate content for the web notification to the web proxy.

C5.7. Web Proxy Sends Response: The web proxy then forwards the HTTP response containing the web notification to the client web browser.

C5.8. User Response: The user observes the critical web notification, and clicks an appropriate option, such as: OK/ acknowledged, snooze/remind me later, etc.

C5.9. More Information: Depending upon the notification, the user may be provided with more information. For example, as noted previously, the system was designed to provide critical notifications concerning malware infection. Thus, in the case of malware infection, the user may be advised to go to a malware remediation web page that provides directions on how to attempt to remove the malware and attempt to secure hosts against future malware infection.

C5.10. Turn Down Diffserv: Once the notification transaction has completed, remove any special Diffserv settings.

## 6. Communication between Web Proxy and ICAP Server, as Implemented

The web proxy and ICAP server are critical components of the system. This section shows the communication that occurs between these two components.

```
   +------------+
   |  www URI   |
   +------------+
      ^      |
   (2)|      |(3)
      |      v
     +--------+     (4)     +--------+     (4)     +--------+
     |        |------------>|        |------------>|        |
     |        |     (5)     |        |     (5)     |        |
     | Proxy  |<------------|  ICAP  |<------------|  ICAP  |
     | Module |     (6)     | Client |     (6)     | Server |
     |        |------------>|        |------------>|        |
     |        |     (7)     |        |     (7)     |        |
     |        |<------------|        |<------------|        |
     +--------+             +--------+             +--------+
      ^      |
   (1)|      |(8)
      |      v
   +------------+              (9)             +------------+
   |            |----------------------------->|            |
   |  Browser   |              (10)            | Web Server |
   |            |<-----------------------------|            |
   +------------+                              +------------+
```

(1) - HTTP GET (TCP 80)

(2) - Proxy HTTP GET (TCP 80)

(3) - HTTP 200 OK w/ Response

(4) - ICAP RESPMOD

(5) - ICAP 200 OK

(6) - TCP Stream - Encapsulate Header

(7) - ICAP 200 OK Insert Message

(8) - HTTP 200 OK w/ Response + Message Frame

(9) - HTTP GET for Message

(10) - HTTP 200 w/ Message Content

Figure 2: Communication between Web Proxy and ICAP Server

## 7. End-to-End Web Notification Flow, as Implemented

This section describes the exact flow of an end-to-end notification, in order to show in detail how the system functions.

### 7.1. Step-by-Step Description of the End-to-End Web Notification Flow

Policy-Based Routing

1. TCP port 80 packets from the user that needs to be notified are routed to the web proxy via policy-based routing.

2. Packets are forwarded to the Session Management Broker, which establishes a session with the web proxy and routes the packets to the web proxy.

Web Proxy

1. The user’s HTTP request is directed to the web proxy.

2. The web proxy receives HTTP traffic and retrieves content from the requested website.

3. The web proxy receives the response and forwards it to the ICAP server for response adaptation.

4. The ICAP server checks the HTTP content in order to determine whether the notification message can be inserted.

5. The ICAP server initiates a request to the Messaging Service cache process with the IP address of the user.

6. If a notification message for the user exists, then the appropriate notification is cached on the Messaging Service. The Messaging Service then returns the appropriate notification content to the ICAP server.

7. Once the notification message is retrieved from the Messaging Service cache, the ICAP server may insert the notification message in the HTTP response body without altering or modifying the original content of the HTTP response.

8. The ICAP server then sends the response back to the web proxy, which in turn forwards the HTTP response back to the browser.

9. If the user’s IP address is not found or provisioned for a notification message, then the ICAP server should return a "204 No modifications needed" response to the ICAP client as defined in Section 4.3.3 of [RFC3507]. As a result, the user will not receive any web notification message.

10. The user observes the web notification, and clicks an appropriate option, such as: OK/acknowledged, snooze/remind me later, etc.

## 7.2. Diagram of the End-to-End Web Notification Flow

The two figures below show the communications flow from the web browser, through the web notification system.

Figure 3 illustrates what occurs when a notification request cannot be inserted because the notification type for the user’s IP address is not cached in the Messaging Service.

```
                            ICAP     ICAP    Message          Customer
         Browser   Proxy   Client   Server   Service  Internet    DB
           |  HTTP  |         |         |        |        |        |
           |  GET   | Proxy   |         |        |        |        |
           +------->| Request |         |        |        |        |
           |        +---------|---------|--------|------->|        |
           |        |         |         |        | 200 OK |        |
           |        |<--------|---------|--------|--------+        |
           |        | ICAP    |         |        |        |        |
           |        | RESPMOD | ICAP    |        |        |        |
           |        +-------->| RESPMOD | Check  |        |        |
           |        |         +-------->| Cache  |        |        |
           |        |         |         | for IP |        |        |
           |        |         |         | Match  |        |        |
           |        |         |         +------->|        |        |
           |        |         |         | Cache  |        |        |
           |        |         |         | Miss   |        |        |
           |        |         |         |<-------+ Request|        |
           |        |         | 204 No  |        | Type   |        |
           |        |         | Modif.  |        +--------|------->|
           |        |         | Needed  |        |        |        |
           |        | No      |<--------+        |        | Type   |
           |        | Insert  |         |        |        |Returned|
           | 200 OK |<--------+         |        |<-------|--------+
           | w/o    |         |         |        |        |        |
           | Insert |         |         |        |        |        |
           |<-------+         |         |        |        |        |
           |        |         |         |        |        |        |
```

Figure 3: End-to-End Web Notification Flow - with Cache Miss

Figure 4 illustrates what occurs when a notification request for the user’s IP address is cached in the Messaging Service.

```
                            ICAP     ICAP    Message         Customer
         Browser   Proxy   Client   Server   Service  Internet    DB
           |  HTTP  |         |         |        |        |        |
           |  GET   | Proxy   |         |        |        |        |
           +------->| Request |         |        |        |        |
           |        +---------|---------|--------|------->|        |
           |        |         |         |        | 200 OK |        |
           |        |<--------|---------|--------|--------+        |
           |        | ICAP    |         |        |        |        |
           |        | RESPMOD | ICAP    |        |        |        |
           |        +-------->| RESPMOD | Check  |        |        |
           |        |         +-------->| Cache  |        |        |
           |        |         |         | for IP |        |        |
           |        |         |         | Match  |        |        |
           |        |         |         +------->|        |        |
           |        |         |         | Cache  |        |        |
           |        |         |         | Hit    |        |        |
           |        |         | Insert  |<-------+        |        |
           |        | Return  | Type    |        |        |        |
           |        | 200 OK  |<--------+        |        |        |
           |        | with    |         |        |        |        |
           |        | Insert  |         |        |        |        |
           | 200 OK |<--------+         |        |        |        |
           | w/     |         |         |        |        |        |
           | Notify |         |         |        |        |        |
           |<-------+         |         |        |        |        |
           |        |         |         |        |        |        |
```

Figure 4: End-to-End Web Notification Flow - with Cache Hit

## 8. Example HTTP Headers and JavaScript for a Web Notification

The figure below shows an example of a normal HTTP GET request from the user’s web browser to www.example.com, a web server on the Internet.

```
------------------------------------------------------------------------
1.  HTTP GET Request to www.example.com
------------------------------------------------------------------------
http://www.example.com/

GET / HTTP/1.1
Host: www.example.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8.1.14)
        Gecko/20080404 Firefox/2.0.0.14
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Pragma: no-cache
------------------------------------------------------------------------
```

Figure 5: Example HTTP Headers for a Web Notification - HTTP GET

In the figure below, the traffic is routed via the web proxy, which communicates with the ICAP server and returns the response from www.example.com.  In this case, that response is a 200 OK, with the desired notification message inserted.

```
------------------------------------------------------------------------
2.  Response from www.example.com via PROXY
------------------------------------------------------------------------
HTTP/1.x 200 OK
Date: Thu, 08 May 2008 16:26:29 GMT
Server: Apache/2.2.3 (CentOS)
Last-Modified: Tue, 15 Nov 2005 13:24:10 GMT
Etag: "b80f4-1b6-80bfd280"
Accept-Ranges: bytes
Content-Length: 438
Connection: close
Content-Type: text/html; charset=UTF-8
Age: 18
X-Cache: HIT from localhost.localdomain
Via: 1.0 localhost.localdomain (squid/3.0.STABLE5)
Proxy-Connection: keep-alive
------------------------------------------------------------------------
```

Figure 6: Example HTTP Headers for a Web Notification - HTTP Response

The figure below shows an example of the web notification content inserted in the 200 OK response, in this example JavaScript code.

```
-----------------------------------------------------------------------
3.  Example of JavaScript containing Notification Insertion
-----------------------------------------------------------------------
<!--all elements used in a notification should have cascading style
sheet (css) properties defined to avoid unwanted inheritance from
parent page-->

<style type="text/css">
#example {
  position: absolute; left: 100px; top: 50px;
  z-index: 9999999; height: auto; width: 550px;
  padding: 10px;
  border: solid 2px black;
  background-color:#FDD017;
  opacity: 0.8; filter: alpha(opacity = 80);
}
</style>

<script language="javascript" type="text/javascript">
// ensure that content is not part of an iframe
if (self.location == top.location) {
  // this is a floating div with 80% transparency
  document.write(’<div id="example" name="example">’);
  document.write(’<h2>IMPORTANT MESSAGE</h2>’);
  document.write(’<p>Lorem ipsum dolor sit amet, consecteteur ’);
  document.write(’adipisicing elit, sed do eiusmod tempor ’);
  document.write(’incididunt ut labore et dolore magna aliqua. ’);
  document.write(’Ut enim ad minim veniam, quis nostrud ’);
  document.write(’exercitation ullamco laboris nisi ut aliquip ex ’);
  document.write(’ea commodo consequat.’);
  document.write(’</div>’);
}</script>
-----------------------------------------------------------------------
```

Figure 7: Example JavaScript Used in a Web Notification

## 9. Deployment Considerations

The components of the web notification system should be distributed throughout the network and close to end users. This ensures that the routing performance and the user’s web browsing experience remain excellent. In addition, a HTTP-aware load balancer should be used in each datacenter where servers are located, so that traffic can be spread across N+1 servers and the system can be easily scaled.

## 10. Security Considerations

This critical web notification system was conceived in order to provide an additional method of notifying end user customers that their computer has been infected with malware. Depending upon the specific text of the notification, users could fear that it is some kind of phishing attack. As a result, care has been taken with the text and any links contained in the web notification itself. For example, should the notification text change over time, it may be best to provide a general URI or a telephone number. In contrast to that, the notification must not ask for login credentials, and must not ask a user to follow a link in order to change their password, since these are common phishing techniques. Finally, care should be taken to provide confidence that the web notification is valid and from a trusted party, and/or that the user has an alternate method of checking the validity of the web notification. One alternate method of validating the notification may be to call customer support (in this example, Comcast’s Customer Security Assurance team); this explains a key requirement (specifically, "Should Keep Notification Records for Customer Support Purposes") in Section 3.4.

## 11. Debating the Necessity of Such a Critical Notification System

Some members of the community may question whether it is ever, under any circumstances, acceptable to modify Internet content in order to provide critical service notification concerning malware infection - even in the smallest of ways, even if openly and transparently documented, even if thoroughly tested, and even if for the best of motivations. It is important that anyone with such concerns recognize that this document is by no means the first to propose this, particularly as a tactic to combat a security problem, and in fact simply leverages previous work in the IETF, such as [RFC3507]. Such concerned parties should also study the many organizations using ICAP and the many software systems that have implemented ICAP.

In addition, concerned members of the community should review Section 1, which describes the fact that this is a common feature of DPI systems, made by DPI vendors and many, if not most, major networking equipment vendors. As described herein, the authors of this document are motivated to AVOID the need for widespread, ubiquitous deployment of DPI, via the use of both open source software and open protocols, and are further motivated to transparently describe the details of how such a system functions, what it IS intended to do, what it IS NOT intended to do, and purposes for which it WILL NOT be used.

The authors also believe it is important for ISPs to transparently disclose network management techniques and systems, and to have a venue to do so, as has been done here. In addition, the authors believe it is important for the IETF and other members of the Internet community to encourage and positively reinforce such disclosures. In the publishing of such a document for reference and comment by the Internet community, this may serve to motivate other ISPs to be similarly open and to engage the IETF and other organizations that are part of the Internet community. Not publishing such documents could motivate less disclosure on the part of ISPs and other members of the Internet community, increase the use of DPI, and decrease ISP participation in the critical technical bodies that make up parts of the Internet community.

In addition, it is critical that members of the community recognize the good motivations of ISPs like Comcast to combat the massive and continuing proliferation of malware, which is a huge threat to the security of average Internet users and now represents a multi-billion-dollar underground economy engaged in identity theft, financial fraud, transmission of spam, and other criminal activity. Such a critical notification system in fact is only necessary due to the failure of host-based security at defending against and preventing malware infection. As such, ISPs such as Comcast are being urged by their customers and by other parties such as security and/or privacy organizations, as well as governmental organizations, to take action to help solve this massive problem, since so many other tactics have been unsuccessful. For example, as Howard Schmidt, the Special Advisory for Cyber Security to President Obama, of the United States of America, said in 2005: "As attacks on home-based and unsecured networks become as prevalent as those against large organizations, the need for ISPs to do everything they can to make security easier for their subscribers is critical for the preservation of our nation’s information backbone. Additionally, there is tremendous potential to grow further the use of broadband around the world; and making safety and security part of an ISP’s core offering will enable the end user to fully experience the rich and robust benefits broadband provides".

## 12. Suggesting a Walled Garden as an Alternative

A "walled garden" refers to an environment that controls the information and services that a subscriber is allowed to utilize and what network access permissions are granted. Placing a user in a walled garden is therefore another approach that ISPs may take to notify users, and this method is being explored as a possible alternative in other documents and community efforts. As such, web notifications should be considered one of many possible notification methods that merit documentation.

However, a walled-garden approach can pose challenges and may in some cases be considered disruptive to end users. For example, a user could be playing a game online, via the use of a dedicated, Internet-connected game console, which would likely stop working when the user was placed in the walled garden. In another example, the user may be in the course of a telephone conversation, using a Voice Over IP (VoIP) device of some type, which would also likely stop working when the user was placed in the walled garden. In both cases, the user is not using a web browser and would not have a way to determine the reason why their service seemingly stopped working.

## 13. Intended Next Steps

Unfortunately, at the time of this writing, no existing working group of the IETF is focused on issues of malware infection and related issues. As a result, there was not a definite venue for this document, so it was submitted to the Independent Submissions Editor as an independent submission. While documentation and disclosure of this system are beneficial for the Internet community in and of itself, there are other benefits to having this document published. One of those reasons is that members of the community, including members of the IETF, have a stable document to refer to in the case of any potential new work that the community may undertake in the area of malware, security, and critical notification to end users. It is also hoped that, in the tradition of a Request for Comment, other members of the community may be motivated to propose alternative systems or other improvements.

## 14. Acknowledgements

The authors wish to thank Alissa Cooper for her review of and comments on the document, and Nevil Brownlee for his excellent feedback, as well as others who reviewed the document.

## 15. References

### 15.1. Normative References

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.

[RFC2474] Nichols, K., Blake, S., Baker, F., and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, December 1998.

[RFC2475] Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., and W. Weiss, "An Architecture for Differentiated Services", RFC 2475, December 1998.

[RFC2597] Heinanen, J., Baker, F., Weiss, W., and J. Wroclawski, "Assured Forwarding PHB Group", RFC 2597, June 1999.

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

[RFC2854] Connolly, D. and L. Masinter, "The ’text/html’ Media Type", RFC 2854, June 2000.

[RFC3022] Srisuresh, P. and K. Egevang, "Traditional IP Network Address Translator (Traditional NAT)", RFC 3022, January 2001.

[RFC3140] Black, D., Brim, S., Carpenter, B., and F. Le Faucheur, "Per Hop Behavior Identification Codes", RFC 3140, June 2001.

[RFC3246] Davie, B., Charny, A., Bennet, J., Benson, K., Le Boudec, J., Courtney, W., Davari, S., Firoiu, V., and D. Stiliadis, "An Expedited Forwarding PHB (Per-Hop Behavior)", RFC 3246, March 2002.

[RFC3260] Grossman, D., "New Terminology and Clarifications for Diffserv", RFC 3260, April 2002.

[RFC3507] Elson, J. and A. Cerpa, "Internet Content Adaptation Protocol (ICAP)", RFC 3507, April 2003.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RFC4329] Hoehrmann, B., "Scripting Media Types", RFC 4329, April 2006.

[RFC4594] Babiarz, J., Chan, K., and F. Baker, "Configuration Guidelines for DiffServ Service Classes", RFC 4594, August 2006.

## 15.2. Informative References

[CableLabs_DOCSIS]
CableLabs, "Data-Over-Cable Service Interface Specifications", CableLabs Specifications, Various DOCSIS Reference Documents, <http://www.cablelabs.com/specifications/archives/docsis.html>.

[RFC3360] Floyd, S., "Inappropriate TCP Resets Considered Harmful", BCP 60, RFC 3360, August 2002.

## Authors’ Addresses

Chae Chung
Comcast Cable Communications
One Comcast Center
1701 John F. Kennedy Boulevard
Philadelphia, PA  19103
US
EMail: chae_chung@cable.comcast.com
URI:   http://www.comcast.com

Alex Kasyanov
Comcast Cable Communications
One Comcast Center
1701 John F. Kennedy Boulevard
Philadelphia, PA  19103
US
EMail: alexander_kasyanov@cable.comcast.com
URI:   http://www.comcast.com

Jason Livingood
Comcast Cable Communications
One Comcast Center
1701 John F. Kennedy Boulevard
Philadelphia, PA  19103
US
EMail: jason_livingood@cable.comcast.com
URI:   http://www.comcast.com

Nirmal Mody
Comcast Cable Communications
One Comcast Center
1701 John F. Kennedy Boulevard
Philadelphia, PA  19103
US
EMail: nirmal_mody@cable.comcast.com
URI:   http://www.comcast.com

Brian Van Lieu
Unaffiliated
Bethlehem, PA  18018
US
EMail: brian@vanlieu.net