

Prepared for:

## Federal Communications Commission

CMS Alliance to Modernize Healthcare  
Federally Funded Research and Development Center

Contract No. HHSM-5000-2012-000081

Task Order No. FCC15D0002

## ACE Direct Platform Release Documentation

Version 2.1

May 24, 2018

The views, opinions, and/or findings contained in this report are those of The MITRE Corporation and should not be construed as official government position, policy, or decision unless so designated by other documentation.

Approved for Public Release; Distribution Unlimited. Case Number 18-1719.

© 2018, The MITRE Corporation. All Rights Reserved.

## Record of Changes

Version	Date	Author / Owner	Description of Change
1.0	November 4, 2016	CAMH	Version 1.0 for release to Sponsor
1.1	February 17, 2017	CAMH	Version 1.1 for release to Sponsor
2.0	November 1, 2017	CAMH	Version 2.0 for release to Sponsor
2.1	May 24, 2018	CAMH	Version 2.1 for release to Sponsor

## About the CMS Alliance to Modernize Healthcare

The Centers for Medicare & Medicaid Services (CMS) sponsors the CMS Alliance to Modernize Healthcare (CAMH), the first Federally Funded Research and Development Center (FFRDC) dedicated to strengthening our nation's healthcare system.

The CAMH FFRDC enables CMS, the Department of Health and Human Services (HHS), and other government entities to access unbiased research, advice, guidance, and analysis to solve complex business, policy, technology, and operational challenges in health mission areas. The FFRDC objectively analyzes long-term health system problems, addresses complex technical questions, and generates creative and cost-effective solutions in strategic areas such as quality of care, new payment models, and business transformation.

Formally established under Federal Acquisition Regulation (FAR) Part 35.017, FFRDCs meet special, long-term research and development needs integral to the mission of the sponsoring agency—work that existing in-house or commercial contractor resources cannot fulfill as effectively. FFRDCs operate in the public interest, free from conflicts of interest, and are managed and/or administered by not-for-profit organizations, universities, or industrial firms as separate operating units.

The CAMH FFRDC applies a combination of large-scale enterprise systems engineering and specialized health subject matter expertise to achieve the strategic objectives of CMS, HHS, and other government organizations charged with health-related missions. As a trusted, not-for-profit adviser, the CAMH FFRDC has access, beyond what is allowed in normal contractual relationships, to government and supplier data, including sensitive and proprietary data, and to employees and government facilities and equipment that support health missions.

CMS conducted a competitive acquisition in 2012 and awarded the CAMH FFRDC contract to The MITRE Corporation (MITRE). MITRE operates the CAMH FFRDC in partnership with CMS and HHS, and maintains a collaborative alliance of partners from nonprofits, academia, and industry. This alliance provides specialized expertise, health capabilities, and innovative solutions to transform delivery of the nation's healthcare services. Government organizations and other entities have ready access to this network of partners.

The FFRDC is open to all CMS and HHS Operating Divisions and Staff Divisions. In addition, government entities outside of CMS and HHS can use the FFRDC with permission of CMS, CAMH's primary sponsor.

## Executive Summary

The Federal Communications Commission (FCC) Telecommunications Relay Service (TRS) Center of Expertise (COE) Project promotes the Commission's goal to foster innovations that advance functionally equivalent telecommunications. Toward that end, the project ensures that the Telecommunications Relay Service employs improved technology for persons who are deaf, hard of hearing, deaf-blind, and/or have speech disabilities. The FCC has embraced a research-based approach to achieve this goal by engaging the Centers for Medicare & Medicaid Services (CMS) Alliance to Modernize Healthcare (CAMH) Federally Funded Research and Development Center (FFRDC), operated by The MITRE Corporation (MITRE), to conduct independent engineering assessments that promote and demonstrate TRS's functional equivalence.

CAMH is independently assessing voice telephone services, video access services, and Internet Protocol (IP)-based captioning technology; improvements to TRS efficiency; solutions for direct communication between people with communication disabilities and other telephone users; and the effectiveness, efficiency, and consumer response to current and future approaches for delivering TRS.

At the FCC's request, CAMH developed a Direct Video Calling (DVC) Auto-Routing Proof of Concept (POC) in support of the FCC's Accessible Communications for Everyone (ACE)<sup>1</sup> program. This DVC auto-routing platform enables direct calling from deaf and hard-of-hearing individuals to an American Sign Language (ASL)-trained agent within the organization's call center. The agent handles the call using a video-capable phone with real-time video connection. To demonstrate the capabilities of DVC, the FCC and CAMH have further advanced the original auto-routing POC into a call center platform for two to twenty customer service representatives. This new DVC platform is called ACE Direct.

Table ES-1 describes the new features released in this version of ACE Direct. Subsection 2.4 provides a complete history of ACE Direct releases and their associated features.

---

<sup>1</sup> <https://www.fcc.gov/ace>

Table ES-1. New ACE Direct Features

Version	Release Date	New Feature or Capability
2.1	May 24, 2018	<ul style="list-style-type: none"> <li>Added the capability for a Manager to modify the contact center's Hours of Operations</li> </ul>
		<ul style="list-style-type: none"> <li>Added a function for a Manager to close the contact center in case of emergency</li> </ul>
		<ul style="list-style-type: none"> <li>Integration with Zendesk using CDC Software. This will be a permanent feature and is configurable during installation</li> </ul>
		<ul style="list-style-type: none"> <li>Developed ACE Direct skinny modes for both the Agent and Consumer screens when a separate CRM system is in use</li> </ul>
		<ul style="list-style-type: none"> <li>Outbound calling can be conducted from the Agent portal</li> </ul>
		<ul style="list-style-type: none"> <li>UI Enhancements: <ul style="list-style-type: none"> <li>Present the agent's name to Consumer during a call</li> <li>Re-styled the back button on the Consumer Portal</li> <li>Created a new dialog to inform Consumers that they are in queue</li> <li>Clear/enable/disable chat using context (both Agent and Consumers)</li> <li>Created an Agent option to Return to Away / Return to Ready after a call</li> <li>Enhanced the Management Portal UI, including the Agent PIE chart, etc.</li> <li>Incorporated resizable/movable/profile-able Agent forms</li> </ul> </li> </ul>
		<ul style="list-style-type: none"> <li>Improved Installation and Operations <ul style="list-style-type: none"> <li>Created a global configuration file to simplify the installation process</li> <li>Updated installation procedures for global configuration</li> <li>Established a consolidated database for both ACE Direct and Asterisk</li> <li>Created an ACE Direct sample DB (script) for an initial installation</li> <li>Redesigned ACE Direct/NGINX/OpenAM routing for simplicity and HSTS directive</li> </ul> </li> </ul>

Implementing the Direct Video Calling platform provides critical benefits toward achieving functionally equivalent telecommunications:

- **Improved Communications** – DVC improves privacy and decreases misrepresentation, which improves efficiency, effectiveness, and productivity.
- **Career Opportunities** – Employing native ASL users to handle customer service video calls expands hiring opportunities. Executive Order 13548 (July 2010) directed federal agencies to increase employment opportunities for people with disabilities.
- **Simple Implementation** – The technology to implement a DVC system is readily obtainable, affordable, and easy to set up.
- **Secure Communications** – With proper configuration, agencies can use high-speed broadband and their own internal networks without compromising security or contending with barriers created by firewalls.

- **Maintain ADA Compliance** – DVC ensures compliance with the Americans with Disabilities Act mandates.
- **Cost Savings** – Replacing three-way interpreted calls with two-way direct communication saves money by minimizing the need for repeat calls due to miscommunication and/or misunderstanding.

As part of this effort, CAMH developed and documented requirements and features, including user stories and associated use cases. CAMH also configured, tested, and integrated provider endpoint video devices with the ACE Direct platform. Detailed configuration and source code files are available for download and reproduction to improve solutions to support the community. The public can download or clone these files at <https://github.com/FCC/ACEDirect>.

# Table of Contents

<b>1. Introduction .....</b>	<b>1</b>
1.1 Background .....	1
1.2 Purpose and Scope .....	2
<b>2. Overview of Direct Video Calling and ACE Direct .....</b>	<b>3</b>
2.1 DVC Is an Alternative to Traditional Relay Calls .....	3
2.2 Open Source Development to Promote Community Involvement.....	3
2.3 Conceptual System Overview .....	3
2.4 ACE Direct Components and Technology Features .....	6
2.5 Highlighted User Stories .....	11
2.6 Agent Desktop.....	12
2.6.1 Logging into ACE Direct.....	12
2.6.2 Side Panels.....	13
2.6.3 Video and Real-Time Text Communications .....	16
2.6.4 Agent Desktop Portal Header .....	18
2.6.5 Video Relay Service and Ticket Information .....	18
2.7 Kuando BusyLight™ Visual Ring Indicator and Agent Status .....	20
2.7.1 Agent Status.....	20
2.7.2 Kuando BusyLight™ Light Configuration.....	20
2.7.3 Lightserver .....	22
2.8 Consumer Help Center .....	23
2.8.1 Submit a Complaint .....	23
2.8.2 Use Real-Time Text Chat .....	25
2.8.3 Leave a Videomail .....	26
2.9 Management Portal .....	27
2.9.1 Management Dashboard .....	27
2.9.2 CDR Dashboard .....	30
2.9.3 Videomail Dashboard .....	32
2.9.4 Hours of Operation .....	32
2.10 Identity and Access Management .....	33
2.10.1 Login Screen .....	34
2.10.2 User Dashboard.....	37
2.11 NGINX .....	41
2.12 Redis.....	42
<b>3. Installation and Configuration .....</b>	<b>43</b>
3.1 SSL/TLS Certificate .....	43
3.2 Asterisk Installation and Configuration Script.....	44
3.2.1 How It Works.....	44
3.2.2 Web Secure Sockets.....	47
3.2.3 Sample Configuration Files .....	47
3.3 Node.js.....	49
3.3.1 Node.js Components Installation.....	49

3.4	Management Portal .....	49
3.4.1	Log Files .....	49
3.4.2	Management Dashboard .....	50
3.4.3	Call Detail Record Dashboard .....	54
3.5	Agent / Agent Database (Provider) .....	58
3.5.1	MySQL Database Server Configuration.....	58
3.6	Video Relay Service User Database (Provider) .....	58
3.6.1	MySQL Database Server Configuration.....	58
3.7	STUN Server .....	59
3.7.1	Installation .....	59
3.7.2	System Startup .....	60
3.8	iTRS ENUM Database .....	62
3.9	StrongSWAN for Secure Socket Layer Tunnel .....	62
3.9.1	Installation .....	63
3.9.2	AWS-Specific Configuration.....	66
3.9.3	Troubleshooting .....	66
3.10	Commercial Customer Relationship Management .....	71
3.11	FenDesk Customer Relationship Management .....	72
3.12	Enterprise Service Bus .....	72
3.12.1	Background.....	72
3.12.2	Installation Overview.....	73
3.12.3	Editing blueprint.xml Application File.....	79
3.12.4	Testing the Broker Application.....	80
<b>Acronyms.....</b>		<b>81</b>
<b>Notice .....</b>		<b>84</b>



## List of Figures

Figure 1. Notional Diagram for ACE Direct Platform .....	4
Figure 2. Screenshot of Agent Desktop Login.....	12
Figure 3. Screenshot of Agent Desktop .....	13
Figure 4. Screenshots of the Agent Statuses .....	14
Figure 5. Dialpad for Outbound Calling .....	14
Figure 6. Screenshot of Agent Status.....	15
Figure 7. Screenshot of Agent Mailbox .....	15
Figure 8. Screenshot of Videomail Playback.....	16
Figure 9. Screenshot of User Chat Box .....	17
Figure 10. Screenshot of Script Box .....	17
Figure 11. Screenshot of Call Duration and Assistance Button.....	18
Figure 12. Screenshot of Agent Profile.....	18
Figure 13. Screenshot of VRS Information .....	19
Figure 14. Screenshot of Ticket Information .....	19
Figure 15. Screenshot of Kuando BusyLight™ Light Configuration Page.....	21
Figure 16. Screenshot of Kuando BusyLight™ Default Color Scheme .....	21
Figure 17. Lightserver GUI.....	22
Figure 18. Screenshot of Consumer Help Center .....	23
Figure 19. Screenshot of Consumer Complaint Form .....	24
Figure 20. Screenshot of Complaint Ticket Form .....	24
Figure 21. Screenshot of Video Chat Window .....	25
Figure 22. Screenshot of Real-Time Text Chat .....	25
Figure 23. Screenshot of Ready to Record .....	26
Figure 24. Screenshot of Videomail Recording in Progress.....	27
Figure 25. Screenshot of Management Dashboard .....	28
Figure 26. Screenshot of Resource Status.....	29
Figure 27. Screenshot of Call Detail Record .....	30
Figure 28. Screenshot of Videomail Dashboard .....	32
Figure 29. Screenshot of Hours of Operation Page .....	33
Figure 30. Screenshot of Login Screen.....	34
Figure 31. Screenshot of Retrieve Your Username .....	35

Figure 32. Screenshot of Reset Your Password.....	36
Figure 33. Screenshot of Register Your Account .....	37
Figure 34. Screenshot of User Dashboard .....	37
Figure 35. Screenshot of Basic Info Tab .....	38
Figure 36. Screenshot of Update Password Tab .....	39
Figure 37. Screenshot of Security Questions Tab.....	39
Figure 38. Screenshot of User Dashboard .....	40
Figure 39. Screenshot of Logout Screen.....	41
Figure 40. Internal and External Paths with NGINX.....	41
Figure 41. NGINX and OpenAM Integration.....	42

## List of Tables

Table 1. ACE Direct Components .....	5
Table 2. ACE Direct Version History.....	7
Table 3. ACE Direct Component Features .....	9
Table 4. Highlighted User Stories for ACE Direct.....	11
Table 5. Agent Status.....	20
Table 6. Lightserver GUI Data Elements .....	22
Table 7. Call Detail Record Column Definition .....	31
Table 8. Initial Installation Requirements for ACE Direct.....	43
Table 9. Example Asterisk Endpoint Extensions.....	45
Table 10. AMI Actions and Descriptions .....	50
Table 11. Asterisk AMI Events and Descriptions .....	51

# 1. Introduction

The Federal Communications Commission (FCC) Telecommunications Relay Service (TRS) Center of Expertise (COE) Project promotes the Commission's goal to foster innovations that advance functionally equivalent telecommunications. Toward that end, the project ensures that the Telecommunications Relay Service employs improved technology for persons who are deaf, hard-of-hearing, deaf-blind, and/or have speech disabilities.

## 1.1 Background

The FCC has embraced a research-based approach to achieve this goal by engaging the Centers for Medicare & Medicaid Services (CMS) Alliance to Modernize Healthcare (CAMH) Federally Funded Research and Development Center (FFRDC), operated by The MITRE Corporation (MITRE), to conduct independent engineering assessments that promote and demonstrate TRS's functional equivalence. As part of the Accessible Communications for Everyone (ACE) program, CAMH independently assesses voice telephone services, video access services, and Internet Protocol (IP)-based captioning technology; improvements to TRS efficiency; solutions for direct communication between people with communication disabilities and other telephone users; and the effectiveness, efficiency, and Consumer response to current and future approaches for delivering TRS.

In continuing pursuit of the Commission's goal to advance functionally equivalent telecommunications, CAMH developed ACE Direct, an open source call center platform that supports Direct Video Calling (DVC) for two to twenty Agents. Implementing ACE Direct in a corporate production environment requires customization to ensure adherence to corporate practices and policies related to security, system configurations, cloud services, and availability.

The FCC encourages government agencies and private businesses to make DVC part of their call center strategy because it offers significant gains for providing functionally equivalent telecommunications, including:

- **Improved Communications** – DVC improves privacy and decreases misrepresentation, which enhances efficiency, effectiveness, and productivity.
- **Career Opportunities** – Employing native American Sign Language (ASL) consumers to handle customer service video calls expands hiring opportunities. Executive Order 13548 (July 2010) directed federal agencies to increase employment opportunities for people with disabilities.
- **Simple Implementation** – The technology to implement a DVC system is readily obtainable, affordable, and easy to set up.
- **Secure Communications** – With proper configuration, agencies can use high-speed broadband and their own internal networks without compromising security or contending with barriers created by firewalls.
- **Maintain ADA Compliance** – DVC ensures compliance with the Americans with Disabilities Act (ADA) mandates.

- **Cost Savings** – Replacing three-way interpreted calls with two-way direct communication saves money by minimizing the need for repeat calls due to miscommunication and/or misunderstanding.

CAMH developed and documented ACE Direct requirements and features, including consumer stories and associated use cases. CAMH also configured, tested, and integrated provider endpoint video devices using the ACE Direct platform.

## 1.2 Purpose and Scope

This document presents an overview of the ACE Direct architecture, user stories, and describes how to integrate DVC within an agency's current call center workflow to provide an independent, on-demand service.

In addition to this release documentation, detailed configuration and source code files are available to the public at <https://github.com/FCC/ACEDirect> for download and reproduction of the platform to support and promote future platform enhancements for the deaf, hard-of-hearing, deaf-blind, and speech-disabled community.

## 2. Overview of Direct Video Calling and ACE Direct

Deaf, hard-of-hearing, deaf-blind, or speech-disabled people use TRS to communicate with hearing people over the phone. Since the early 2000s, video relay service (VRS) calls have been the primary way that ASL consumers access telecommunications. VRS involves the use of third-party communication assistants (CA) as sign language interpreters to place telephone calls. The interpreter translates between ASL and spoken English for the non-signing party. People who communicate in ASL use VRS to place telephone calls to customer assistance divisions of government agencies and businesses in the United States every day, but there are other solutions.

### 2.1 DVC Is an Alternative to Traditional Relay Calls

The FCC's sponsorship of the ACE program includes creating a direct video calling platform. The ASL Consumer Support Line<sup>2</sup>—the first of its kind in the federal government—allows ASL users to make video calls directly to an agent fluent in ASL. English is not the first language of many TRS deaf, hard-of-hearing, deaf-blind, and speech-disabled TRS consumers. One-to-one communication in ASL is most often preferred.

When comparing calls made to the FCC ASL Consumer Support Line with calls placed through VRS, the FCC found that VRS calls were handled on average 33 percent faster and the number of deaf consumers increased approximately threefold. Most impressive is that the FCC achieved these results without adding staff to handle the increased call volume.

### 2.2 Open Source Development to Promote Community Involvement

ACE Direct is open source technology that offers one option for implementing DVC. Open source promotes universal access via a free license to a product's design/blueprint and universal redistribution of that design/blueprint, including subsequent improvements to it. The open source model employs a decentralized model of production. A main principle of open source software development is peer production: products such as source code, "blueprints," and documentation are available to the public at no cost.

The FCC encourages government agencies, educational institutions, and others seeking to enhance the lives of citizens who are deaf, hard of hearing, deaf-blind, and/or have speech disabilities to adopt and improve on the existing code base to provide additional features, improve the workflow, and introduce new technologies to the open source ACE Direct platform.

### 2.3 Conceptual System Overview

CAMH developed the open source-based ACE Direct platform for implementation in the Amazon Web Services (AWS) cloud environment. Figure 1 presents a notional view of the architecture of the ACE Direct components from a configuration and programming standpoint.

---

<sup>2</sup> Available at <https://www.fcc.gov/document/fcc-adds-american-sign-language-consumer-support-line-videophone>.

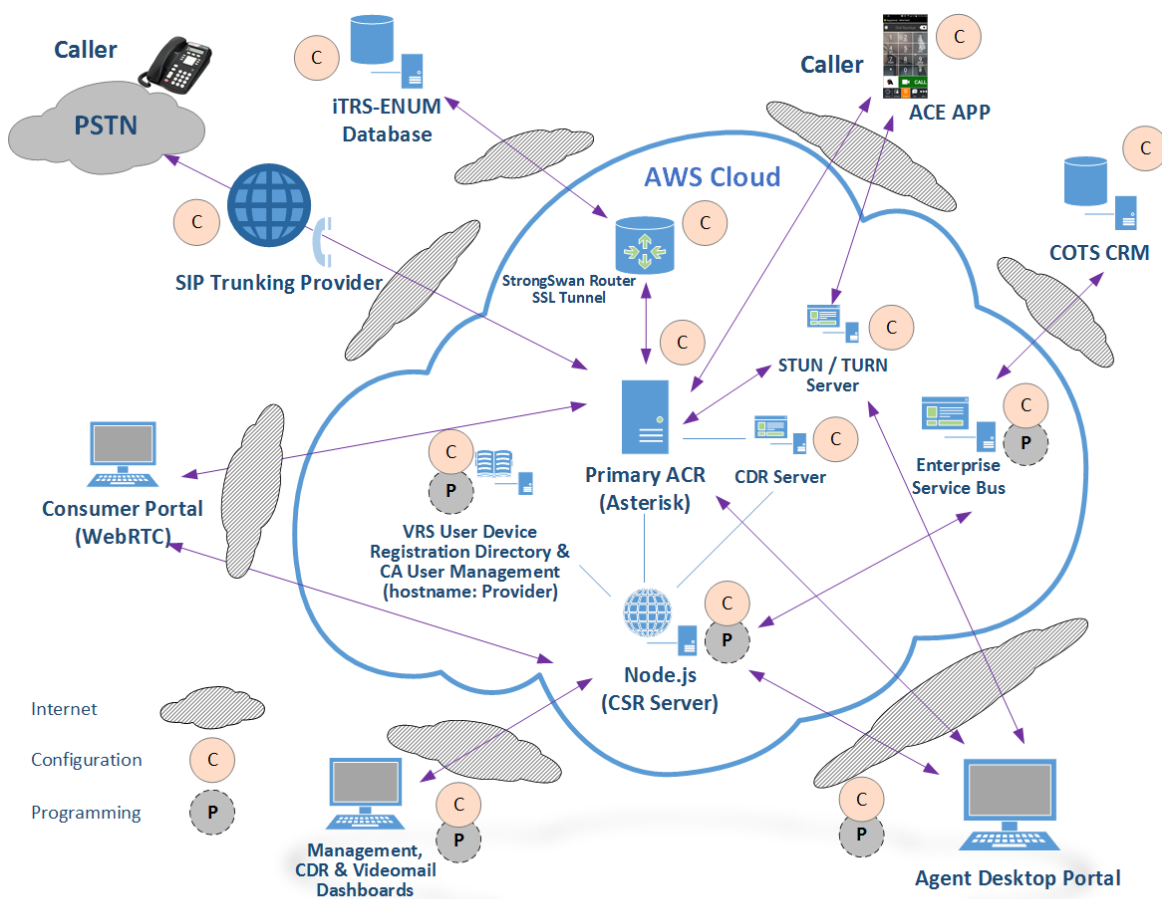


Figure 1. Notional Diagram for ACE Direct Platform

As Figure 1 shows, some ACE Direct components require only configurations (noted as “C”) and other components require both configuration and programming (noted as “P”). Table 1 presents an overview of these components. Section 3 provides detailed information about installation and configuration.

Table 1. ACE Direct Components

Component	Description	Additional Details
Asterisk Open Source PBX (Private Branch Exchange)	The Asterisk Open Source PBX supports direct video communication via both Public Switched Telephone Network (PSTN) and video calls.	Subsection 3.2
Node.js	Node.js is an open source platform that can be used to develop applications and servers. For ACE Direct, the Node.js server contains several services running on ports to support the Agent Desktop Portal and other management-related portals, including the management dashboard and call detail record (CDR) dashboard. The Node.js server supports the Real-Time Text (RTT) between the Agent Desktop Portal and the Consumer Portal. It also provides services for VRS lookup to verify that the phone number is a valid number in the VRS database.	Subsection 3.3
Screenshot of Consumer Help Center	The Consumer Portal combines form submission with real-time audio, video, and text communication to an Agent.	Subsection 2.7
Management Portal	The Management dashboard provides Key Performance Indicators (KPI) that the call center manager can monitor in real time. The CDR dashboard provides the view and export functions of the Asterisk CDRs stored in its MySQL database.	Subsection 2.8
Agent Desktop	The Agent desktop provides a user interface to the Agent—the Customer Service Representative (CSR)—for login and conducting DVC services to the ACE Direct Consumers.	Subsection 2.5
STUN Server	STUN (formerly Simple Traversal of UDP through NAT RFC 3489) is reflexive and identifies if the endpoint is behind a Network Address Translation (NAT) or firewall and determines the public IP address. This helps STUN establish a peer-to-peer connection.	Subsection 3.7
iTRS-ENUM database	The iTRS (Internet Telecommunications Relay Service) database maps 10-digit U.S. telephone numbers to IP addresses using the industry-standard ENUM (E.164 Number to URI Mapping) protocol. VRS providers assign these 10-digit telephone numbers to their customers.	Subsection 3.8
StrongSWAN Router for Secure Socket Layer (SSL) Tunnel	To support the iTRS-ENUM database lookup, a VPN tunnel is established to Neustar from a StrongSwan (open source network operating system) router instance running in AWS. The router instance has a loopback interface with an Elastic IP (EIP) on it that is the encryption domain for the tunnel.	Subsection 3.9

Component	Description	Additional Details
Commercial Off-the-Shelf (COTS) Customer Relationship Management (CRM)	To demonstrate integration with a CRM service, ACE Direct connects to the Zendesk RESTful application programming interface (API) via the Enterprise Service Bus. ACE Direct sends Java Script Object Notation (JSON)-based messages to the RESTful Zendesk API to manage and query customer records.	Subsection 3.10
Enterprise Service Bus (ESB)	The ESB provides a generic method to update legacy database systems as well as the diverse number of databases and unstructured data repositories on the market and in use today. ACE Direct ESB integrates with a COTS CRM service (e.g., Zendesk) as a ticketing system for the Agent to document service cases.	Subsection 3.12
Identity and Access Management	ACE Direct uses the OpenAM and OpenIDM components from ForgeRock to: <ul style="list-style-type: none"> <li>• Provide secure access to the Agent and Management portals</li> <li>• Allow self-help features, such as registration and lost password</li> <li>• Allow a Manager to set the hours an Agent can be active in the system</li> </ul>	Subsection 2.9
External Visual Ring Indicator and Agent Status	The Kuando BusyLight™ is used as an external visual ring indicator and Agent status instrument. ACE Direct supports both of its models, Alpha and Omega.	Subsection 2.6
Reverse Proxy, Load-balancer and HTTP Cache	NGINX is used as a reverse proxy to only expose HTTPS/port 443 and hide internal port number and internal script names to prevent spoofing and hacking by external entities.	Subsection 2.10
State Management and Key Information Storage	Redis is an in-memory key-value data store, used as a database to store data previously stored in memory to manage state.	Subsection 2.12

## 2.4 ACE Direct Components and Technology Features

The following subsections recount ACE Direct's history and the features found in the ACE Direct components. Table 2 describes the history of ACE Direct, while Table 4 describes the features of the ACE Direct components.



Table 2. ACE Direct Version History

Version	Release Date	Enhancements / Features Introduced
1.0	November 4, 2016	<ul style="list-style-type: none"> <li>• The first open source, omnichannel auto call-routing contact center platform designed for 2 to 20 Agents</li> <li>• Browser-based interface allowing for remote use by Agents and Managers</li> <li>• A Management Portal for contact center statistics such as calls waiting, calls abandoned, and average hold time</li> <li>• Video, audio, and Real-Time Text (RTT) communications</li> <li>• WebRTC technology to facilitate browser-to-browser video communication</li> <li>• Enterprise Service Bus (ESB) for enterprise data integration</li> <li>• Support for multiple queues: Complaints and General Questions</li> </ul>
1.1	February 17, 2017	<ul style="list-style-type: none"> <li>• Call transfer functionality from one Agent to another</li> <li>• Increased character limit in WebRTC RTT implementation</li> <li>• Acceptance of inbound PSTN calls</li> <li>• Data transmission is encrypted using TLS and HTTPS throughout the platform</li> <li>• Segmentation of application servers to increase system scalability</li> <li>• Code modifications to improve reliability and scalability</li> </ul>
2.0	November 1, 2017	<ul style="list-style-type: none"> <li>• Introduced a “single pane of glass” for the Agent Portal, thus all communications occur through the browser</li> <li>• Enhanced system security through: <ul style="list-style-type: none"> <li>– An identity and access management solution, OpenAM, to manage system access</li> <li>– URL masking using NGINX to prevent external cyberattacks</li> </ul> </li> <li>• Added an external visual ring indicator (Kuando Busylight™) to inform the Agent of an incoming call and others of an Agent’s status</li> <li>• Introduced videomail recording and retrieval</li> <li>• Usability enhancements to the Agent, Management, and Consumer portals</li> <li>• Simplified the installation process for quicker installations</li> </ul>

Version	Release Date	Enhancements / Features Introduced
2.1	May 2018	<ul style="list-style-type: none"><li>• Added ability for a Manager to modify the contact center's hours of operations</li><li>• Added function for a Manager to close the contact center in case of emergency</li><li>• Integration with Zendesk using CDC Software. This will be a permanent feature and is configurable during installation.</li><li>• Developed ACE Direct skinny modes for both the Agent and Consumer screens when a separate CRM system is in use</li><li>• Outbound calling can be conducted from the Agent Portal</li><li>• UI Enhancements:<ul style="list-style-type: none"><li>– Present the Agent's name to Consumer during a call</li><li>– Re-style the back button on the Consumer Portal</li><li>– New dialog to inform Consumers that they are in queue</li><li>– Clear/enable/disable chat using context (both Agent and Consumers)</li><li>– Agent option to Return to Away / Return to Ready after a call</li><li>– Enhancements to the Management Portal UI, including the Agent PIE chart, etc.</li><li>– Incorporated resizable/movable/profile-able Agent forms</li></ul></li><li>• Improved Installation and Operations<ul style="list-style-type: none"><li>– Global configuration file to simplify the installation process</li><li>– Updated installation procedure for global configuration</li><li>– Consolidated database for both ACE Direct and Asterisk</li><li>– Created an ACE Direct sample DB (script) for an initial installation</li><li>– Redesigned ACE Direct/NGINX/OpenAM routing for simplicity and HSTS directive</li></ul></li></ul>

Table 3. ACE Direct Component Features

Component	Component Features
Agent Portal – The Agent interface to the Consumer	<ul style="list-style-type: none"> <li>• Browser-based to allow for remote access</li> <li>• Data transmission is encrypted using TLS and HTTPS</li> <li>• All video and RTT communications conducted through a single browser</li> <li>• Video display can be set to full screen on command. This is particularly useful when video communication is less than ideal.</li> <li>• Outbound calling using an integrated dialer</li> <li>• Videomails can be viewed and the display sorted on any data fields listed</li> <li>• Videomail callbacks can be made with the click of the mouse</li> <li>• Number of unread videomails displayed</li> <li>• Get Help feature to contact a Manager</li> <li>• External visual ring indicator to notify the Agent of an incoming call and others when the Agent is in a call</li> <li>• Support for multiple queues: Complaints and General Questions</li> <li>• Displays the number of calls waiting in the queue</li> <li>• Visibility into the status of other Agents. Useful if Agents are geographically disbursed.</li> <li>• Duration of the call provided to the Agent while in the call</li> <li>• CRM ticket information and scripts can be integrated into ACE Direct and displayed in the Agent Portal</li> <li>• Sections of the interface can be resized and moved based on Agent preferences</li> <li>• Skinny mode hides CRM forms</li> <li>• Disable chat during calls from provider devices because these devices do not currently provide a chat feature</li> </ul>
Consumer Portal – The Consumer interface to the Agent	<ul style="list-style-type: none"> <li>• Browser-based to allow remote access</li> <li>• Data transmission is encrypted using TLS and HTTPS</li> <li>• All video and RTT communications conducted through a single browser</li> <li>• Video display can be set to full screen on command. This is particularly useful when video communication is less than ideal.</li> <li>• Agent's name displayed during video and RTT calls to enhance interaction</li> <li>• Displays the Consumer's position in the queue</li> <li>• Displays a dialog when the call center is after hours</li> <li>• May be a standalone web page or integrated with an existing portal</li> <li>• Skinny mode bypasses CRM ticket input</li> <li>• Videomail capability</li> <li>• Configurable redirect to a specific URL</li> </ul>

Component	Component Features
<p>Management Dashboard – Provides contact center statistics and KPIs</p>	<ul style="list-style-type: none"> <li>• Browser-based to allow for remote access</li> <li>• Data transmission is encrypted using TLS and HTTPS</li> <li>• Support for multiple queues to direct your customers to the proper Agent</li> <li>• There are two queue templates in ACE Direct out of the box: ComplaintsQueue and GeneralQuestionQueue. The following KPIs are a summary over both queues combined. <ul style="list-style-type: none"> <li>– Calls Waiting – Number of calls waiting in all queues.</li> <li>– Calls Handled – Number of calls completed in all queues.</li> <li>– Average Hold Time (minutes:seconds) – Average call holding time in all queues.</li> <li>– Calls Abandoned – Number of calls not answered in all queues.</li> </ul> </li> <li>• <b>Queue-related KPIs</b> – The following KPIs are displayed per queue template: (Logged In – Number of Agents currently logged into the system. <ul style="list-style-type: none"> <li>– Available Agents – Number of Agents currently in a ready state.</li> <li>– Current Calls – Number of calls currently in progress.</li> <li>– Total Calls – Total number of calls made.</li> <li>– Calls Handled – Total number of calls answered by an Agent.</li> <li>– Calls Abandoned – Total number of calls abandoned.</li> <li>– Talk Time – Average talk time (minutes:seconds).</li> <li>– Hold Time – Average hold time (minutes:seconds).</li> <li>– Longest Hold Time – The longest hold (minutes:seconds).</li> </ul> </li> <li>• <b>Agent-related KPIs</b> – The following KPIs are displayed per Agent. The Agent name, extension, and registered queues are displayed along with the KPI: <ul style="list-style-type: none"> <li>– Agent name – Name of the Agent.</li> <li>– Registered extension – Extension assigned to the Agent.</li> <li>– Registered queues – Asterisk queues assigned to the Agent. All queue names are displayed if an Agent is assigned to more than one queue.</li> <li>– Calls Completed – Number of calls handled (answered and completed) by the Agent.</li> <li>– Average Call Time – Talk Time divided by number of calls.</li> <li>– Talk Time – The cumulative time the Agent has spent on calls.</li> <li>– Status – Logged Off, Ready, Away, or In-Call.</li> </ul> </li> </ul>
<p>Call Detail Record dashboard – Provides a means of auditing call activity, tracking a call Agent's activity, and creating a report of both incoming and outgoing calls</p>	<ul style="list-style-type: none"> <li>• Provides a method to view, sort, search, and export the Asterisk call detail records (CDR) stored in the database for additional reporting by your business intelligence or report writing tool.</li> </ul>
<p>Kuando BusyLight™ Configuration provides a graphical user interface to customize the light display of the Kuando BusyLight™</p>	<ul style="list-style-type: none"> <li>• Agent statuses, light colors, and light behaviors (solid/blinking) are customizable to fit your environment</li> <li>• Function to reset to a default configuration</li> <li>• Color/behavior changes are applied to the Agent Portal automatically in real time</li> </ul>

Component	Component Features
Hours of Operation UI – A feature to implement and manage call center hours of operation	<ul style="list-style-type: none"> <li>• This UI allows the Manager to establish the days and hours of operation for the contact center instead of having the Asterisk administrator perform this operation through an Asterisk command line</li> <li>• Ability to force open, force close, or resume normal business operation with the click of the mouse</li> <li>• Lists hours of operation in all U.S. time zones for easy readability</li> <li>• Time zone graphical map</li> <li>• Contact Center open/closed indicator for a quick view of the contact center status</li> </ul>
Videomail Dashboard – A manager view of all videomails in the system	<ul style="list-style-type: none"> <li>• Sort, view, and filter videomails to organize them in your preferences</li> <li>• Pie chart for videomail statuses for easy viewing</li> </ul>

## 2.5 Highlighted User Stories

The FCC and CAMH partnered with federal agencies to derive typical requests for services and call center workflows. CAMH built ACE Direct to encompass the core functions of a traditional hearing-based call center. ACE Direct focuses on the responsibilities of Agents and their Managers. For version 2.0, CAMH added the role of Administrator to provision and deprovision Agents to the system. Table 4 presents a summary of ACE Direct user stories, which demonstrate these functions and capabilities.

Table 4. Highlighted User Stories for ACE Direct

User Story	Description
Direct Video Call to an ASL-fluent Agent	As an ASL user, I want to speak with another ASL user when I contact a call center.
Direct Video Call from an ASL-fluent Agent	As an ASL user, I want to receive a callback from another ASL user when at a call center.
CRM Integration	As an Agent, I want to view, update, and enter new information regarding contact with the Consumer from the corporate CRM system.
Call Script Integration	As an Agent, I want to view corporate call scripts based on the needs of the Consumer.
Call-handling capabilities	As an ACE Direct Agent, I want to perform “Call on Hold” and “Call Transfer” as needed.
Screenshot of Consumer Help Center	(A complaint process illustrates this story.) As a Consumer, I want to file a complaint through a web portal on my website. I also want the option of conversing with the Agent through video and Real-Time Text. Please refer to subsection 2.6 for details.
Videomail	As an Agent, I want to retrieve a videomail left by Consumers.
Management Dashboard	As the ACE Direct Manager/Operator, I want to access near real-time information on the dashboard.

User Story	Description
Call Detail Record	As the ACE Direct Administrator, I want to access the Call Detail Record through a web portal and export CDRs as needed for audit purposes. Please refer to subsection 2.7.2 for details.
Web-based Application	As an Agent, I want the ability to work remotely from the main call center, if necessary.
Multi-CSR Login with Status	As the ACE Direct Agent, I want to log in using the Agent Desktop along with other Agents and I want to change my status between "Ready" and "Away".
Add, Suspend, or Remove an Agent's Access	As the Administrator, I want to add, suspend or remove an Agent's access to ACE Direct.

## 2.6 Agent Desktop

In this section, the Agent Desktop User Guide provides a walkthrough of the Agent Portal, highlighting each of the available functionalities.

At the time of publication, the ACE Direct Agent Portal is compatible with the Chrome browser, which is WebRTC compatible. WebRTC technology allows ACE Direct to present video directly through the browser, eliminating the need for a second monitor and providing a full omnichannel experience for the Agent.

### 2.6.1 Logging into ACE Direct

Upon navigating to the portal host URL, a login screen appears as shown in Figure 2. To access the portal, Agents must enter their username and password.

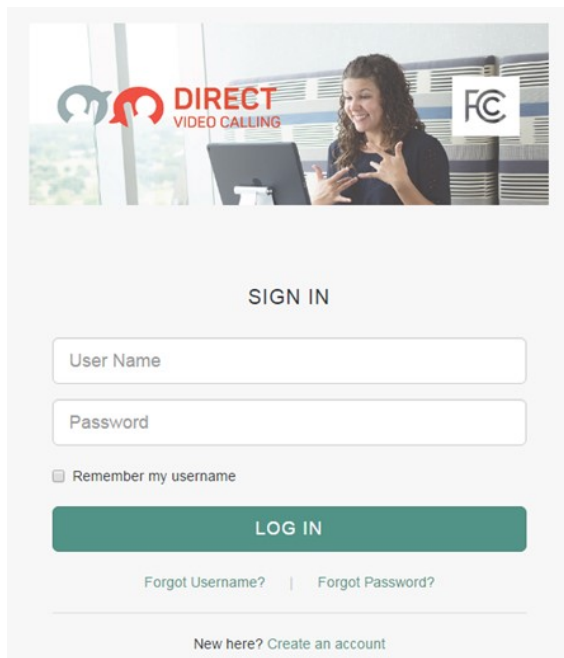


Figure 2. Screenshot of Agent Desktop Login

Figure 3 presents a screenshot of the Agent Desktop, which consists of the following elements:

- Side panels (left and right) to provide navigation and information to the Agent, including a videomail retrieval panel and an outbound calling dial pad
- A user chat area for RTT chats with the VRS Consumer
- A header area that displays call duration information and a help button
- Profile information displaying the Agent's name and picture and the capability to sign out of the system
- VRS Consumer information such as first name, last name, etc.
- Current CRM ticket information provided by the VRS Consumer

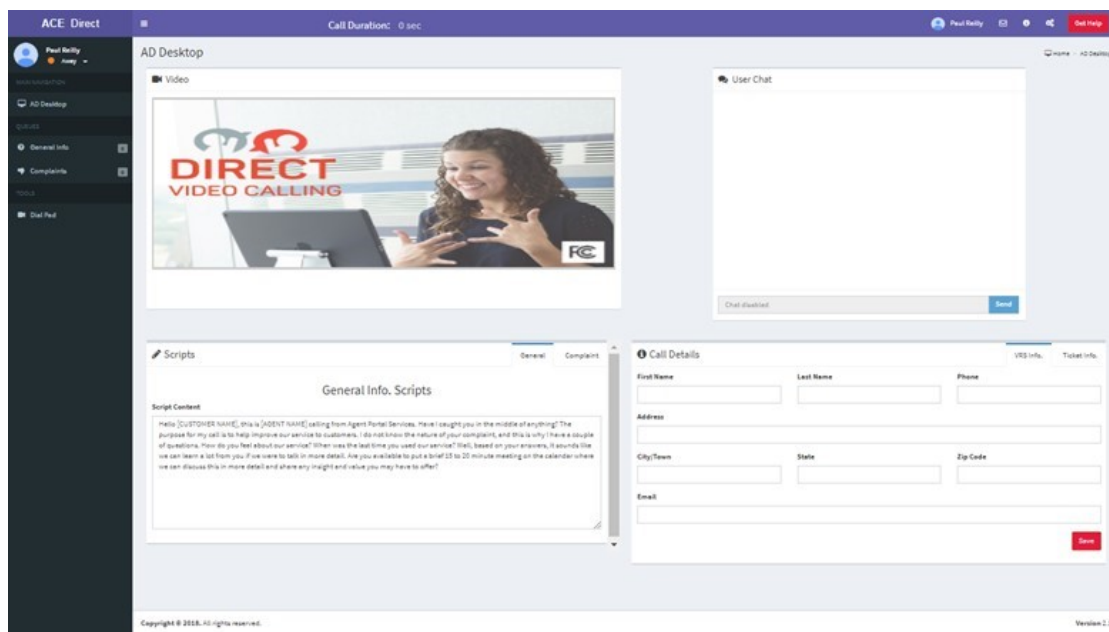


Figure 3. Screenshot of Agent Desktop

## 2.6.2 Side Panels

The ACE Direct Agent Desktop portal has two side panels that provide navigation and information to the Agent.

### 2.6.2.1 Left Side Panel (Main Navigation)

As shown in Figure 4, the left-side panel of the Agent Portal provides both Consumer Status and the Main Navigation. Here Agents can select their status as “Ready” or “Away” via the dropdown status change button. When an Agent first signs into the portal, the status defaults to “Away”. When the Agent is ready to receive calls, the Agent selects the “Ready” status. For an incoming call, an intermediate “Incoming Call” status appears, along with a modal alert dialog that takes the foreground. Once the Agent enters a call, the status changes to “In Call”. After the Agent leaves the call, the Agent is presented with a modal to enter either an “Away” or a “Ready” state. If the Agent chooses “Away”, the Agent can perform any tasks related to the call. The agent can then select the “Ready” state to enter the call queue.

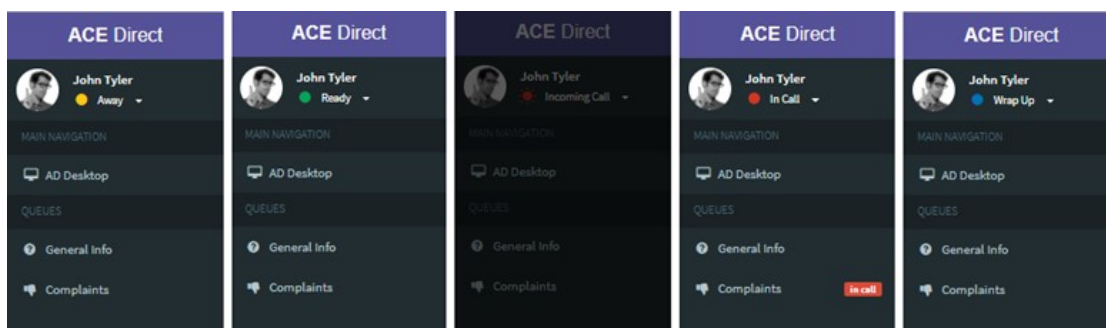


Figure 4. Screenshots of the Agent Statuses

As shown in Figure 5, a new Dial Pad feature at the bottom of the left side panel allows Agents to place outbound calls. Clicking the Dial Pad icon brings up the dialpad for outbound calling.

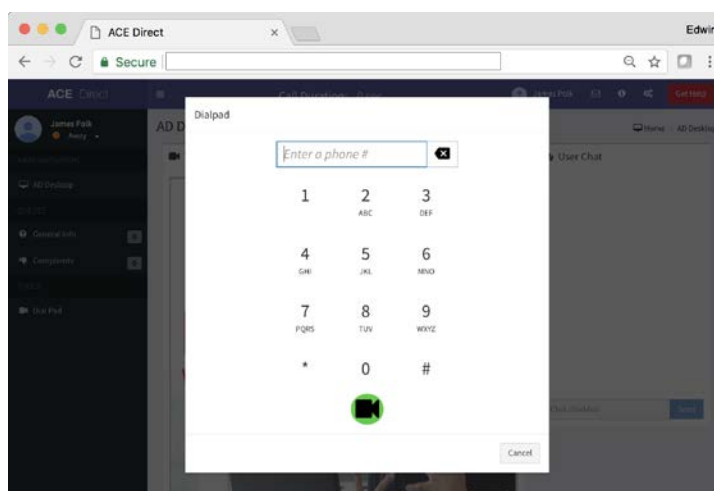
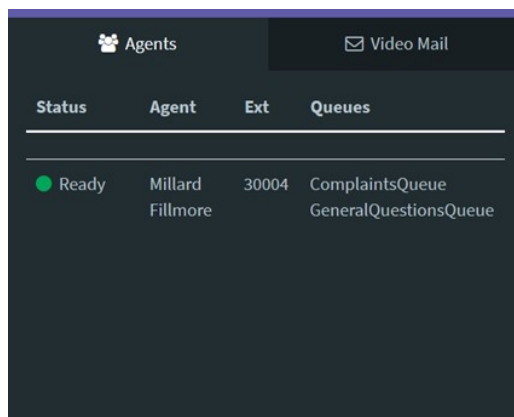


Figure 5. Dialpad for Outbound Calling

### 2.6.2.2 Right Side Panel (Agent Status and Videomail mailbox)

The right-side panel as shown in Figure 6 is accessible by clicking on the gears icon in the top right corner of the portal. The Agent can reach the videomail mailbox directly by clicking on the envelope icon in the top right corner of the portal. This section can be collapsed to give the Agent more space for the main content area. Upon opening the right-side panel, the Agent can access two tabbed content areas (Agents and Video Mail).





Status	Agent	Ext	Queues
Ready	Millard Fillmore	30004	ComplaintsQueue GeneralQuestionsQueue

Figure 6. Screenshot of Agent Status

#### 2.6.2.2.1 Agent Status

The Agent Status section provides the Agent with a list of Agents logged into ACE Direct. The Agent can view information about each Agent listed, such as their status, extension, and queues.

#### 2.6.2.2.2 Videomail

The Videomail tab, as shown in Figure 7, displays a list of videomails received while the Agents were unavailable to take calls. This list provides the Agent with the videophone number, time, date, duration and status of the videomail. The Agent can sort the videomail table by any of the columns in ascending or descending order, and can filter the videomail by status. The status may be “Unread”, “Read”, “In Progress”, or “Closed”. Unread videomails are highlighted in **bold**. An indicator at the top right of the screen provides the Agent with a count of unread videomails. All Agents are presented with the same list of videomails as videomails are not specific to a particular Agent.

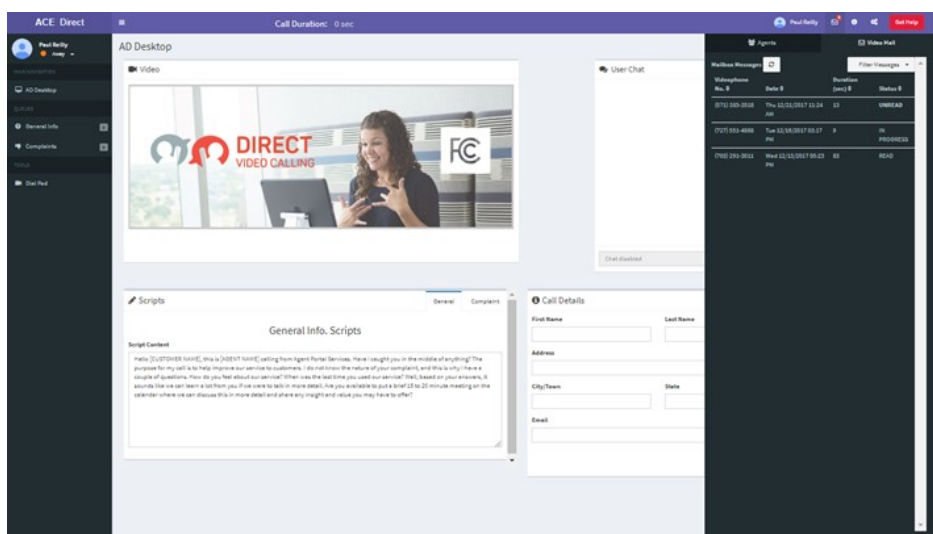


Figure 7. Screenshot of Agent Mailbox

By clicking on a specific videomail, the Agent can view the contents and update the status. Figure 8 displays the playback screen.

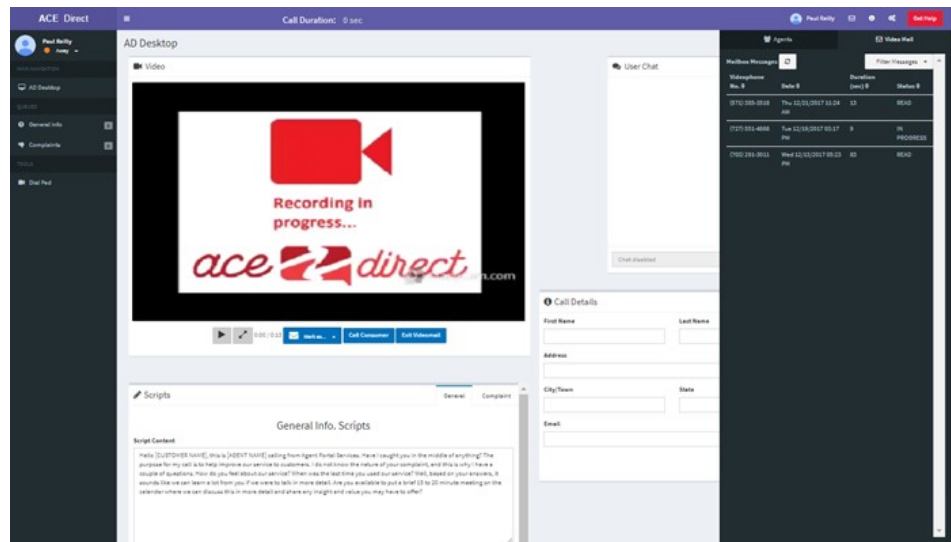


Figure 8. Screenshot of Videomail Playback

The videomail status can be changed to “Unread”, “Read”, “In Progress” or “Closed”. If the Agent deletes the videomail, it is removed from the videomail mailbox but can be reviewed in the Management Portal before being permanently deleted. The Agent can also place a call to the videophone number associated with the videomail.

## 2.6.3 Video and Real-Time Text Communications

The ACE Direct platform supports two methods of communication; video and Real-Time Text. This subsection describes how each method operates in the platform.

### 2.6.3.1 Video Chat

Video Chat communications on the platform occur through the browser using WebRTC technology. The Consumer must be using a WebRTC-compatible browser to enable this functionality if the Consumer is using a computer or smartphone. Video Chat can also be used with a videophone. During a call, the Agent has button options to mute audio, mute video, or view the Consumer’s video in full screen mode.

### 2.6.3.2 Real-Time Text Chat

The User Chat box, as shown in Figure 9, provides the Agent a secondary method of communication with the Consumer. As the Agent types a message to the Consumer in the input field, the Consumer will see the message in real time. The chat history remains visible to the Agent until the Agent closes the ticket. Use of this feature is not available for videophones as of this publication.

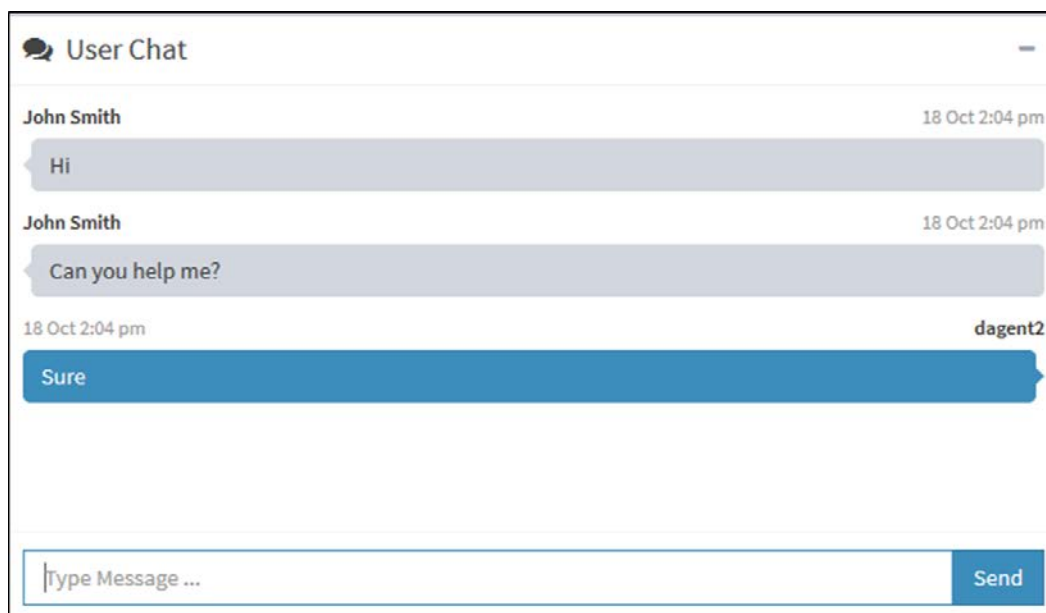


Figure 9. Screenshot of User Chat Box

### 2.6.3.3 Scripts

The Scripts box, as shown in Figure 10, provides the Agent with two sub-tabbed areas where the Agent can select the appropriate script to apply to the conversation. This area is customizable during integration at a site.

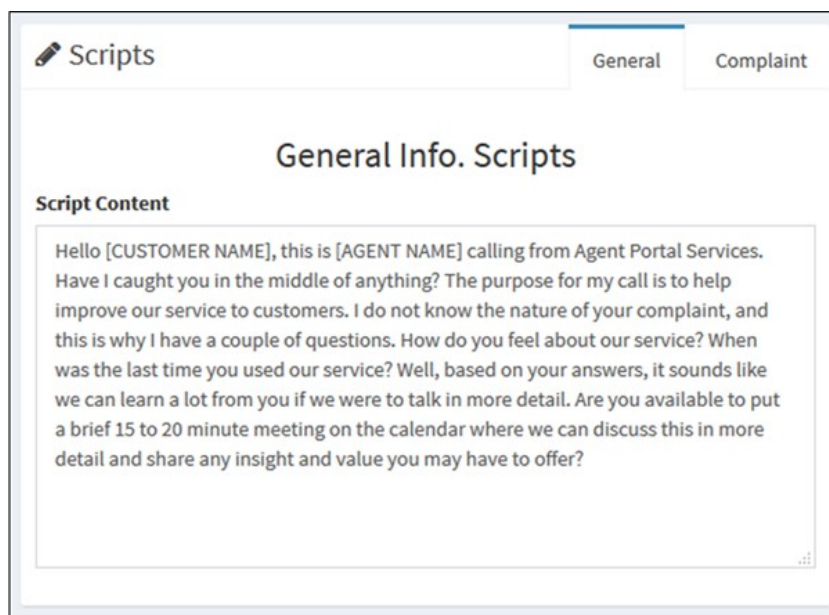


Figure 10. Screenshot of Script Box

## 2.6.4 Agent Desktop Portal Header

The Agent Desktop Portal header provides the Agent with the call duration information, a help button, and profile information about the Agent along with the capability to sign out of the system.

### 2.6.4.1 Call Duration and Get Help Button

The Call Duration shows a running clock of the call length once the Agent accepts the incoming Consumer call. As shown in Figure 11, the Get Help button allows the Agent to request help from a Manager during a call. When the Agent clicks the Get Help button, the Agent's name will change color and begin to flash on the Management Dashboard to indicate the Agent needs help.

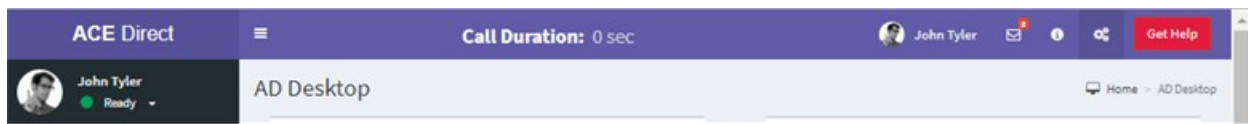


Figure 11. Screenshot of Call Duration and Assistance Button

### 2.6.4.2 Agent Profile

After logging in, the Agent's name and picture will appear in the Agent Profile at the top right corner of the Agent Desktop Portal head as shown in Figure 12. (Currently, all Agents display the same profile picture.) The Agent has the capability to log out of the ACE Direct Portal by clicking the "Sign out" button. If the Agent has changed the layout of the Agent Portal, it can be reverted to the original layout by clicking the "Default Layout" button.

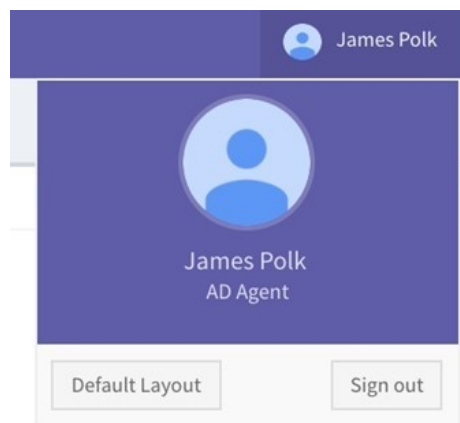
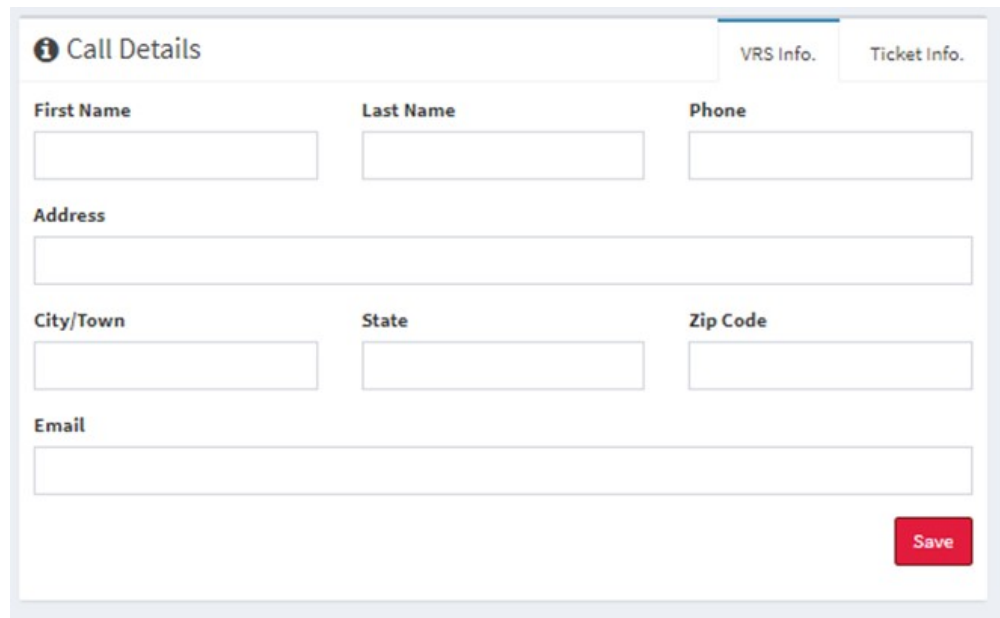


Figure 12. Screenshot of Agent Profile

## 2.6.5 Video Relay Service and Ticket Information

### 2.6.5.1 Video Relay Service Information

The VRS Information section displays the information about the Consumer currently on file in the CRM system. Figure 13 shows that after the call has ended, the Agent must click on the "Save" button in the VRS Information box to return to the queue and receive new calls.

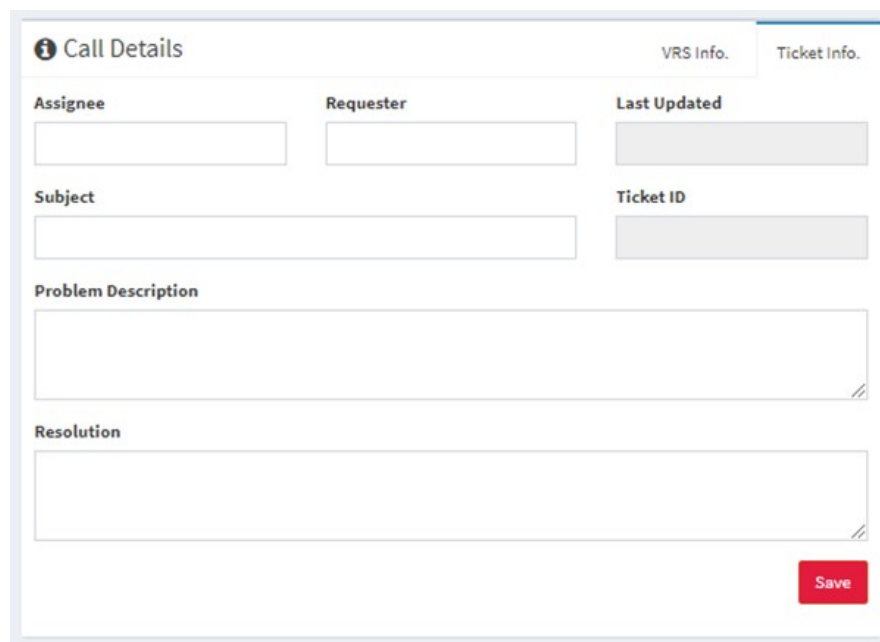


The screenshot shows a web form titled "Call Details" with an information icon. It has two tabs: "VRS Info." (selected) and "Ticket Info.". The form contains several input fields: "First Name", "Last Name", "Phone", "Address", "City/Town", "State", "Zip Code", and "Email". A red "Save" button is located at the bottom right.

Figure 13. Screenshot of VRS Information

### 2.6.5.2 Ticket Information

Figure 14 shows the Ticket Information section that provides the Agent with the Consumer's submitted information.



The screenshot shows the same "Call Details" form, but with the "Ticket Info." tab selected. The form contains input fields for "Assignee", "Requester", "Last Updated", "Subject", "Ticket ID", "Problem Description", and "Resolution". The "Last Updated" and "Ticket ID" fields are currently disabled (grayed out). A red "Save" button is located at the bottom right.

Figure 14. Screenshot of Ticket Information

## 2.7 Kuando BusyLight™ Visual Ring Indicator and Agent Status

ACE Direct incorporates the Kuando BusyLight™ device as an integral part for notifying call center personnel of incoming calls and the status of an Agent (such as “Away”, “Ready”, “In call,” etc.). The Administrator can configure the color of the light and ensure a consistent configuration across all Agents. Subsection 2.6.2 presents an example configuration.

The Kuando BusyLight™ is not included with the ACE Direct platform and must be purchased separately. The Kuando BusyLight™ is available in several different models and from several online vendors.

### 2.7.1 Agent Status

Each Agent has a status based on the Agent’s activity with ACE Direct, as shown in Table 5.

Table 5. Agent Status

Agent Status	Definition
Away	The Agent is not available and no calls will be directed to them.
Ready	The Agent is available to take calls from the queues.
In a Call	The Agent is currently handling a call.
Incoming Call	The Agent is receiving a call, but has not yet answered it.
Wrap Up	The Agent just finished a call, but has not yet hit “Return to Ready” or “Return to Away”. No calls can be directed to the Agent.

The Agent Status, presented by coordinated color and lighting pattern, is communicated via the Kuando BusyLight™ device and displayed in the Agent Portal as depicted in subsection 2.6.2. Figure 15 shows the definitions of different colors available. The configurations can be reset to the default values at any time by clicking “Reset to Default” in Figure 16.

### 2.7.2 Kuando BusyLight™ Light Configuration

Managers can customize the color associated with each possible Agent status through the Light Configuration page in the Management Portal as shown in Figure 15. Soon after a Manager saves the form, the color is updated in real time and appears on the Agent’s Kuando BusyLight™ as well as in the Agent Portal, as shown in Figure 15 and Figure 16.

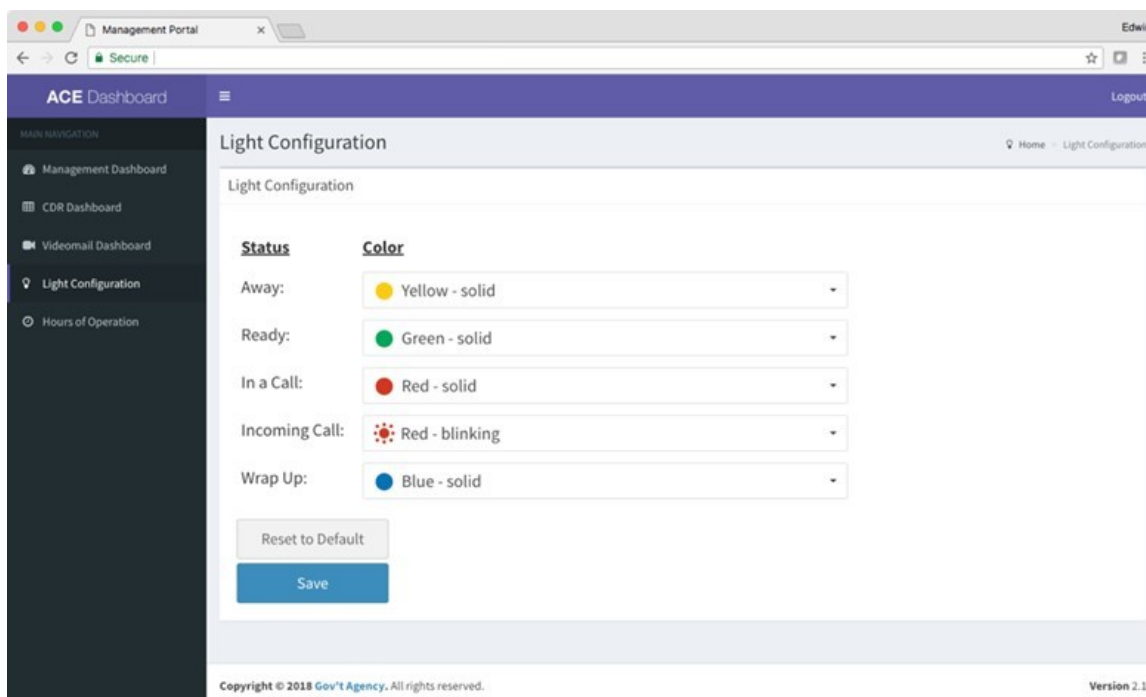


Figure 15. Screenshot of Kuando BusyLight™ Light Configuration Page

Figure 16 shows the default color scheme, which is Section 508 compliant. Using the “Reset to Default” button, the status and color selections will revert to the default settings.

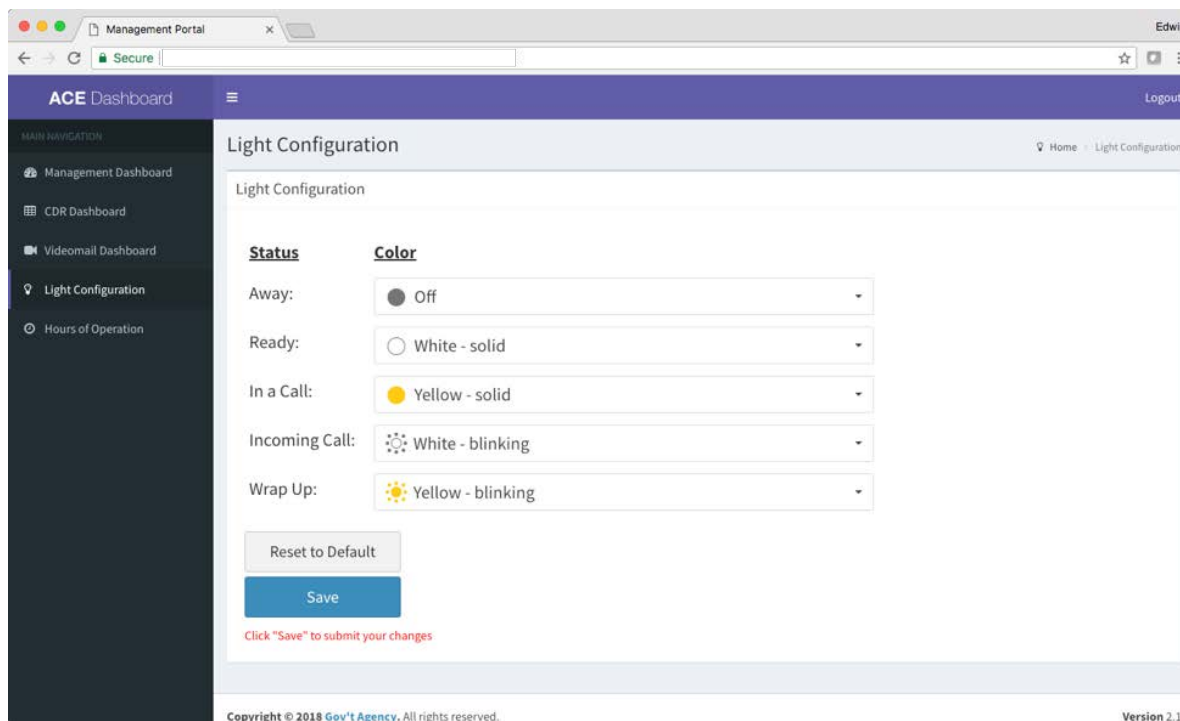


Figure 16. Screenshot of Kuando BusyLight™ Default Color Scheme

### 2.7.3 Lightserver

Lightserver is a standalone Java application that must execute on the desktop computer of the Agent. The Lightserver program is a graphical user interface (GUI) for integrating the Kuando BusyLight™ with the ACE Direct platform. It provides a RESTful interface via localhost only to the ACE Direct Agent Portal. When the Agent status changes, the Agent Portal makes RESTful calls to the Lightserver. A Kuando BusyLight™ device must be connected to a USB port on the same computer before starting the Java application.

The ACE Direct Portal makes the initial connection to Lightserver when the Agent navigates to the ACE Direct Agent Portal. This connection enables all requests from the ACE Direct Agent Portal to Lightserver, as shown in Figure 17.

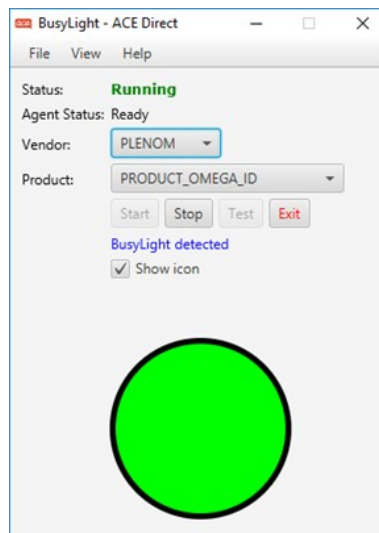


Figure 17. Lightserver GUI

At startup, the Lightserver GUI attempts to detect a connected BusyLight™ device, perform a self-test, and start its server. At this point, an ACE Direct Agent may connect to the Kuando BusyLight™ device from the ACE Direct Agent Portal. The Lightserver GUI has the data elements shown in Table 6.

Table 6. Lightserver GUI Data Elements

Data Element	Description
Status	The current status of the Lightserver program (e.g., Running, Stopped, ...)
Agent Status	The status of the connected Agent (e.g., ready, away, in call, ...)
Vendor	The vendor of the light device; currently only PLENOM is supported
Product	The BusyLight™ device model
Start	Starts the local server
Stop	Stops the local server
Test	Perform a self-test of the connected BusyLight™ device
Show icon	Show or hide the graphical BusyLight™



## 2.8 Consumer Help Center

As shown in Figure 18, the design of the Consumer Help Center, also known as the Consumer Portal, gives Consumers the option to submit information before a call with an Agent. The Consumer uses a web form to submit information to document the complaint.

The following two steps are required to access the Consumer Help Center:

- Start the browser on a machine that can access the Consumer Help Center Node.js server,
- Enter a URL similar to, <https://<hostname>/ACEDirect/Complaint>, where <hostname> is the host name of the ACE Direct server. The exact URL depends on your installation and customization of ACE Direct.

### 2.8.1 Submit a Complaint

The descriptions and web forms presented in Figure 18 through Figure 20 demonstrate how to submit Consumer complaints in the ACE Direct system.

#### 2.8.1.1 Verify Videophone Number

Figure 18 shows the opening page of the Consumer Portal (Consumer Help Center). Consumers enter their videophone numbers here. The ACE Direct system validates the videophone number before allowing the Consumer to proceed.

Figure 18. Screenshot of Consumer Help Center

#### 2.8.1.2 Complete the Consumer Complaint Form

After verifying the videophone number through the iTRS-ENUM database, the portal displays the Consumer complaint form, as shown in the Figure 19 example.

If the Consumer had a prior ticket in the CRM system, the videophone number information provided on the previous form is displayed in the VRS Information section via a ticket lookup. These fields will be empty if this is the Consumer's first call.

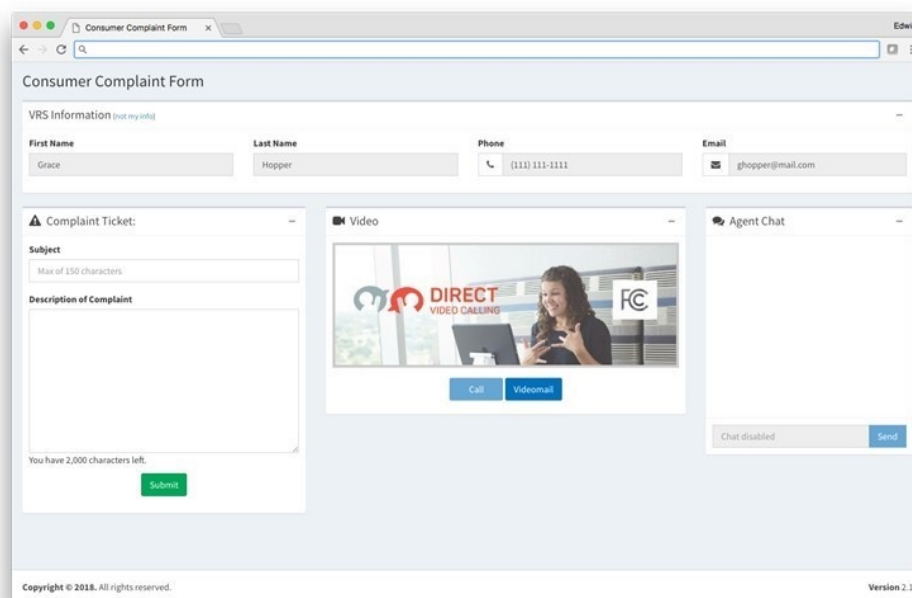
The screenshot shows a web browser window titled "Consumer Complaint Form". The form is divided into several sections. At the top, there's a "VRS Information" section with fields for "First Name" (Grace), "Last Name" (Hopper), "Phone" ((111) 111-1111), and "Email" (ghopper@mail.com). Below this is the "Complaint Ticket:" section, which includes a "Subject" field (Max of 150 characters) and a "Description of Complaint" text area (You have 2,000 characters left). A green "Submit" button is at the bottom of this section. To the right of the "Description of Complaint" is a "Video" section featuring a video player with a woman on the screen and "Call" and "Videomail" buttons. Further right is an "Agent Chat" section with a "Chat disabled" message and a "Send" button. The footer of the browser window shows "Copyright © 2018. All rights reserved." and "Version 2.1".

Figure 19. Screenshot of Consumer Complaint Form

### 2.8.1.3 Complete the Complaint Ticket

The complaint ticket section appears on the left side of the screen. The Consumer enters a subject and description of the complaint to provide background information to the Agent. After populating the subject and description fields, the Consumer clicks “Submit” to file the complaint. The complaint ticket is created in the ticketing system, which enables the “Call” button. This allows the Consumer to communicate with an Agent via video and chat. Figure 20 shows a screenshot of the complaint ticket form.

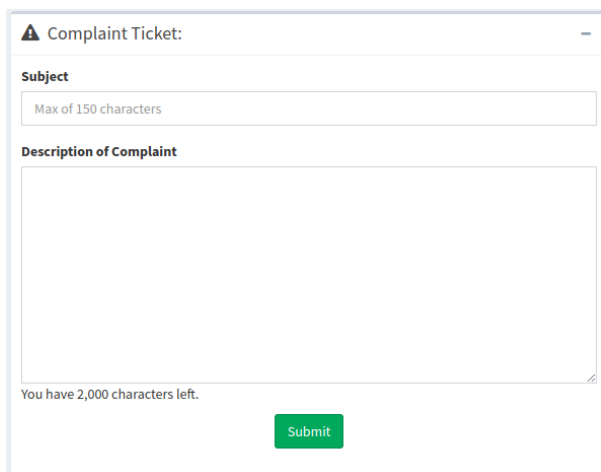
This is a close-up screenshot of the "Complaint Ticket:" section of the form. It shows the "Subject" field with a placeholder "Max of 150 characters" and the "Description of Complaint" text area with a placeholder "You have 2,000 characters left." A green "Submit" button is located at the bottom right of the form.

Figure 20. Screenshot of Complaint Ticket Form

### 2.8.1.4 Use Video Chat

After submitting the complaint ticket and receiving a ticket number, the Consumer presses the “Call” button and is connected to an available Agent. Video is the primary form of communication. As shown in Figure 21, the Agent’s video is displayed in the video box in the center of the screen. During a call, the Consumer has button options to mute audio, mute video, or view the Agent’s video in full screen mode.

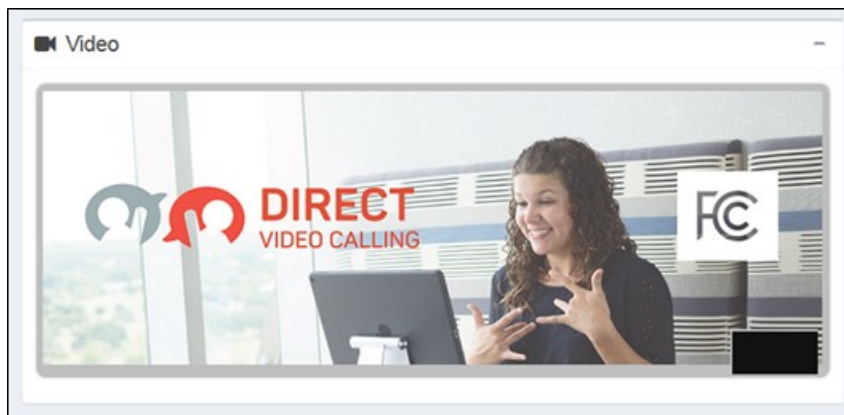


Figure 21. Screenshot of Video Chat Window

### 2.8.2 Use Real-Time Text Chat

The Agent Chat pane on the right side of the screen provides the Consumer with a secondary method of communication with the Agent. Notifications appear while the Consumer is typing a message to the Agent and vice versa. The messages will show up in real time, and the chat history will remain visible until the Agent closes the ticket. As shown in Figure 22, a maximum of 500 characters per line are allowed during the Agent chat.

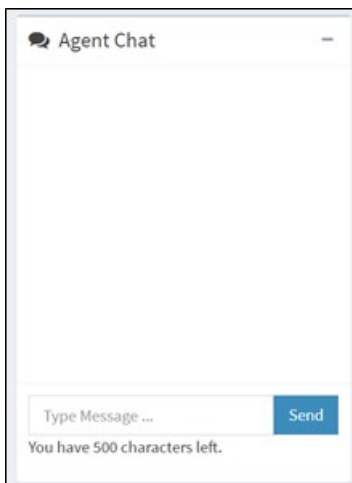


Figure 22. Screenshot of Real-Time Text Chat

When an Agent becomes available, the video chat will begin. After the Consumer or the Agent hangs up, the Consumer is redirected to a page of his or her choosing. For the initial configuration, the FCC.gov website is used for the redirect.

### 2.8.3 Leave a Videomail

A Consumer may leave a videomail during the Consumer complaint by pressing the “Record” button on the screen as shown in Figure 23. This flow is for illustrative purposes and should be customized to fit the needs of your Consumers.

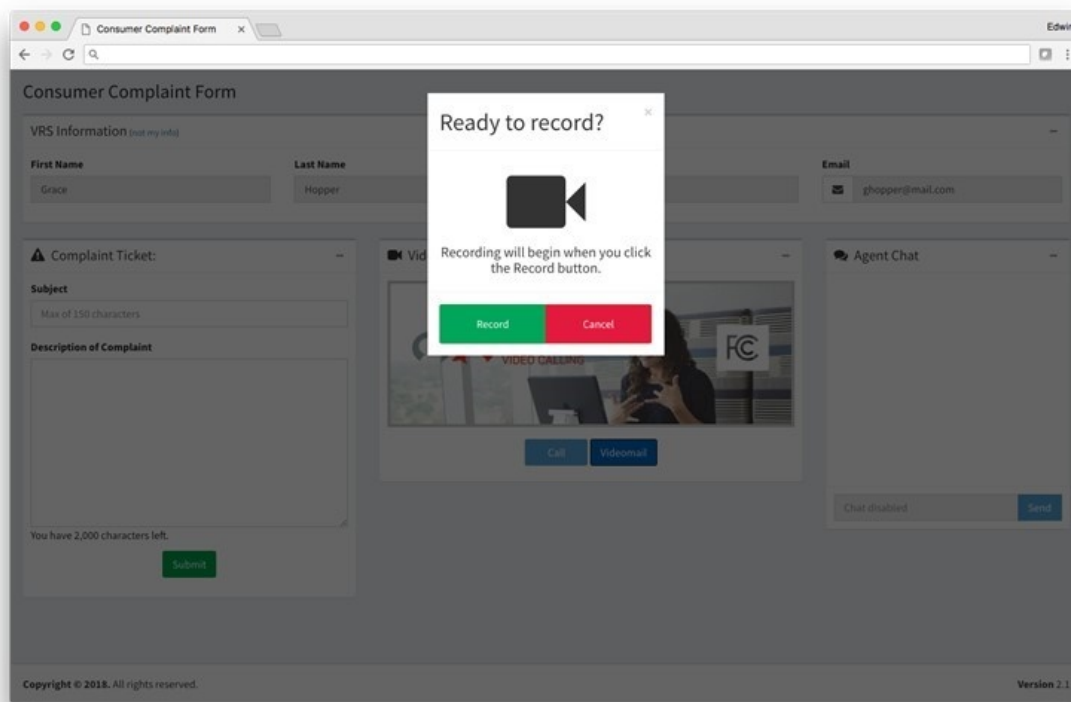


Figure 23. Screenshot of Ready to Record

Consumers see a self-view during recording. A status bar shows the remaining time for the recording (which currently defaults to 90 seconds). The maximum videomail length is a configurable parameter. Figure 24 shows the Consumer Portal with a videomail recording in progress.

The screenshot shows a web browser window titled "Consumer Complaint Form" with a user named "Edwin". The form is divided into several sections:

- VRS Information (not my info)**: Fields for First Name (Grace), Last Name (Hopper), Phone ((111) 111-1111), and Email (ghopper@mail.com).
- Complaint Ticket:** A section with a "Subject" field (Max of 150 characters) and a "Description of Complaint" text area (You have 2,000 characters left). A green "Submit" button is at the bottom.
- Video**: A central video player showing a red "Recording in progress..." overlay with a large red "X" and the "ace direct" logo at the bottom. A red progress bar at the bottom of the video indicates "22 seconds remaining".
- Agent Chat**: A chat window on the right with a "Send" button and a status "Chat disabled".

At the bottom of the page, it says "Copyright © 2018. All rights reserved." and "Version 2.1".

Figure 24. Screenshot of Videomail Recording in Progress

## 2.9 Management Portal

The Management Portal consists of five main components: the Management Dashboard, the Call Detail Record dashboard, Videomail dashboard, Light Configuration page, and Hours of Operation page. These pages present the Manager with information about the operations of the call center, information about incoming calls, videomail management, the ability to customize colors associated with an Agent status on the Kuando BusyLight™ (please refer to subsection 2.6.2), and the ability to manage hours of operation for the call center.

### 2.9.1 Management Dashboard

The Management Dashboard, as shown in Figure 25, provides KPIs for monitoring in real time. Follow these two steps to access the Management Dashboard:

- Start the browser on a machine that can access the Management Portal Node.js server.
- Enter a URL similar to, <https://<hostname>/ManagementPortal>, where <hostname> is the host name of the Management Portal server. The exact URL depends on your installation and customization of ACE Direct.

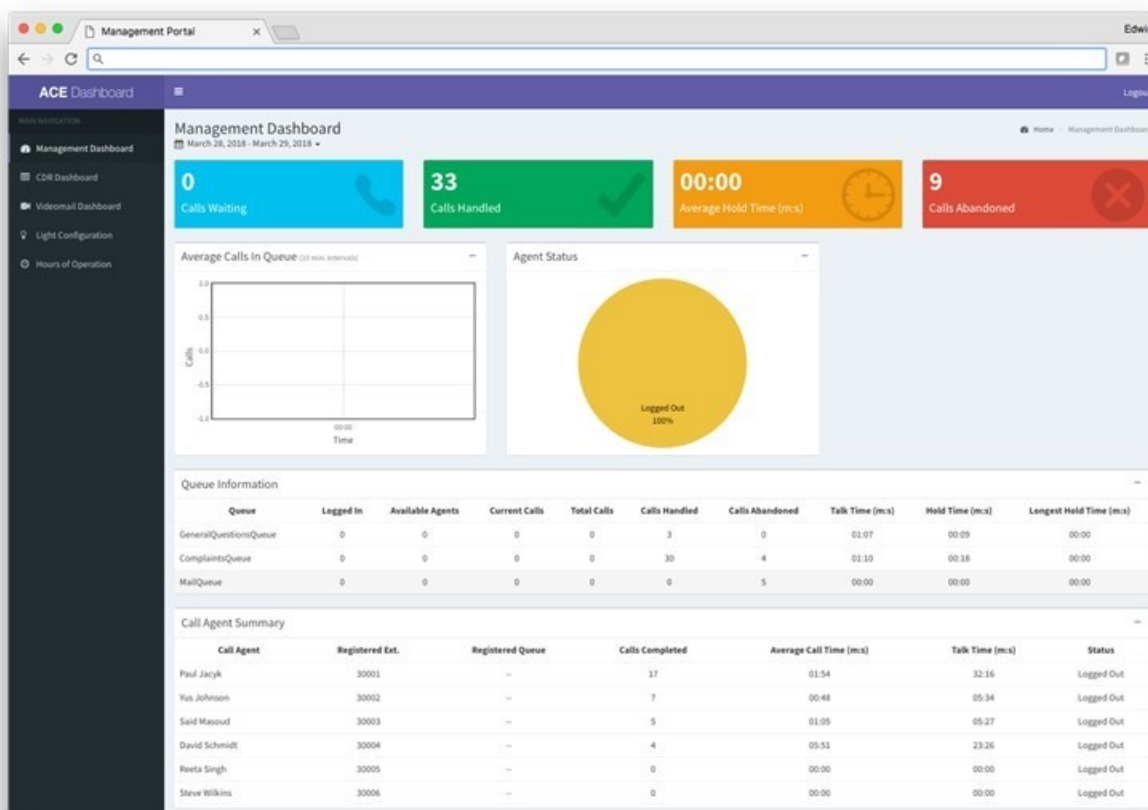


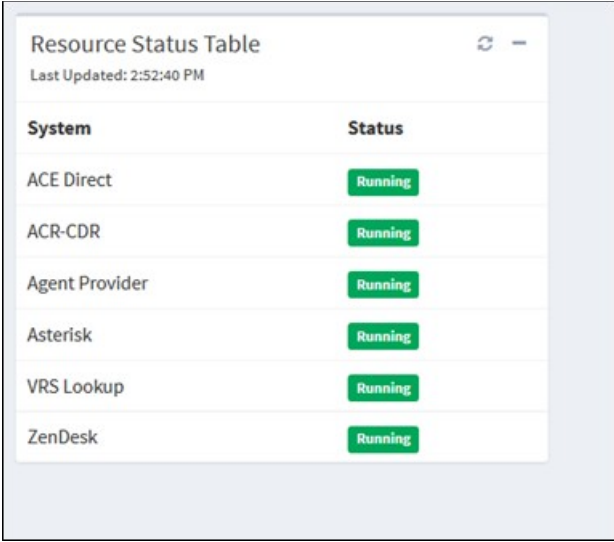
Figure 25. Screenshot of Management Dashboard

## Key Performance Indicator Types

The ACE Direct Management Dashboard presents three types of KPIs:

1. **Summary Data** – There are two queues in ACE Direct: ComplaintsQueue and GeneralQuestionQueue. The following KPIs are a summary over both queues combined.
  - a. Calls Waiting – Number of calls waiting in all queues.
  - b. Calls Handled – Number of calls completed in all queues.
  - c. Average Hold Time (minutes:seconds) – Average call holding time in all queues.
  - d. Calls Abandoned – Number of calls not answered in all queues.
  - e. Average Calls in Queue (chart) – Real-time chart of average calls in queue.
  - f. Agent Status (chart) – Real-time chart of agents logged in/out.
2. **Queue-related KPIs** – The following KPIs are displayed per queue (ComplaintsQueue and GeneralQuestionQueue):
  - a. Logged In – Number of Agents currently logged into the system.
  - b. Available Agents – Number of Agents currently in a ready state.
  - c. Current Calls – Number of calls currently in progress.

- d. **Total Calls** – Total number of calls made.
  - e. **Calls Handled** – Total number of calls answered by an Agent.
  - f. **Calls Abandoned** – Total number of calls abandoned.
  - g. **Talk Time** – Average talk time (minutes:seconds).
  - h. **Hold Time** – Average hold time (minutes:seconds).
  - i. **Longest Hold Time** – The longest hold time (minutes:seconds).
3. **Agent-related KPIs** – The following KPIs are displayed per Agent. The Agent name, extension, and registered queues are displayed along with the KPI:
- a. **Agent name** – Name of the Agent.
  - b. **Registered extension** – Extension assigned to the Agent.
  - c. **Registered queues** – Asterisk queues assigned to the Agent. All queue names are displayed if an Agent is assigned to more than one queue.
  - d. **Calls Completed** – Number of calls handled (answered and completed) by the Agent.
  - e. **Average Call Time** – Talk Time divided by number of calls.
  - f. **Talk Time** – The cumulative time the Agent has spent on calls.
  - g. **Status** – Logged Off, Ready, Away, or In-Call.
4. **Resource Status KPIs** – Figure 26 shows the resource status KPIs:
- a. **Resources** – A list of services required for ACE Direct to properly operate (ACE Direct, ACR-CDR, Agent Provider, Asterisk, VRS Lookup and Zendesk).
  - b. **Status** – The current status of each service (Running or Unavailable).



The screenshot shows a web interface titled "Resource Status Table" with a refresh icon and a timestamp "Last Updated: 2:52:40 PM". Below the title is a table with two columns: "System" and "Status". The table lists six systems, all of which are in a "Running" state, indicated by green buttons.

System	Status
ACE Direct	Running
ACR-CDR	Running
Agent Provider	Running
Asterisk	Running
VRS Lookup	Running
ZenDesk	Running

Figure 26. Screenshot of Resource Status

## 2.9.2 CDR Dashboard

Asterisk generates an Agent Event when a call is completed. A Call Detail Record contains metadata that describes each call, such as the time of the call, call source, and the call destination. The CDR dashboard provides the capability to audit call activity, track a call Agent's activity, and create a report of both incoming and outgoing calls. The CDR Dashboard, as shown in Figure 27, facilitates viewing and exporting of Asterisk CDRs stored in the MySQL database.

The screenshot shows the ACE Dashboard interface. The left sidebar contains navigation links: Management Dashboard, CDR Dashboard (selected), Videomail Dashboard, Light Configuration, and Hours of Operation. The main content area is titled 'CDR Dashboard' and shows a date range of 'March 23, 2018 - March 29, 2018'. Below this is a 'Call Detail Records' section with a search bar and a table of 10 entries. The table has columns for Call Date, Source, Destination, Destination Context, Channel, Last Application, Last Data, Duration Seconds, and Billable Seconds. The data includes calls from 2018/03/27 and 2018/03/26, with destinations like 'Complaints' and 'General Questions'.

Call Date	Source	Destination	Destination Context	Channel	Last Application	Last Data	Duration Seconds	Billable Seconds
2018/03/27 01:00:58 pm	90003	if_vpb	Complaints	PJSIP/90003-0000096	Queue	ComplaintsQueue.....1(codedid*if_vpb*1).T	64	64
2018/03/27 01:00:58 pm	90003	if_vpb	Complaints	PJSIP/90003-0000096	Queue	ComplaintsQueue.....1(codedid*if_vpb*1).T	64	64
2018/03/27 12:54:37 pm	90003	if_vpb	Complaints	PJSIP/90003-0000094	Queue	ComplaintsQueue.....1(codedid*if_vpb*1).T	82	82
2018/03/27 12:54:37 pm	90003	if_vpb	Complaints	PJSIP/90003-0000094	Queue	ComplaintsQueue.....1(codedid*if_vpb*1).T	82	82
2018/03/27 12:47:52 pm	90003	if_vpb	Complaints	PJSIP/90003-0000092	Queue	ComplaintsQueue.....1(codedid*if_vpb*1).T	139	139
2018/03/27 12:47:52 pm	90003	if_vpb	Complaints	PJSIP/90003-0000092	Queue	ComplaintsQueue.....1(codedid*if_vpb*1).T	139	139
2018/03/26 01:04:09 pm	7275514870	start	from-providers_caller_query	PJSIP/ProviderZVRS-00000091	Background	je_mainmenu	1	1
2018/03/26 01:04:09 pm	7275514870	start	from-providers_caller_query	PJSIP/ProviderZVRS-00000091	Background	je_mainmenu	1	1
2018/03/26 01:03:11 pm	7275514870	start	Provider_General_Questions	PJSIP/ProviderZVRS-0000008f	Queue	GeneralQuestionsQueue.....1(from internal*7033439580*1).T	13	13
2018/03/26 01:03:11 pm	7275514870	start	Provider_General_Questions	PJSIP/ProviderZVRS-0000008f	Queue	GeneralQuestionsQueue.....1(from internal*7033439580*1).T	13	13

Showing 1 to 10 of 10 entries

Figure 27. Screenshot of Call Detail Record

The CDR Dashboard allows the user to perform the following actions on the CDRs:

- **Select Date Range** – The Consumer can select a date range for the report. Predefined values are Today, Yesterday, Last 7 days, Last 30 days, This Month, Last Month, All Time (January 1st, 2016 to Today), and Custom Range. The default selection is “Last 7 Days”.
- **Sort Column** – The Consumer can sort on any column by clicking the sort icon located next to each column name. To multi-sort columns, the Consumer depresses the shift key when selecting columns.
- **Show/Hide Columns** – This action expands/condenses the table to show/hide the following columns: Caller ID Text, Destination Channel, Disposition, AMA Flags, Account Code, User Field, Unique ID, Linked ID, Sequence, and Peer Account.



- **Download CSV File** – This action downloads the table as a Comma Separated Values (CSV) file. The CSV file contains only data within the date range.
- **Search** – The user can search the entire table. Search results are displayed in near real time.

Table 7 presents the Call Detail Record Column Definitions in the CDR table.

Table 7. Call Detail Record Column Definition

Display Name	Database Column	Description
Call Date	Calldate	The start datetime of the call. Default format: 2016-09-07T09:35:41Z dashboard formats the date to 2016/09/07 09:35:41 pm (adjusted for time zone).
Caller ID Text	Clid	The full consumer ID, including the name, of the calling party. This field is set automatically and is read-only.
Source	Src	The calling party's caller ID number. It is set automatically and is read-only.
Destination	Dst	The destination extension for the call. This field is set automatically and is read-only.
Destination Context	Dcontext	The destination context for the call. This field is set automatically and is read-only.
Channel	Channel	The calling party's channel. This field is set automatically and is read-only.
Destination Channel	Dstchannel	The called party's channel. This field is set automatically and is read-only.
Last Application	Lastapp	The last dialplan application that was executed. This field is set automatically and is read-only.
Last Data	Lastdata	The arguments passed to the lastapp. This field is set automatically and is read-only.
Duration Seconds	Duration	The number of seconds between the start and end times for the call. This field is set automatically and is read-only.
Billable Seconds	Billsec	The number of seconds between the answer and end times for the call. This field is set automatically and is read-only.
Disposition	Disposition	An indication of what happened to the call. This may be NO ANSWER, FAILED, BUSY, ANSWERED, or UNKNOWN.
AMA Flags	Amaflags	The Automatic Message Accounting (AMA) flag associated with this call. This may be one of the following: OMIT, BILLING, DOCUMENTATION, or Unknown.
Account Code	accountcode	An account ID. This field is user defined and is empty by default.
User Field	Userfield	A general-purpose user field. This field is empty by default and can be set to a user-defined string.
Unique ID	Uniqueid	The unique ID for the src channel. This field is set automatically and is read-only.

Display Name	Database Column	Description
Linked ID	Linkedid	A unique identifier that unites multiple CDR records.
Sequence	Sequence	A numeric value that, combined with uniqueid and linkedid, can be used to uniquely identify a single CDR record.
Peer Account	peeraccount	The account code of the called party's channel

### 2.9.3 Videomail Dashboard

The videomail dashboard, as shown in Figure 28, allows the Manager to track videomail-related information. It shows all videomails that are present in the Agent Portal, along with the date received, Agent that viewed and processed it, caller videophone number, and the videomail status. If an Agent deletes the videomail, the status will show “Marked for Deletion”, at which point the Manager can review it and choose whether to permanently delete it. If the Manager does not act, then the videomail is permanently deleted after 14 days.

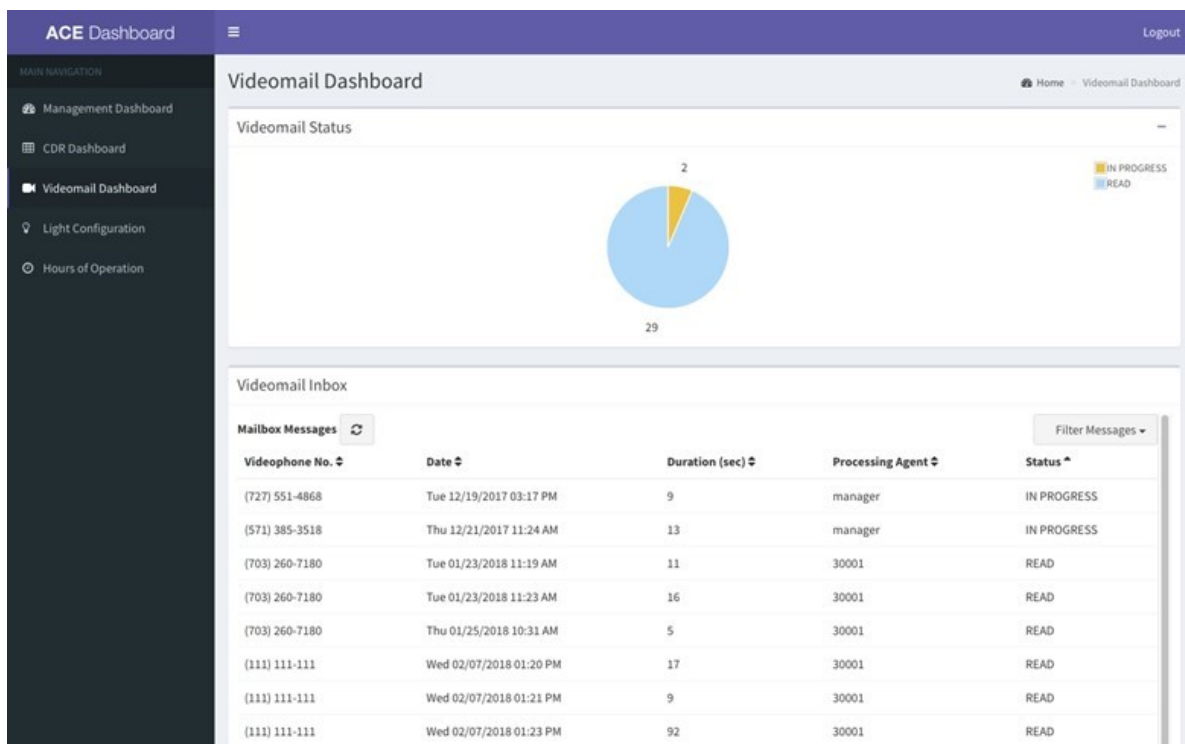


Figure 28. Screenshot of Videomail Dashboard

### 2.9.4 Hours of Operation

The Hours of Operation page, as shown in Figure 29, allows the Manager to control the operating hours of ACE Direct. The page displays the call center hours of operations for each time zone. A Manager can select Open and Close times for the call centers. The Manager also has the option to override the duty hours with an Always Open or Always Closed option. These override options could be used in the case of an emergency closure or holiday. A Consumer who

accesses the ACE Direct Consumer Portal after hours will be presented a message advising that the call center is closed.

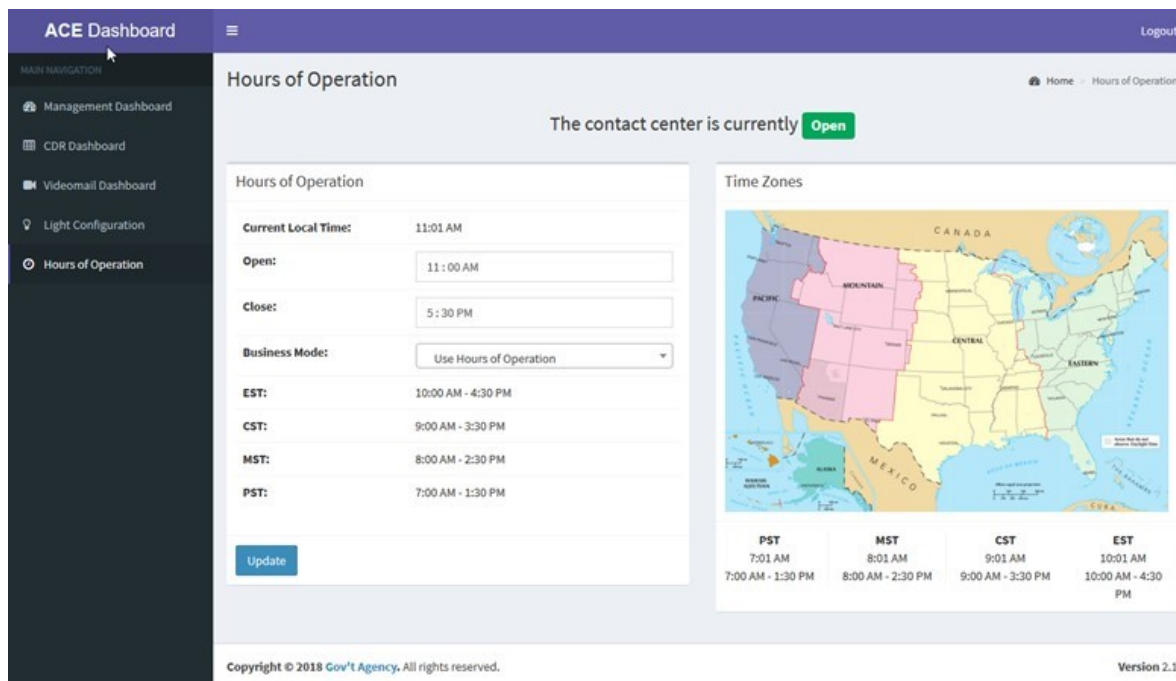


Figure 29. Screenshot of Hours of Operation Page

## 2.10 Identity and Access Management

ACE Direct integrates with ForgeRock OpenAM, an open source identity and access management enterprise solution that provides user management and access control capabilities. The ForgeRock OpenAM and embedded OpenDJ are the only ForgeRock packages used in ACE Direct.

Every type of ACE Direct user, such as Agent, Manager, or Supervisor, must be provisioned in OpenAM before users access the Management, Agent, or Consumer portals. Provisioning instructions to create new roles can be found in Provision.md under <https://github.com/FCC/ACEDirect> in the 'iam' section.

There are two ways a user authenticates with the Agent or Management portals:

1. **ACE Direct URL** – Access ACE Direct directly by entering the ACE Direct URL. If the user is already authenticated with ACE Direct and the session is still valid, the Agent Portal will be displayed. Otherwise, the user is re-directed to the Main login page to authenticate. The Agent Portal is displayed after a successful login.
2. **An OpenAM login URL (Main login page)** – This is the main login page for a user to authenticate, reset the password, or create a new account. The user selects the application to access as listed on the dashboard after logging in.

The following subsections describe the identity and access management capabilities in detail.

## 2.10.1 Login Screen

Figure 30 shows the login screen.

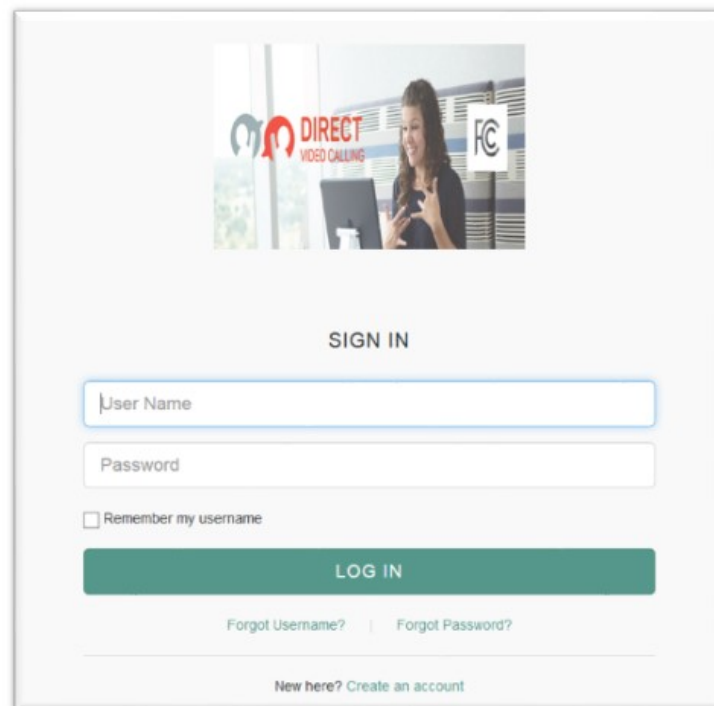


Figure 30. Screenshot of Login Screen

The login screen consists of the following elements:

1. **Login** – User provides login credentials (username, password) to authenticate to ACE Direct.
2. **Forgot Username** – Redirects the user to the “Forgot Username” page to retrieve username.
3. **Forgot Password** – Redirects the user to the “Forgot Password” page to reset password.
4. **Self-Registration** – Allows the user to self-register and create an account.

### 2.10.1.1 Forgot Username

A user clicks the “Forgot Username” link to retrieve a forgotten username. The following “Retrieve Your Username” screen is displayed as shown in Figure 31.

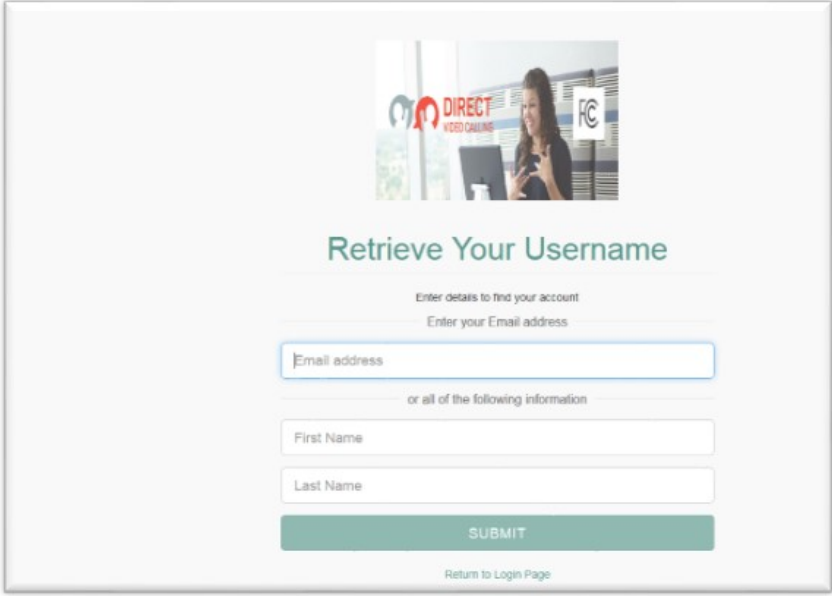
The screenshot shows a web form titled "Retrieve Your Username" in a teal font. At the top, there is a banner image featuring a woman on a video call, with the "DIRECT VIDEO CALLING" logo and the FCC logo. Below the title, the text "Enter details to find your account" is displayed. The form has two main sections: "Enter your Email address" with a single text input field labeled "Email address", and "or all of the following information" which includes two text input fields labeled "First Name" and "Last Name". A teal "SUBMIT" button is positioned below these fields. At the bottom of the form, there is a link that says "Return to Login Page".

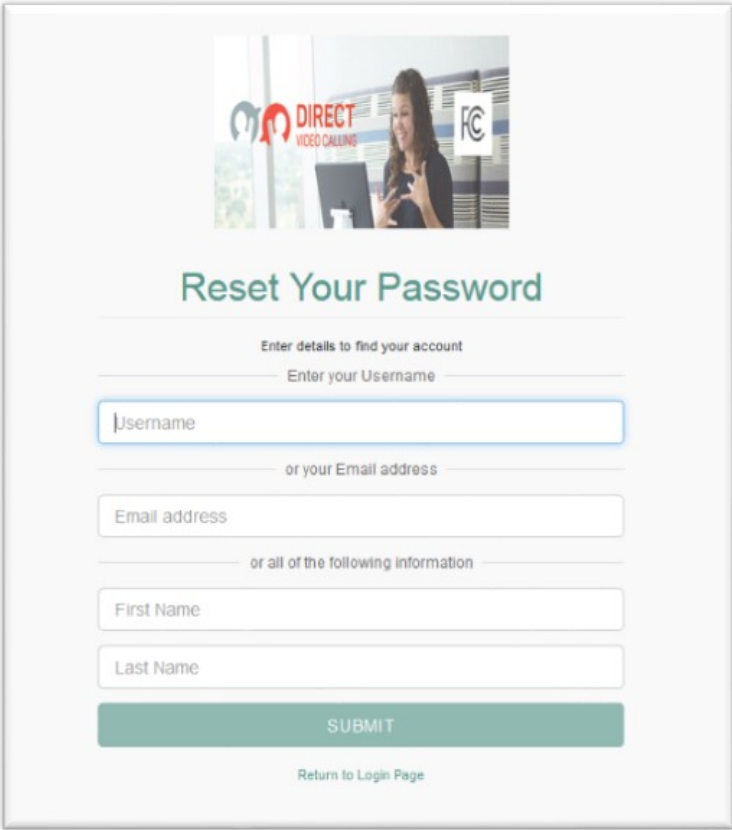
Figure 31. Screenshot of Retrieve Your Username

A user may retrieve their username via:

- **Email** – If the user entered his/her email on the reset username screen.
- **Answering security questions** – The user is prompted with security questions after he/she submits First Name/Last name on the retrieve username screen. If the user answers the security questions correctly, the screen displays the user's dashboard.

### 2.10.1.2 Forgot Password

Figure 32 shows the “Reset Your Password” screen displayed when the user clicks the “Forgot Password” link to reset password.



**Reset Your Password**

Enter details to find your account

Enter your Username

Username

or your Email address

Email address

or all of the following information

First Name

Last Name

SUBMIT

[Return to Login Page](#)

Figure 32. Screenshot of Reset Your Password

Users may reset the password via:

- **Email** – If the user entered his/her email on the reset password screen.
- **Answering security questions** – The user is prompted with security questions after he/she submits username or First Name/Last name on the reset password screen. If the user answers the security questions correctly, the screen displays the user's dashboard.

### 2.10.1.3 Self-Registration

Any person can self-register and create an account to become a user; however, the account is not activated until the ACE Direct Administrator activates the user account. To create such an account, this person clicks the “Create an account” link at the bottom of the login page in Figure 30. As shown in Figure 33, the “Register Your Account” screen will pop up to allow the new user to enter his or her email address and submit the registration request. An email will be sent to the user's email address to continue and complete the creation of the account.

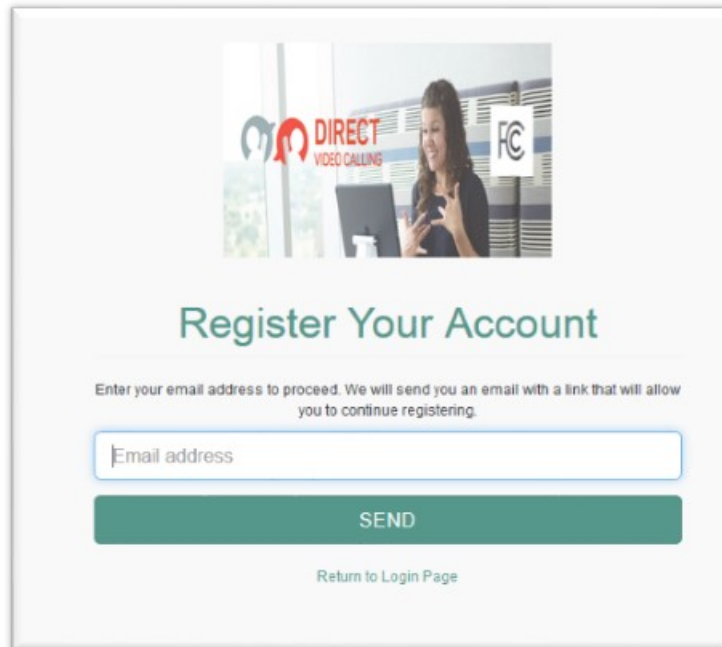


Figure 33. Screenshot of Register Your Account

## 2.10.2 User Dashboard

Figure 34 shows the User Dashboard page after the user is authenticated with the Main login page.

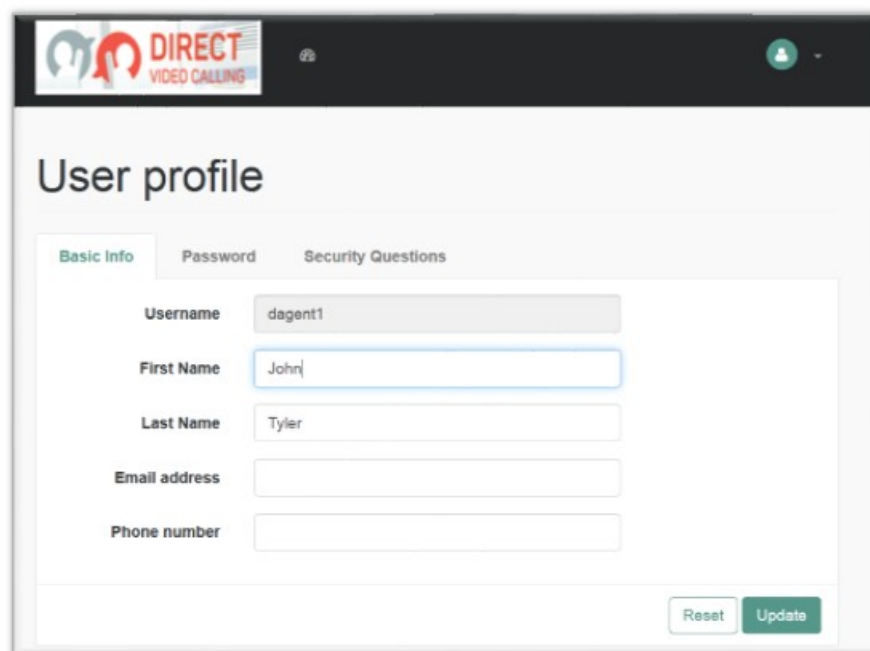


Figure 34. Screenshot of User Dashboard

The User Dashboard page consists of the following sections:

- User profile
  - Basic Info
  - Password
  - Security Questions
- Dashboard
- Logout

### 2.10.2.1 User Profile

#### 2.10.2.1.1 Basic Info

Figure 35 shows the Basic Info tab on the User Profile screen.

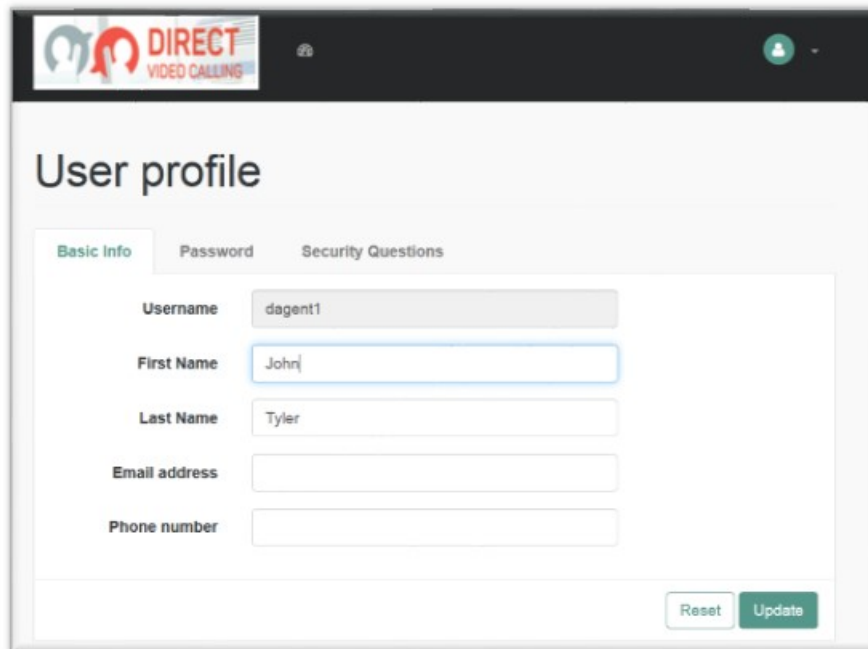
The screenshot shows a web interface for a 'User profile'. At the top, there is a dark header with the 'DIRECT VIDEO CALLING' logo on the left and a user profile icon on the right. Below the header, the page title 'User profile' is displayed. Underneath the title are three tabs: 'Basic Info' (which is highlighted with a green border), 'Password', and 'Security Questions'. The 'Basic Info' tab contains a form with the following fields: 'Username' (with the value 'dagent1'), 'First Name' (with the value 'John'), 'Last Name' (with the value 'Tyler'), 'Email address', and 'Phone number'. At the bottom right of the form are two buttons: 'Reset' and 'Update'.

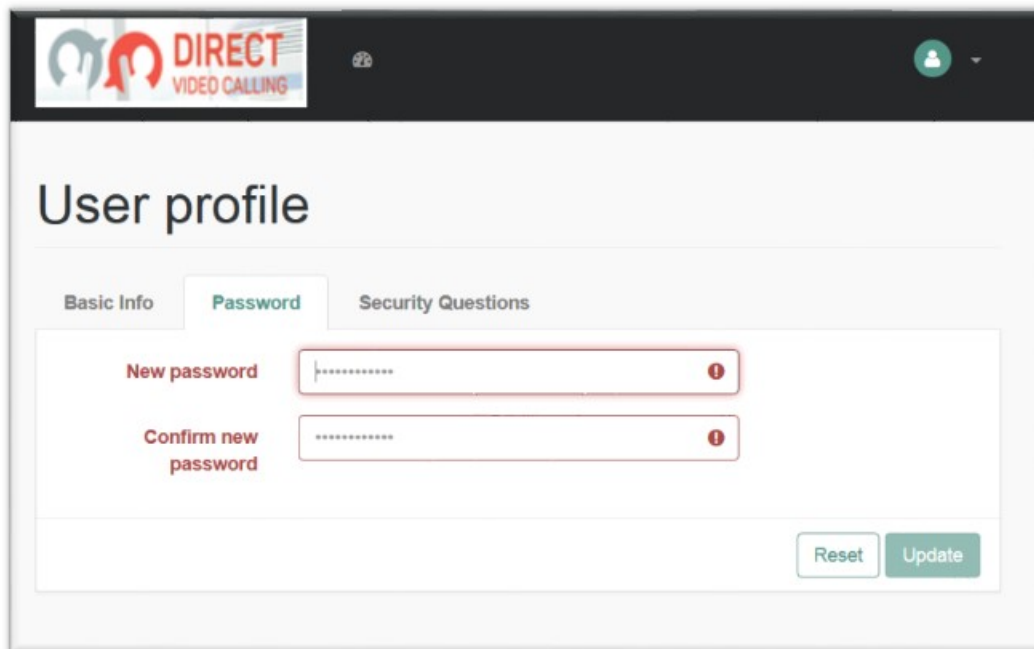
Figure 35. Screenshot of Basic Info Tab

The user views or updates his/her first name, last name, email address, or phone number on this page, and clicks “Update” to save the changes.

#### 2.10.2.1.2 Password

Figure 36 shows the Password tab on the User Profile screen. This screen allows the user to update his or her password. The user clicks on the Password tab to change password, and clicks “Update” to save changes.



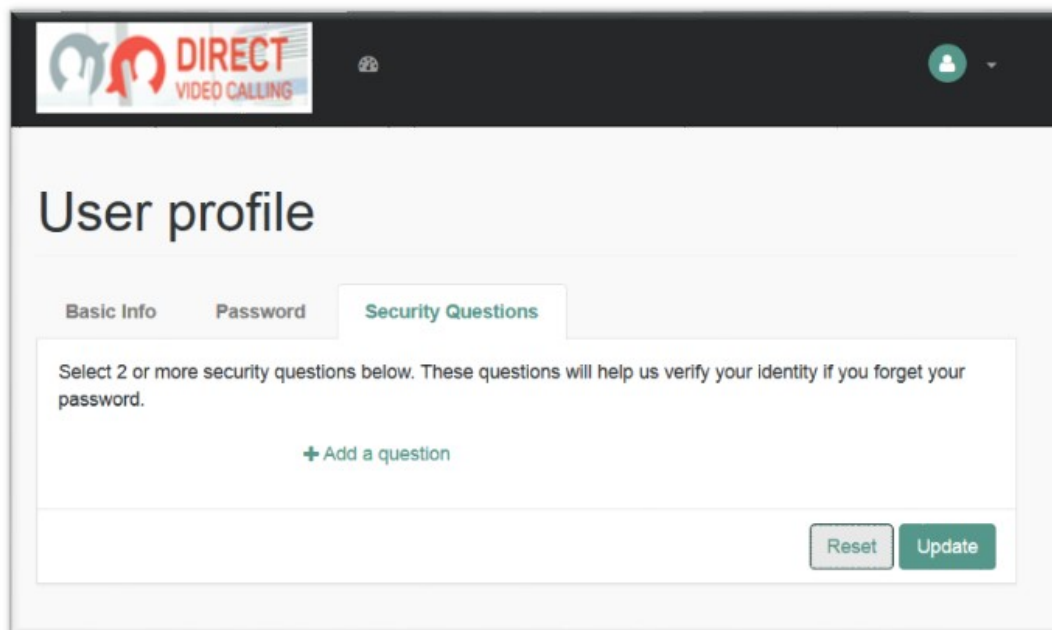


The screenshot shows the 'User profile' page with the 'Password' tab selected. It features two input fields: 'New password' and 'Confirm new password', both with red error icons. 'Reset' and 'Update' buttons are at the bottom right.

Figure 36. Screenshot of Update Password Tab

#### 2.10.2.1.3 Security Questions

Figure 37 shows the Security Questions tab on the User Profile screen. A user may retrieve his/her password via email by answering security questions. This screen allows the user to add security questions to retrieve a forgotten password by email.



The screenshot shows the 'User profile' page with the 'Security Questions' tab selected. It displays instructions: 'Select 2 or more security questions below. These questions will help us verify your identity if you forget your password.' Below this is a '+ Add a question' link. 'Reset' and 'Update' buttons are at the bottom right.

Figure 37. Screenshot of Security Questions Tab

Figure 37 is a screenshot of the Security Questions tab as described in the text immediately preceding and following the figure.

The user must provide at least two security questions for this purpose. To accomplish this, the user must:

- Select the “Security Questions” tab.
- Select a pre-defined security question or create a new security question.
- Provide an answer to the question.
- Repeat for at least one additional security question.
- Click “Update” to save the changes.

### 2.10.2.2 Dashboard

The dashboard is an area where the user views the list of applications he/she is approved to access. Figure 38 shows an example of an Agent’s dashboard and a list of approved applications under the My Application Drop Down.

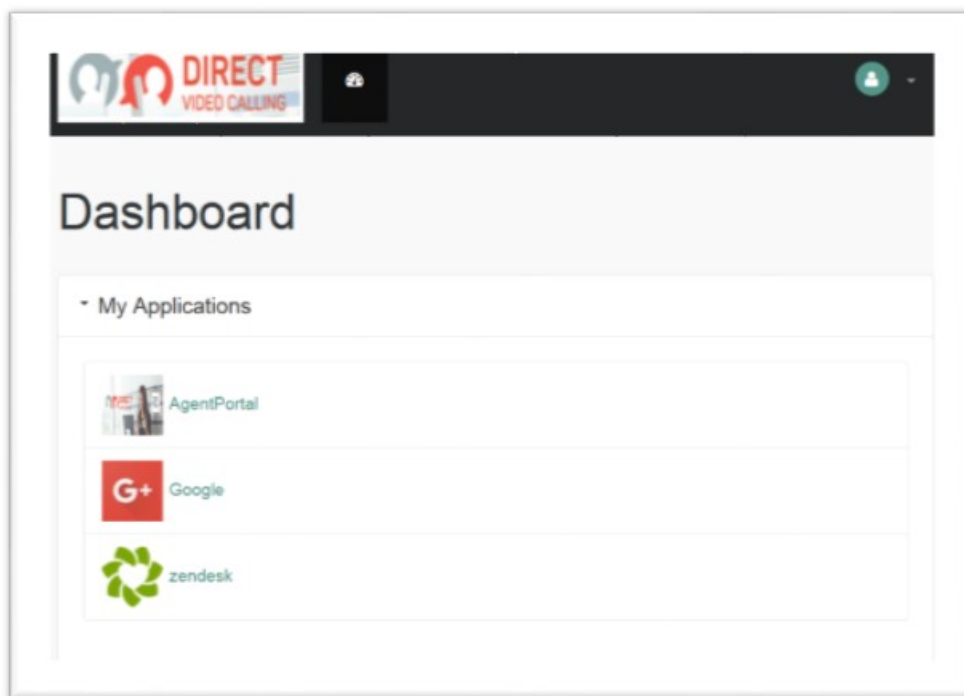


Figure 38. Screenshot of User Dashboard

### 2.10.2.3 Logout

To logout from the Main Login page and all open sessions to the Agent and/or Management Portal, select “LOG OUT” from the dropdown list at the top right corner of the page as shown in Figure 39.

Figure 39. Screenshot of Logout Screen

## 2.11 NGINX

NGINX is an open source HTTP and reverse proxy server. For ACE Direct, only the reverse proxy component is used. A reverse proxy allows ACE Direct to hide internal topology (ports and script names) from users.

Figure 40 shows the mapping between public-facing and internal URLs performed by NGINX. The left side of the figure shows the URLs available to the public via HTTPS. Each URL maps to another path and port internally on the Node.js server. In this example, NGINX is mapping port 443 on the public-facing interface to ports 8005 and 8081 on the internal side.

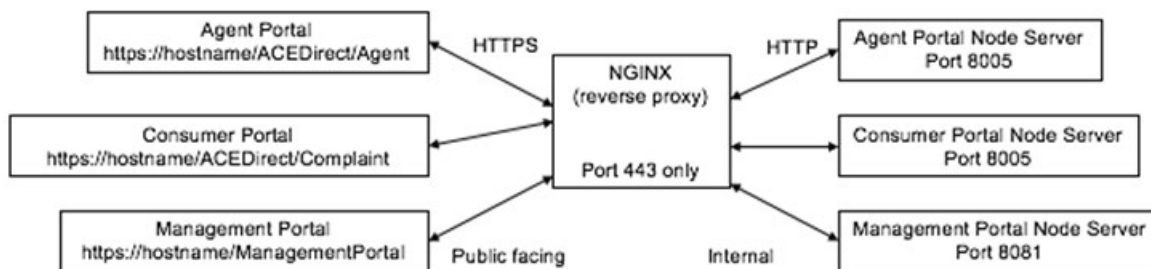


Figure 40. Internal and External Paths with NGINX

In ACE Direct, NGINX is configured to work with OpenAM, directing incoming connections to the correct location based on URL and authentication level. Figure 41 shows the relationship between NGINX and OpenAM and the mapping between public-facing URLs (to the left or outside of the firewall) and the internal processes.

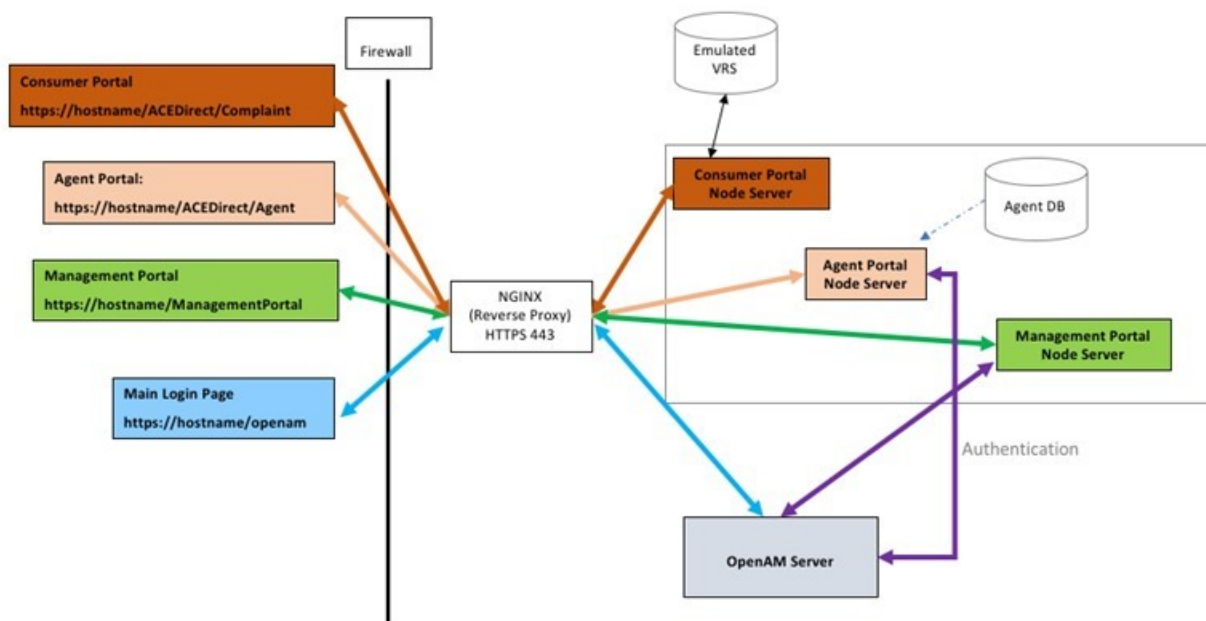


Figure 41. NGINX and OpenAM Integration

## 2.12 Redis

Redis is an open source, in-memory, key-value data store. In ACE Direct, Redis is used to store state data that were previously stored in memory in the Node.js server. Some of the data stored in Redis include agent status (active or away), maps of agent extensions, and agent information.

Redis offers advantages over in-memory storage. For example, Redis allows several applications to share the same data. This ensures that all applications have the same view of system state. Another advantage is that Redis supports different data formats, which means it can store more than just typical key-value pairs. The most widely used data types are strings; however, Redis also supports lists, sets, sorted sets, and hashes. Because Redis can also write data to disk, it can maintain state after a system restart. Additional information about Redis can be found here: <https://redis.io>.

### 3. Installation and Configuration

This section presents guidance for the installation and configuration of the ACE Direct components except for OpenAM and the Kuando BusyLight™. To reproduce this version of the platform, review the README.md file in the autoinstall folder at <https://github.com/FCC/ACEDirect>.

Table 8 presents a list of initial requirements that must be met before installation.

Table 8. Initial Installation Requirements for ACE Direct

Prerequisite	Rationale
A domain name (your domain name must not contain an underscore)	Needed to provide end users with access to the ACE Direct Platform via a web browser.
Second-level DNS subdomain names and corresponding Address Records (A records) (e.g., "host.example.com")	Needed to point the subdomains to a specific IP address used in components of ACE Direct.
Servers must have external internet access	Needed for external connectivity and public-facing components.
The Asterisk/NGINX/STUN servers have a public and local IP address	Needed to facilitate application network traffic.
SELinux and IPv6 disabled on all hosts	ACE Direct does not currently support these technologies.
A "dial-in" number that has been registered in iTRS and/or a SIP trunk provider (such as Twilio)	To allow Consumers to place calls to ACE Direct. <b>Note:</b> FCC rules govern access to the iTRS database.
IPtables disabled for Asterisk and NGINX	Disable if not required.
Wildcard certificate (preferred)	Allows a single certificate to be used for all subdomains; otherwise, each FDQN will need its own certificate.
Create 'acedirect' user on servers	Required for node.js installation.
Access to system package mirrors (Needed for 'yum install')	Needed to install the necessary package dependencies for the ACE Direct Platform.
Chrome browser is required for Agent Desktop	Google Chrome browser leads the industry in WebRTC integration. Additionally, Chrome is used for all ACE Direct testing and provide the highly level of stability.
Purchase Busylight hardware (optional)	The BusyLight™ provides call centers with a visual indication of the Agent's status.

#### 3.1 SSL/TLS Certificate

The applications within ACE Direct use Hypertext Transport Protocol Secure (HTTPS) to provide security during web transmissions. To prevent major web browsers from potentially flagging the application as untrusted, it is recommended to acquire and implement a Secure Sockets Layer/Transport Layer Security (SSL/TLS) certificate from a trusted certificate authority (CA), such as LetsEncrypt (<https://letsencrypt.org/>). After obtaining a certificate, follow the

instructions in the respective README.md files in the various ACE Direct repositories to install the SSL certificate.

ACE Direct also employs the HTTP Strict Transport Security (HSTS) web policy mechanism to protect against protocol downgrade attacks and cookie hijacking.

## 3.2 Asterisk Installation and Configuration Script

An automated installation script has been developed to assist in the deployment and configuration of the Asterisk server. The previous manual installation procedures have been deprecated and are no longer supported. The Asterisk install script will install and configure Asterisk for ACE Direct on a CentOS 7 server. This script, which is available at <https://github.com/mitrefccace/asterisk> in the ‘scripts’ directory of the Asterisk repository, can be used to quickly stand up an Asterisk instance. The script will perform the following tasks automatically:

- Update current packages and install required packages for Asterisk
- Install PJSIP
- Install Asterisk
- Configure and apply Asterisk configs, media, scripts, and custom patches
- Start Asterisk

It is necessary to satisfy several prerequisites to the script before installing Asterisk. These prerequisites are listed in the top-level README.md file of the Asterisk repo. Review the README.md file in the Asterisk repository before running the script.

### 3.2.1 How It Works

The Asterisk PBX system relies on the configuration of three key files: pjsip.conf, extensions.conf, and http.conf. The pjsip.conf file contains the connection endpoints and parameters, and the extensions.conf file contains the syntax and programming for the extensions assigned to those endpoint connections. The http.conf file is the server configuration for Asterisk and defines the certificates Asterisk uses for secure communications.

#### 3.2.1.1 Dial Plan Configuration

The dial plan is defined by the extensions.conf and pjsip.conf files in the /etc/asterisk directory, which work together to establish the connection between devices and route the calls to those devices. An endpoint device is any device that places or receives the call. Table 9 shows an example of common endpoint devices, the associated extension, and the required codecs for the sample dial plan configuration in this guide.

Table 9. Example Asterisk Endpoint Extensions

Extension	Purpose	Device	Codec
30001	ACE Direct Agent 1	softphone	h264/vp8/ulaw
30002	ACE Direct agent 2	softphone	h264/vp8/ulaw
30003	ACE Direct agent 3	softphone	h264/vp8/ulaw
30004	ACE Direct agent 4	softphone	h264/vp8/ulaw

As displayed in Table 9, the extension is the number assigned to the endpoint. The ability of that endpoint to connect to the Asterisk instance is configured in the `pjsip.conf` file and the ability to dial to another endpoint is configured within the `extensions.conf` file. The device can be any phone capable of connecting to the Asterisk instance—a softphone, a desktop phone, a Cisco video communication phone, or other such device. At present, the configuration provided with ACE Direct can only be used with WebRTC and VRS provider devices; other types of devices are not officially supported. The codec refers to the preferred media codec, or in some cases, the only available codec, used by the endpoint. The codec is also configured in the `pjsip.conf` file.

### 3.2.1.2 Extensions.conf Configuration File

The `extensions.conf` file is the configuration of your PBX. This file defines how the endpoint extensions communicate with each other or with the Asterisk server itself. The Asterisk server consists of many different applications working together to perform different functions. These functions can be called in the `extensions.conf` dial plan to route an endpoint device to a desired outcome. For example, a Consumer wants to dial voicemail. In `extensions.conf`, that configuration must state that the Consumer’s phone number is to load the voicemail application, which is done by the following syntax:

```
exten => _6XXX,1,Answer( )
      same => n,VoiceMail(1234@ourpbx)
```

In this example, any extension matching the pattern `_6XXX` will be answered for immediate placement into the videomail application.

For specific phone numbers to route, as well as number schemes for unknown numbers, use the wildcard “X”. In the following example as shown in the context “[from-internal]” of the `extensions.conf` file, all extensions ended with `6xxx` are configured. The “[from-internal]” label refers to the context in which those extensions reside. A context is an organizational container that can host a group of extensions and extension patterns that can route and dial to other extensions within the same context. As shown in this example, the file needs no other changes.

```
[from-internal]
exten => _50XX,1,Dial(PJSIP/${EXTEN})
same => n,DumpChan( )
same => n,HangUp( )

exten => _6XXX,1,Dial(PJSIP/${EXTEN})
same => n,DumpChan( )
same => n,HangUp( )
```

```
exten => _70XX,1,Dial(PJSIP/${EXTEN})
same => n,DumpChan( )
same => n,HangUp( )
```

Extension patterns use wildcards to encompass a greater range of possible matches to the same string. To dial to an extension in another context, that context must be specifically identified in the coding. The context defined must also be the same for the “Context” attribute of the endpoints defined in the pjsip.conf file (please refer to subsection 3.2.1.3). For the Dial() function call, the extensions/numbers defined within the function call must be defined in pjsip.conf, as shown in the following subsection.

### 3.2.1.3 pjsip.conf Configuration File

The pjsip.conf file defines the endpoint connection parameters. Key elements must be configured in this file. The transport protocol settings and extension must be identified. The extension profile serves as its SIP identity and phone extension. In addition, the file must define the authentication method for this extension. PJSIP does not require the specification of a protocol and will dynamically select the best available option to use. The extension profile must specify the configuration settings for the endpoint, the network connection, the allowed codecs, and NAT traversal settings. The extension profile may also require that particular settings be specified to not be used or disabled. It is highly recommended that you visit the Digium Asterisk website to thoroughly understand the function of the pjsip.conf file and its associated settings.

The Asterisk PBX uses the pjsip.conf file to load the connection points from which your endpoint devices will acquire their information and connect. This means that this file is responsible for (1) the endpoint connection, (2) handling the contact that connects to that endpoint, (3) routing the call, and (4) passing information. The following is a list of critical attributes defined in a profile within pjsip.conf:

- **Transport** – The transport defined within the pjsip.conf file specifies the configuration for TCP, UDP, WS, or WSS connections. These transport settings require configuration as to the external IP, internal IP, and in the case of web sockets, certificate information to be used for secure communication.
- **Endpoint** – An endpoint to the pjsip.conf file is a profile that matches to an endpoint device. The endpoint can be defined in pjsip independently or as a template that numerous extensions can call. This template helps to keep a condensed file because otherwise this file can grow quite large. Endpoint templates are identified by (!) after the identifying name.
- **Register** – A registration is a type of authentication from an endpoint device. The device will send authentication information to Asterisk that will then “Register” the device to the PBX with the assigned endpoint profile. A registration is a temporary assignment of configuration options and network settings to identify the endpoint device and route calls.
- **Contact** – A contact is the network identifying information of a device that has registered and its corresponding extension. Viewing the contact information within the Asterisk console verifies that an endpoint device has authenticated and been assigned its configuration correctly.



Once the profile has been defined, extensions and numbers can be associated with the profile, which will be referred to back in the extensions.conf file to help route calls to their proper destination. Use the pre-defined profiles in the Asterisk repository, because they are already configured specifically for use with ACE Direct.

### 3.2.2 Web Secure Sockets

This Consumer portal uses a technology called WebRTC. Asterisk must use web sockets and, for most browsers, secure web sockets to successfully communicate with WebRTC. It is highly recommended that Asterisk use a certificate from a valid certificate authority when using WebRTC.

### 3.2.3 Sample Configuration Files

**Note:** The following samples are not up to date with the latest configuration in the Asterisk repository. The intent is to provide a basic overview of some of the configuration and their associated parameters. To view the latest configuration files, visit the Asterisk GitHub repository.

#### 3.2.3.1 Pjsip.conf Configuration

The following is an example of the Pjsip.conf file used by Asterisk:

```
-----top of pjsip.conf file-----
-----
[transport-      wss] ;name of transport configuration

type=transport          ; type being configured is of type
transport
protocol=      wss      ;protocol is web secure socket, can be
tcp,udp,ws,wss
bind=0.0.0.0:443        ;bound ip/port
external_media_address=52.45.109.230 ;external ip of Asterisk
cert_file  =/etc/asterisk/keys/star.pem      ;certificate pem
priv_key_file=/etc/asterisk/keys/star.key     ;certificate key

[endpoint-basic](!) ;Defines the name of the endpoint, the (!)
designates it as a template

type= endpoint      ;defines the type
transport=transport-wss ;defines the transport to be used
context=    from-internal ;defines the context, must correspond
to context in extensions.conf
disallow=    all      ;explicitly deny all media unless specified
allow=      ulaw      ;specify allow ulaw audio codec
allow=vp8      ;specify allow vp8 video
allow=h264      ;specify h264 video
allow=t140      ;specify allow t140 text protocol
force_rport=yes ;network configuration, reflexive port
direct_media=no ;network configuration, do not establish direct
media link (NAT)
```

```

rewrite_contact=yes
media_address=52.45.109.230
rtp_symmetric=yes
ice_support=yes
message_context =internal-im      ;context for internal messenger

[30001](endpoint-basic)      ;Extension information, extension
username is 30001
auth =auth30001 ;The auth profile to use
aors =30001      ;The aors profile to use

[auth30001](auth-userpass)
password= changeit! ;password to be changed
username=30001      ;username for authentication, typically
extension number

[30001](aor-single-reg)
remove_existing =yes ; Remove pre-existing contact
max_contacts    =1   ;Number of simultaneous contacts registered
to an AOR
qualify_frequency    =5   ;Interval in seconds of qualifying a
registered contact
authenticate_qualify=yes ;Send authentication request on
qualify if required
-----end of pjsip.conf file-----
-----

```

### 3.2.3.2 WebRTC Sample Configuration

The following is a sample WebRTC-enabled endpoint template in pjsip.conf:

```

[endpoint-webrtc](!)
type=endpoint
transport=transport-wss
context=from-internal
disallow=all
allow=ulaw
allow=h264
allow=vp8
allow=tl140
force_avp=yes
use_avpf=yes
media_encryption=dtls
dtls_verify=fingerprint
dtls_fingerprint=SHA-1
dtmf_mode=auto
dtls_rekey=0
dtls_cert_file=/etc/asterisk/keys/asterisk.pem
dtls_ca_file=/etc/asterisk/keys/ca.crt
dtls_setup=actpass
ice_support=yes

```

```
media_use_received_transport=yes
rtp_symmetric=yes
force_rport=yes
rewrite_contact=yes
message_context=internal-im
rtcp_mux=yes
```

### 3.3 Node.js

Node.js is a platform built on Chrome's V8 JavaScript run-time engine, which was developed to build fast and scalable network applications. Node.js is event driven, lightweight, and efficient—ideal for data-intensive real-time applications that run across distributed devices.

Node.js servers provide the functionality of ACE Direct. More specifically, the servers host the Agent, Consumer, Management, CDR, and Virtual Agent Videomail portals. Node.js also provides Agent Data and Provider Data through RESTful APIs to ACE Direct.

Examples of these RESTful services are VRS lookup and Agent verify functions. An Asterisk instance should already be up and running to support most of the Node.js instances.

Instructions for downloading and installing Node.js can be found on the official Node.js website (<https://nodejs.org/en/>). For this version of ACE Direct, the Node.js server is built with the following software versions:

- Operating System: CentOS 7.x
- Node.js: Version 7.2.1

#### 3.3.1 Node.js Components Installation

The manual installation procedures for the Node.js components of ACE Direct have been deprecated and are no longer supported. To install the Node.js components, follow the README.md file in the autoinstall folder located at <https://github.com/FCC/ACEDirect>.

### 3.4 Management Portal

The ACE Direct Management Portal provides a number of key performance indicators for real-time monitoring by the call center Manager. This information may be used to improve user experience, increase user satisfaction, and monitor overall call center performance. Some of the configurations can be found in `parameter_desc.json` of the `dat` repo, and `queues.conf` of the `asterisk` repo.

#### 3.4.1 Log Files

Log files are located in `managementportal/logs`. The debug level is defined by the `debug_level` field in the configuration file. The valid options are as follows: ALL, TRACE, DEBUG, INFO, WARN, ERROR, and FATAL.

## 3.4.2 Management Dashboard

### 3.4.2.1 Web Server and Client Configuration

The Management Dashboard functionality relies on the Asterisk server to collect reports on call agents and queue status. The Management Dashboard (dashboard.js and dashboard.ejs) is hosted on the Node.js server (server-db.js). The following subsections describe the operational data flow.

#### 3.4.2.1.1 Server Side

- **server-db.js** – Communicates with the Asterisk Server through the Asterisk Management Interface (AMI) protocol. AMI events provide the Management Portal with data on queue and agent status.

#### 3.4.2.1.2 Client Side

- **dashboard.ejs**– Controls the UI for the Management Portal Dashboard.
- **dashboard.js** – Uses JavaScript data structures to store information to be displayed on the dashboard.ejs page.

### 3.4.2.2 Asterisk Management Interface

The AMI allows a client application to connect to an Asterisk instance and issue actions (requests) and receive events (responses). The Management Portal performs five unique AMI action calls to collect data on queue and agent status. Table 10 shows the AMI actions and descriptions.

Table 10. AMI Actions and Descriptions

AMI Action	Description	Syntax	Arguments	Triggered Events
Agents	Queries for information about all agents	Action: Agents ActionID: <value>	<ul style="list-style-type: none"> <li>• ActionID – ActionID for this transaction. Will be returned.</li> </ul>	Agents AgentsComplete
QueueReset	Resets the running statistics for a queue	Action: QueueReset ActionID: <value> Queue: <value>	<ul style="list-style-type: none"> <li>• ActionID – ActionID for this transaction. Will be returned.</li> <li>• Queue – The name of the queue on which to reset statistics.</li> </ul>	
Queues	Queries for queues information	Action: Queues		Queues

AMI Action	Description	Syntax	Arguments	Triggered Events
QueueStatus	Check the status of one or more queues	Action: QueueStatus ActionID: <value> Queue: <value> Member: <value>	<ul style="list-style-type: none"> <li>ActionID – ActionID for this transaction. Will be returned.</li> <li>Queue – Limit the response to the status of the specified queue.</li> <li>Member – Limit the response to the status of the specified member.</li> </ul>	QueueStatus QueueStatusComplete QueueMember QueueParams
QueueSummary	Requests Asterisk to send a QueueSummary event	Action: QueueSummary ActionID: <value> Queue: <value>	<ul style="list-style-type: none"> <li>ActionID – ActionID for this transaction. Will be returned.</li> <li>Queue – Queue for which the summary is requested.</li> </ul>	QueueSummary QueueSummaryComplete

The dashboard server listens for the Asterisk AMI events shown in Table 11.

Table 11. Asterisk AMI Events and Descriptions

AMI Action	Description	Syntax	Fields
AgentsComplete	Generated when an agent has finishes servicing a member in the queue	Event: AgentComplete Queue: <value> Member: <value> MemberName: <value> HoldTime: <value> [Variable:] <value> TalkTime: <value> Reason: <value> Queue: <value> Uniqueid: <value> Channel: <value> Member: <value> MemberName: <value> HoldTime: <value>	<ul style="list-style-type: none"> <li>Queue – The name of the queue.</li> <li>Member – The queue member's channel technology or location.</li> <li>MemberName – The name of the queue member.</li> <li>HoldTime – The time the channel was in the queue, expressed in seconds since 00:00, Jan 1, 1970 UTC.</li> <li>Variable – Optional channel variables from the ChannelCalling channel</li> <li>TalkTime – The time the agent talked with the member in the queue, expressed in seconds since 00:00, Jan 1, 1970 UTC.</li> <li>Reason <ul style="list-style-type: none"> <li>– consumer</li> <li>– agent</li> <li>– transfer</li> </ul> </li> <li>Queue</li> <li>Uniqueid</li> <li>Channel</li> <li>Member</li> <li>MemberName</li> <li>HoldTime</li> </ul>

AMI Action	Description	Syntax	Fields
AgentLogin	Generated when an agent logs in	Event: AgentLogin Agent: <value> Channel: <value> Uniqueid: <value>	<ul style="list-style-type: none"><li>• Agent – The name of the agent.</li><li>• Channel</li><li>• Uniqueid</li></ul>
AgentLogoff	Generated when an agent logs off	Event: AgentLogoff Agent: <value> Agent: <value> Logintime: <value> Uniqueid: <value>	<ul style="list-style-type: none"><li>• Agent – The name of the agent.</li><li>• Agent</li><li>• Logintime</li><li>• Uniqueid</li></ul>
FullyBooted	Generated when all Asterisk initialization procedures complete	Event: FullyBooted Status: <value>	<ul style="list-style-type: none"><li>• Status</li></ul>

AMI Action	Description	Syntax	Fields
QueueMemberAdded	Generated when a new member is added to the queue	Event: QueueMemberAdded Queue: <value> Location: <value> MemberName: <value> StateInterface: <value> Membership: <value> Penalty: <value> CallsTaken: <value> LastCall: <value> Status: <value> Paused: <value> Queue: <value> Location: <value> MemberName: <value> StateInterface: <value> Membership: <value> Penalty: <value> CallsTaken: <value> LastCall: <value> Status: <value> Paused: <value>	<ul style="list-style-type: none"> <li>• Queue – The name of the queue.</li> <li>• Location – The queue member's channel technology or location.</li> <li>• MemberName – The name of the queue member.</li> <li>• StateInterface – Channel technology or location from which to read device state changes.</li> <li>• Membership <ul style="list-style-type: none"> <li>– dynamic</li> <li>– realtime</li> <li>– static</li> </ul> </li> <li>• Penalty – The penalty associated with the queue member.</li> <li>• CallsTaken – The number of calls this queue member has serviced.</li> <li>• LastCall – The time this member last took call, expressed in seconds since 00:00, Jan 1, 1970 UTC.</li> <li>• Status – The numeric device state status of the queue member. <ul style="list-style-type: none"> <li>– 0 - AST_DEVICE_UNKNOWN</li> <li>– 1 - AST_DEVICE_NOT_INUSE</li> <li>– 2 - AST_DEVICE_INUSE</li> <li>– 3 - AST_DEVICE_BUSY</li> <li>– 4 - AST_DEVICE_INVALID</li> <li>– 5 - AST_DEVICE_UNAVAILABLE</li> <li>– 6 - AST_DEVICE_RINGING</li> <li>– 7 - AST_DEVICE_RINGINUSE</li> <li>– 8 - AST_DEVICE_ONHOLD</li> </ul> </li> <li>• Paused <ul style="list-style-type: none"> <li>– 0</li> <li>– 1</li> </ul> </li> <li>• Queue</li> <li>• Location</li> <li>• MemberName</li> <li>• StateInterface</li> <li>• Membership</li> <li>• Penalty</li> <li>• CallsTaken</li> <li>• LastCall</li> <li>• Status</li> <li>• Paused</li> </ul>

AMI Action	Description	Syntax	Fields
QueueMemberRemoved	Generated when a queue member is removed from the queue	Event: QueueMemberRemoved Queue: <value> Location: <value> MemberName: <value> Queue: <value> Location: <value> MemberName: <value>	<ul style="list-style-type: none"> <li>• Queue – The name of the queue.</li> <li>• Location – The queue member's channel technology or location.</li> <li>• MemberName – The name of the queue member.</li> <li>• Queue</li> <li>• Location</li> <li>• MemberName</li> </ul>

### 3.4.3 Call Detail Record Dashboard

The Asterisk PBX system is capable of collecting and storing call data for each call that traverses the system. These data records, referred to as CDRs, are collected and stored for use by business intelligence (BI) tools, statistical analysis, and even compensation and reimbursement. The information collected is invaluable and can be retrieved and stored using various methods.

By default, the Asterisk server stores CDRs in a file called master.csv located in /var/log/cdr-csv. CDR records can be collected in real time, or near-real time, in various ways. The Asterisk server also supports different database connections. One method is to use Asterisk on an internal application for connecting to a database. Another is to configure an Open Database Connectivity (ODBC) connection for Asterisk. There is no restriction on the type of relational database used. For example, MySQL, MariaDB, and others are all viable options. CAMH elected to use a MySQL database server with Asterisk connecting through an ODBC connection.

#### 3.4.3.1 MySQL Installation

To start, it is necessary to install additional packages for the MySQL server and the ODBC connector as follows:

```
sudo yum install unixODBC unixODBC-devel libtool-ltdl libtool-ltdl-devel mysql-connector-odbc
wget http://repo.mysql.com/mysql-community-release-el7-5.noarch.rpm
sudo rpm -ivh mysql-community-release-el7-5.noarch.rpm
sudo yum install mysql-server
sudo systemctl start mysqld
sudo yum update
```

Once the packages are installed, it is necessary that Asterisk recognizes these new packages. To do so, migrate to the Asterisk directory, which in our build is located in /usr/src/asterisk-13.8.0/.

From this directory, run:

```
./configure
make menuselect
```



Once the graphic window is displayed, check to make sure that the res\_odbc module is selected. Save and Exit the graphic window.

Next, run the following:

```
make && make install
```

This will configure the Asterisk installation for use with the ODBC connection. Once the installation script finishes successfully, you can proceed to start the sql server.

Once MySQL has been installed, run the basic hardening script for MySQL to remove anonymous connections, the test database, and establish the root password. To do this, run the following script and follow the prompts:

```
sudo /usr/bin/mysql_secure_installation
```

### 3.4.3.2 Database Configuration

After installing MySQL and completing the initial configuration, log into MySQL using:

```
mysql -u root -p
```

Once logged in, create the database and the table needed to store the CDRs. To do so, run the following commands:

```
CREATE DATABASE asterisk;
USE asterisk;

CREATE TABLE `bit_cdr` (
  `calldate` datetime NOT NULL default '0000-00-00 00:00:00',
  `clid` varchar(80) NOT NULL default '',
  `src` varchar(80) NOT NULL default '',
  `dst` varchar(80) NOT NULL default '',
  `dcontext` varchar(80) NOT NULL default '',
  `channel` varchar(80) NOT NULL default '',
  `dstchannel` varchar(80) NOT NULL default '',
  `lastapp` varchar(80) NOT NULL default '',
  `lastdata` varchar(80) NOT NULL default '',
  `duration` int(11) NOT NULL default '0',
  `billsec` int(11) NOT NULL default '0',
  `disposition` varchar(45) NOT NULL default '',
  `amaflags` int(11) NOT NULL default '0',
  `accountcode` varchar(20) NOT NULL default '',
  `userfield` varchar(255) NOT NULL default '',
  `uniqueid` VARCHAR(32) NOT NULL default '',
  `linkedid` VARCHAR(32) NOT NULL default '',
  `sequence` VARCHAR(32) NOT NULL default '',
  `peeraccount` VARCHAR(32) NOT NULL default '' );
```

```
ALTER TABLE `bit_cdr` ADD INDEX ( `calldate` );
ALTER TABLE `bit_cdr` ADD INDEX ( `dst` );
ALTER TABLE `bit_cdr` ADD INDEX ( `accountcode` );
```

To create a user account for remote connections, use the following syntax:

```
CREATE USER 'username'@'%' IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON *.* TO 'username'@'%' WITH GRANT OPTION;
FLUSH PRIVILEGES;
```

### 3.4.3.3 Asterisk Integration

Now that the database is up and running, connect to Asterisk by using the ODBC connector, as follows:

```
vi /etc/odbcinst.ini
[MySQL]
Description      = ODBC for MySQL
Driver           = /usr/lib/libmyodbc5.so
Setup            = /usr/lib/libodbcmyS.so
Driver64         = /usr/lib64/libmyodbc5.so
Setup64          = /usr/lib64/libodbcmyS.so
FileUsage        = 1

vi /etc/odbc.ini
[asterisk-connector]
Description = MySQL connection to 'asterisk' database
Driver = MySQL
Database = asterisk
Server = localhost
User = root
Password = password
Port = 3306
Socket = /var/lib/mysqld/mysqld.sock
```

To test your configuration, run the following command from the CLI:

```
echo "select 1" | isql -v asterisk-connector
```

You should see a message of Connected! returned.

```
| Connected! |
SQLRowCount returns 1 1 rows fetched
```

Now you will need to point Asterisk to your new database. To do so, modify the `/etc/asterisk/res_odbc.conf` file with the database name, username, password and port of the ODBC connection you have set up. The dsn variable will point to the dsn identified:

```
vi /etc/asterisk/res_odbc.conf

[asterisk]
enabled => yes
dsn => asterisk-connectorusername => username
password => password
pooling => no
limit => 99999
pre-connect => yes
```

Next, edit /etc/asterisk/cdr\_odbc.conf to include the following:

```
vi /etc/asterisk/cdr_odbc.conf

[global]
dsn=asterisk
loguniqueid=yes
table=bit_cdr
```

**Note:** the dsn variable will point to the DSN identified in res\_odbc.conf, NOT /etc/odbc.ini.

Next, Edit /etc/ cdr\_adaptive\_odbc.conf to include the following:

```
vi /etc/asterisk/cdr_adaptive_odbc.conf
[asteriskcdr]
connection=asterisk
table=cdr
alias start=calldate
```

Next, ensure CDR writing is enabled in cdr\_manager.conf:

```
vi /etc/asterisk/cdr_manager.conf
[general]
enabled = yes
```

Finally, restart Asterisk:

```
sudo service asterisk restart
```

### 3.4.3.4 Node.js Integration

The CDR reporting functionality relies on two Node.js servers: the management portal (server-db.js) and the acr-cdr (app.js). These servers provide the following capabilities:

- server-db.js
  - Receives GET call for /cdrinfo.
  - Performs GET call to app.js /getallcdrrecs.
  - Acts as a middleman for the cdr.html page to the app.js node server. This node server can be bypassed by changing the cdr.html GET call from /cdrinfo to http://<app.js

location>/getallcdrrecs if the CDR Dashboard needs to be separated from the Management Portal.

- app.js
  - Receives GET call for /getallcdrrecs.
  - Performs query to the Asterisk CDR database table. Returns a JSON object of the data.

## 3.5 Agent / Agent Database (Provider)

The ACE Direct uses a database to store Agent information and to verify Agent identity when an Agent logs into the system.

To host the agent data, a MySQL database server is required. A RESTful API developed using Node.js provides access to the data for Agent verification. For this effort, CAMH built the Node.js server with the following software versions:

- Operating System: CentOS 7.x
- Node.js: v7.2.1

### 3.5.1 MySQL Database Server Configuration

The CAMH team developed a MySQL database containing several tables to store the agent-related data. Create a database and use the dat/acedirectdefault.sql MySQL script found in <https://github.com/FCC/ACEDirect> to create the application tables.

Verify that the MySQL database is up, running, and accepting connections. A tool like MySQL Workbench can quickly verify that the configuration is correct. The dat/acedirectdefault.sql script pre-populates the database with default agent data; however, you must first modify dat/acedirectdefault.sql to have actual values for the EXTENSION\_PASSWORD, \_ASTERISK\_PASSWORD\_, and \_ACEDIRECT\_PASSWORD\_ placeholders. Remember to update dat/config.json to include your actual database name, users, and passwords.

## 3.6 Video Relay Service User Database (Provider)

The VRS user database was developed to emulate a VRS user lookup in the iTRS-URD from the Agent desktop until full access to the iTRS-URD is obtained. This lookup verifies that the Consumer-provided phone number is a registered VRS number. A MySQL database server is required. A RESTful API developed using Node.js provides access to the data for Consumer verification. For this effort, CAMH built the Node.js server with the following software versions:

- Operating System: CentOS 7.x
- Node.js: v7.2.1

### 3.6.1 MySQL Database Server Configuration

The CAMH team developed a MySQL database containing a single table to store the emulated VRS lookup data. The dat/acedirectdefault.sql script pre-populates the database with default VRS data.

Verify that the MySQL database is up, running, and accepting connections. A tool like MySQL Workbench can quickly verify that the configuration is correct.

## 3.7 STUN Server

If a host is located behind a NAT, then in certain situations it can be impossible for that host to communicate directly with other hosts (peers). In these situations, the host must use the services of an intermediate node as a communication relay. This specification defines a protocol, called TURN (Traversal Using Relays around NAT), which allows the host to control the operation of the relay and to exchange packets with its peers using the relay. TURN differs from other relay control protocols because it allows a client to communicate with multiple peers using a single relay address. A TURN server, which is an implementation of the STUN protocol, uses a relay to provide an alternate method for NAT discovery and traversal (STUN). TURN can traverse symmetric NAT instances.

### 3.7.1 Installation

For CentOS, [turnserver](#) was used for STUN/TURN services. Before installing turnserver, ensure the dependencies are installed:

```
sudo yum -y install
sudo yum install -y make gcc cc gcc-c++ wget openssl-devel
libevent libevent-devel mysql-devel mysql-server
```

Then install the LibEvent modules, a dependency of turnserver:

```
stable.tar.gz
tar xvfz libevent-2.0.21-stable.tar.gz
cd libevent-2.0.21-stable && ./configure
sudo make && sudo make install && cd ..
```

Finally, install turnserver:

```
wget http://turnserver.open-
sys.org/downloads/v3.2.3.8/turnserver-3.2.3.8.tar.gz
tar -xvzf turnserver-3.2.3.8.tar.gz
cd turnserver-3.2.3.8 && ./configure
sudo make && sudo make install
```

If desired, you may configure a turnserver config file to add user credentials, and the address and port to listen on:

```
mkdir /etc/turnserver
vi /etc/turnserver/turnserver.conf

# setting static accounts
# Remember, "static" accounts are not dynamically checked by the
turnserver process.
user=username:password
```

```
# listen ports
listening-port=<port>
listening-ip=<local_ip>
```

Start the turnserver process. Use the first command to implement the config file, and the second not to:

```
nohup turnserver -v -r ip:port -z -c
/etc/turnserver/turnserver.conf &
nohup turnserver -L <local_IP> -v -z --min-port 10000 --max-port
20000 -n &
```

### 3.7.2 System Startup

You can enable turnserver to start on system boot if desired. The following instructions have been validated on CentOS servers, and may or may not work on other Linux distributions. To do so, first create the following script, make it executable, and save it as `/etc/rc.d/init.d/turnserver`:

```
#!/bin/bash
# chkconfig: 2345 20 80
# description: Manage turnserver as a system service so it starts
on boot

# Source function library
. /etc/init.d/functions

# This variable will be used to define what IP address turnserver
listens on
# You may need to change this depending on how your network
service is configured
HOST=$(hostname -I | awk '{print $1}')

# This is the port that STUN will listen on.
PORT=3478

start() {
    /usr/local/bin/turnserver -L $HOST -p $PORT -v -z --min-
port 10000 --max-port 20000 -n
    echo "STUN has started successfully"
}

stop() {
    killall -9 turnserver
    echo "STUN server stopped successfully"
}

case $1 in
```

```

        start)
            start
            ;;
        stop)
            stop
            ;;
        restart)
            stop
            start
            ;;
        status)
            STATUS=$(netstat -tanp | grep turnserver | grep
$PORT)
            if [ ${#STATUS} == 0 ]
            then
                echo "STUN server is not currently
running"
            else
                echo "STUN server is currently running"
                echo $STATUS
            fi
            ;;
        *)
            echo "Usage: service turnserver
(start|stop|restart|status)"
            exit 1
            ;;
    esac
    exit 0

```

Then, create the following systemd configuration file and save it as `/usr/lib/systemd/system/turnserver.service`:

```

[Unit]
Description=Turnserver Service
After=network-online.target
Wants=network-online.target

[Service]
Type=simple
ExecStart=/etc/rc.d/init.d/turnserver start
;ExecStop=/root/start-turn.sh stop
;ExecReload=/root/start-turn.sh restart

[Install]
WantedBy=default.target

```

Finally, reload the systemctl daemon, enable the turnserver service in systemctl, then reboot the server and use the 'netstat' command to confirm that turnserver is now starting upon boot:

```
systemctl daemon-reload
systemctl enable turnserver.service
reboot now
```

## 3.8 iTRS ENUM Database

For phone numbers of deaf and hard-of-hearing users, iTRS is the authoritative database managed by Neustar. The lookup of the iTRS ENUM database performs DNS queries to determine a Uniform Resource Identifier (URI) for a 10-digit telephone number. There is a GUI as well as a programmatic interface. **Note: Access to the iTRS ENUM database requires permission from the FCC.**

To query the production iTRS database requires establishing an IPsec tunnel with Neustar. For the proof of concept, the CAMH team added the IP address 156.154.59.67 first in the DNS settings (/etc/resolv.conf) for the ENUM lookup to work. Note that with the default AWS instance settings, any changes made to the resolv.conf file will be overwritten on restart of the network service because new values are queries from DNS. This behavior must be disabled.

The following example snippet of code, which is part of an Asterisk Dial Plan, comes from an extensions.conf file. Subsection 3.2.2.1 provides more information on the Asterisk Dial Plan.

```
exten => _9.[1-9]XXXXXXXXXX, 1,
Set(sipuri=${ENUMLOOKUP(+${EXTEN:1},sip,,1,itrs.us)})
same => n,NoOp("Outbound Direct Video Call to: ${EXTEN:1}") ;
just for informational purposes

same => s,n,SipAddHeader(P-Asserted-Identity: <sip:nnnnnnnnnn>)
;set the callerID number

same => n,NoOp("sipuri: ${sipuri:1}")

same => n,Dial(SIP/${sipuri:1},30)
```

## 3.9 StrongSWAN for Secure Socket Layer Tunnel

StrongSWAN is an open source VPN software that is widely popular within the IPsec industry. StrongSWAN can be installed on both CentOS and Amazon Linux servers; the following instructions were implemented on an Amazon Linux EC2 instance.

In the following installation scenario, the local side of the VPN is running within AWS. The remote side provider requires both a public IP for the VPN endpoint and for the tunneled traffic. The implementation requires two public IP addresses on the local side, and NATs all local traffic to one of those address for the VPN tunnel. Also, on the remote end, there is only one accessible IP address. Minor adjustments would be required to allow access to more than one destination IP address or to allow a range of IP addresses directly (with or without NAT) through the VPN.

All the following commands must be run as root (or via sudo).



### 3.9.1 Installation

First, use yum to install StrongSWAN:

```
yum install -y strongswan
```

Then, modify the /etc/strongswan/ipsec.conf table to reflect the following:

```
config setup
    #charondebug="ike 2, net 3, knl 2, cfg 2"      #useful
debugs
conn %default
    ikelifetime=480m
    keylife=60m
    rekeymargin=3m
    keyingtries=1
    keyexchange=ikev1
    authby=secret
conn PeerProvider
    auto=start
    ike=aes128-sha1-modp1024          #P1: modp1536 = DH group
2
    esp=aes128-sha1-modp1024          #P2

    left=<StrongSWAN local address>    #Local outside address
    leftsubnet=<dummy address>/32      #network behind Local
    leftid=<StrongSWAN public IP>      #IKEID sent by Local
    leftfirewall=yes

    right=<remote peer address.        #PeerProvider outside
address
    rightsubnet=<remote server address>/32 #network behind
PeerProvider
    rightid=<remote peer address.      #IKEID sent by
PeerProvider
```

Agree with your peer on a shared key and then modify the /etc/strongswan/ipsec.secrets file to reflect the following:

```
<StrongSWAN public IP> <remote server address> : PSK "<Key
obtained from PeerProvider in quotes>"
```

IPTables must be configured to perform NAT between the source subnet and the public IP identified for tunnel traffic. Create the `/etc/iptables.conf` file and add the following:

```
*nat
:PREROUTING ACCEPT [5:436]
:INPUT ACCEPT [1:92]
:OUTPUT ACCEPT [34:9996]
:POSTROUTING ACCEPT [34:9996]
-A POSTROUTING -s <VPC CIDR block>/24 -d <remote server
address>/32 -o eth0 -j SNAT --to-source <dummy address>
COMMIT

*filter
:INPUT ACCEPT [1063:95316]
:FORWARD ACCEPT [12:1032]
:OUTPUT ACCEPT [1018:375057]
COMMIT
```

The server needs a dummy interface associated with the public IP for tunnel traffic, and the IPTables configuration must be loaded at startup. Append the following lines to `/etc/rc.d/rc.local`:

```
/sbin/modprobe dummy
/sbin/ifconfig dummy0 <dummy address> netmask 255.255.255.0
/sbin/iptables-restore < /etc/iptables.conf
```

Move into the `/etc/rc3.d` directory and create the following symbolic links to ensure that StrongSWAN gracefully starts/stops when the EC2 instance is rebooted:

```
cd /etc/rc3.d
ln -s ../init.d/strongswan S48strongswan
ln -s ../init.d/strongswan K52strongswan
```

Routing must be enabled on the server. Modify the variable within the `/etc/sysctl.conf` file to the following:

```
net.ipv4.ip_forward = 1
```

Create the following script in `/root/scripts` to automatically restart StrongSWAN if ITRS queries fail:

```
#!/bin/bash
```

```
DEBUG=true
LOG=/var/log/itrsmon.log

if ! dig @<remote server address> in naptr
9.6.8.4.1.5.5.7.2.7.1.itrs.us>/dev/null 2>&1
then
    echo $(/bin/date) " - ITRS Lookup failed" >> $LOG
    /etc/init.d/strongswan restart
else
    $DEBUG && echo $(/bin/date) " - ITRS Lookup success" >> $LOG
fi
```

Add the following line to the crontab to monitor StrongSWAN by running the script once per minute:

```
$ crontab -e

* * * * * /root/scripts/itrs_mon.sh
```

Finally, reboot the instance to verify that the StrongSWAN service is running (using the ‘strongswan status’ command). Once the service is running, you can move onto the following "AWS Specific Config" section if you are in AWS. When you have completed those steps, your DNS queries to the iTRS database from Asterisk servers should be successful. An example of a successful iTRS query is as follows:

```
$ dig @<remote server address> in naptr
1.1.1.1.1.1.1.1.1.1.itrs.us

; <<>> DiG 9.9.4-RedHat-9.9.4-50.el7_3.1 <<>> @<remote server
address> in naptr 1.1.1.1.1.1.1.1.1.1.itrs.us
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 32364
;; flags: qr aa rd ad; QUERY: 1, ANSWER: 1, AUTHORITY: 0,
ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
```

```

; EDNS: version: 0, flags:; udp: 5120
;; QUESTION SECTION:
;1.1.1.1.1.1.1.1.1.1.1.1.1.itrs.us. IN NAPTR

;; ANSWER SECTION:
1.1.1.1.1.1.1.1.1.1.1.1.1.itrs.us. 900 IN NAPTR 10 1 "u" "E2U+h323"
"!^(.*)$!h323:\l@192.168.1.1!" .

;; Query time: 15 msec
;; SERVER: <remote server address>#53(<remote server address>)
;; WHEN: Fri Aug 04 14:40:59 UTC 2017
;; MSG SIZE rcvd: 115

```

## 3.9.2 AWS-Specific Configuration

### 3.9.2.1 VPC Route Table

A route must be created in the VPC route table to ensure that traffic to the iTRS database is routed to the VPN instance (which in turn routes it out over the VPN tunnel).

### 3.9.2.2 Source/Destination Checking

By default, instances drop packets when source and destination information do not match that of an instance. You can disable this behavior from the AWS console by selecting an instance and going to network options.

## 3.9.3 Troubleshooting

You can run 'strongswan statusall' to view the status of the VPN tunnel. The expected command output will look like the following:

```

$ strongswan statusall

Status of IKE charon daemon (strongSwan 5.4.0, Linux
4.9.38-16.33.amzn1.x86_64, x86_64):
  uptime: 116 minutes, since Aug 04 12:46:06 2017
  malloc: sbrk 1622016, mmap 0, used 502608, free 1119408
  worker threads: 11 of 16 idle, 5/0/0/0 working, job queue:
0/0/0/0, scheduled: 2
  loaded plugins: charon aes des rc2 sha2 sha1 md4 md5
random nonce x509 revocation constraints acert pubkey pkcs1
pkcs8 pkcs12 pgp dnskey sshkey pem openssl gcrypt fips-prf
gmp xcbc cmac hmac ctr ccm gcm curl attr kernel-netlink
resolve socket-default farp stroke vici updown eap-identity
eap-md5 eap-gtc eap-mschapv2 eap-tls eap-ttls eap-peap
xauth-generic xauth-eap xauth-pam xauth-noauth dhcp

```

**Listening IP addresses:**

<StrongSWAN local address>

<dummy address> ← **both IP addresses should be listed**

**Connections:**

PeerProvider: <StrongSWAN local address>...<remote peer address. IKEv1

PeerProvider: local: [<StrongSWAN public address>] uses pre-shared key authentication

PeerProvider: remote: [<remote peer address.>] uses pre-shared key authentication

PeerProvider: child: <dummy address>/32 === <remote server address>/32 TUNNEL

Security Associations (1 up, 0 connecting):

**PeerProvider[1]: ESTABLISHED 116 minutes ago**, <StrongSWAN local address>[<StrongSWAN public address>]...<remote peer address.>[<remote peer address.>]

PeerProvider[1]: IKEv1 SPIs: <<SPI>>, pre-shared key reauthentication in 21 hours

PeerProvider[1]: IKE proposal:

AES\_CBC\_128/HMAC\_SHA1\_96/PRF\_HMAC\_SHA1/MODP\_1024

**PeerProvider{3}: INSTALLED, TUNNEL**, reqid 1, ESP SPIs: <<SPI>>

PeerProvider{3}: AES\_CBC\_128/HMAC\_SHA1\_96/MODP\_1024, 1494 bytes\_i (8 pkts, 2s ago), 611 bytes\_o (8 pkts, 2s ago), rekeying in 47 minutes

PeerProvider{3}: <dummy address>/32 === <remote server address>/32

To find system messages related to StrongSWAN, use the following grep command:

```
$ grep charon /var/log/messages
```

**#Shutdown**

```
Jul 27 00:27:58 ServerName charon: 00[DMN] signal of type SIGINT received. Shutting down
```

```
Jul 27 00:27:58 ServerName charon: 00[IKE] closing CHILD_SA PeerProvider{1} with SPIs <<SPI>> (0 bytes) 5401197d_o (0 bytes) and TS <Dummy Address>/32 === <remote server address>/32
```

```
Jul 27 00:27:58 ServerName charon: 00[IKE] sending DELETE for ESP CHILD_SA with SPI cc95ab50
```

```
Jul 27 00:27:58 ServerName charon: 00[ENC] generating INFORMATIONAL_V1 request 4051018568 [ HASH D ]
```

```
Jul 27 00:27:58 ServerName charon: 00[NET] sending packet: from <StrongSWAN public IP>[500] to <remote peer address.>[500] (76 bytes)
```

```
Jul 27 00:27:58 ServerName charon: 00[IKE] deleting IKE_SA PeerProvider[1] between <StrongSWAN public IP>[<local server address>]...<remote peer address.>[<remote peer address.>]
```

```
Jul 27 00:27:58 ServerName charon: 00[IKE] sending DELETE for IKE_SA PeerProvider[1]
```

```
Jul 27 00:27:58 ServerName charon: 00[ENC] generating
INFORMATIONAL_V1 request 3332107879 [ HASH D ]
Jul 27 00:27:58 ServerName charon: 00[NET] sending packet:
from <StrongSWAN public IP>[500] to <remote peer
address.[500] (92 bytes)
Jul 27 00:27:58 ServerName charon: 00[KNL] received netlink
error: Address family not supported by protocol (97)
```

### **#Startup**

```
Jul 27 00:27:58 ServerName charon: 00[DMN] Starting IKE
charon daemon (strongSwan 5.4.0, Linux 4.9.32-
15.41.amzn1.x86_64, x86_64)
Jul 27 00:27:58 ServerName charon: 00[LIB] openssl FIPS
mode(2) - enabled
```

### **#No IPv6 - that's OK**

```
Jul 27 00:27:58 ServerName charon: 00[NET] could not open
socket: Address family not supported by protocol
Jul 27 00:27:58 ServerName charon: 00[NET] could not open
IPv6 socket, IPv6 disabled
Jul 27 00:27:58 ServerName charon: 00[KNL] received netlink
error: Address family not supported by protocol (97)
Jul 27 00:27:58 ServerName charon: 00[KNL] unable to create
IPv6 routing table rule
```

### **#No certs - that's OK**

```
Jul 27 00:27:58 ServerName charon: 00[CFG] loading ca
certificates from '/etc/strongswan/ipsec.d/cacerts'
Jul 27 00:27:58 ServerName charon: 00[LIB] opening
directory '/etc/strongswan/ipsec.d/cacerts' failed: No such
file or directory
Jul 27 00:27:58 ServerName charon: 00[CFG] reading
directory failed
Jul 27 00:27:58 ServerName charon: 00[CFG] loading aa
certificates from '/etc/strongswan/ipsec.d/aacerts'
Jul 27 00:27:58 ServerName charon: 00[LIB] opening
directory '/etc/strongswan/ipsec.d/aacerts' failed: No such
file or directory
Jul 27 00:27:58 ServerName charon: 00[CFG] reading
directory failed
Jul 27 00:27:58 ServerName charon: 00[CFG] loading ocs
signer certificates from
'/etc/strongswan/ipsec.d/ocspcerts'
Jul 27 00:27:58 ServerName charon: 00[LIB] opening
directory '/etc/strongswan/ipsec.d/ocspcerts' failed: No
such file or directory
Jul 27 00:27:58 ServerName charon: 00[CFG] reading
directory failed
Jul 27 00:27:58 ServerName charon: 00[CFG] loading
attribute certificates from
'/etc/strongswan/ipsec.d/acerts'
Jul 27 00:27:58 ServerName charon: 00[LIB] opening
```

```
directory '/etc/strongswan/ipsec.d/acerts' failed: No such
file or directory
Jul 27 00:27:58 ServerName charon: 00[CFG] reading
directory failed
Jul 27 00:27:58 ServerName charon: 00[CFG] loading crls
from '/etc/strongswan/ipsec.d/crls'
Jul 27 00:27:58 ServerName charon: 00[LIB] opening
directory '/etc/strongswan/ipsec.d/crls' failed: No such
file or directory
Jul 27 00:27:58 ServerName charon: 00[CFG] reading
directory failed
```

### **#Interesting stuff starts here - loading secrets**

```
Jul 27 00:27:58 ServerName charon: 00[CFG] loading secrets
from '/etc/strongswan/ipsec.secrets'
Jul 27 00:27:58 ServerName charon: 00[CFG] loaded IKE
secret for <local server address> <remote server address>
Jul 27 00:27:58 ServerName charon: 00[LIB] loaded plugins:
charon aes des rc2 sha2 sha1 md4 md5 random nonce x509
revocation constraints acert pubkey pkcs1 pkcs8 pkcs12 pgp
dnskey sshkey pem openssl gcrypt fips-prf gmp xcbc cmac
hmac ctr ccm gcm curl attr kernel-netlink resolve socket-
default farp stroke vici updown eap-identity eap-md5 eap-
gtc eap-mschapv2 eap-tls eap-ttls eap-peap xauth-generic
xauth-eap xauth-pam xauth-noauth dhcp
Jul 27 00:27:58 ServerName charon: 00[JOB] spawning 16
worker threads
Jul 27 00:27:58 ServerName charon: 05[CFG] received stroke:
add connection 'PeerProvider'
Jul 27 00:27:58 ServerName charon: 05[CFG] added
configuration 'PeerProvider'
Jul 27 00:27:58 ServerName charon: 09[CFG] received stroke:
initiate 'PeerProvider'
Jul 27 00:27:58 ServerName charon: 09[IKE] initiating Main
Mode IKE_SA PeerProvider[1] to <remote peer address>.
Jul 27 00:27:58 ServerName charon: 09[ENC] generating
ID_PROT request 0 [ SA V V V V ]
```

### **#Sending and RECEIVING packets - good sign**

```
Jul 27 00:27:58 ServerName charon: 09[NET] sending packet:
from <StrongSWAN public IP>[500] to <remote peer
address.[500] (228 bytes)
Jul 27 00:27:58 ServerName charon: 15[NET] received packet:
from <remote peer address.[500] to <StrongSWAN public
IP>[500] (164 bytes)
Jul 27 00:27:58 ServerName charon: 15[ENC] parsed ID_PROT
response 0 [ SA V V V V ]
Jul 27 00:27:58 ServerName charon: 15[ENC] received unknown
vendor ID:
05:16:dc:8a:88:2c:54:a5:66:90:dc:05:bd:da:3b:9e:c8:05:e5:86
```

```
:12:00:00:00:1e:06:00:00
Jul 27 00:27:58 ServerName charon: 15[IKE] received DPD
vendor ID
Jul 27 00:27:58 ServerName charon: 15[ENC] received unknown
vendor ID:
48:65:61:72:74:42:65:61:74:5f:4e:6f:74:69:66:79:38:6b:01:00
Jul 27 00:27:58 ServerName charon: 15[ENC] generating
ID_PROT request 0 [ KE No ]
Jul 27 00:27:58 ServerName charon: 15[NET] sending packet:
from <StrongSWAN public IP>[500] to <remote peer
address.[500] (196 bytes)
Jul 27 00:27:58 ServerName charon: 11[NET] received packet:
from <remote peer address.[500] to <StrongSWAN public
IP>[500] (196 bytes)
Jul 27 00:27:58 ServerName charon: 11[ENC] parsed ID_PROT
response 0 [ KE No ]
Jul 27 00:27:58 ServerName charon: 11[ENC] generating
ID_PROT request 0 [ ID HASH N(INITIAL_CONTACT) ]
Jul 27 00:27:58 ServerName charon: 11[NET] sending packet:
from <StrongSWAN public IP>[500] to <remote peer
address.[500] (108 bytes)
Jul 27 00:27:58 ServerName charon: 07[NET] received packet:
from <remote peer address.[500] to <StrongSWAN public
IP>[500] (76 bytes)
Jul 27 00:27:58 ServerName charon: 07[ENC] parsed ID_PROT
response 0 [ ID HASH ]
#Phase 1 (IKE) Established
Jul 27 00:27:58 ServerName charon: 07[IKE] IKE_SA
PeerProvider[1] established b <StrongSWAN public IP>[<local
server address>]...<remote peer address.[<remote peer
address.]
Jul 27 00:27:58 ServerName charon: 07[IKE] scheduling
reauthentication in 86056s
Jul 27 00:27:58 ServerName charon: 07[IKE] maximum IKE_SA
lifetime 86236s
Jul 27 00:27:58 ServerName charon: 07[ENC] generating
QUICK_MODE request 688951572 [ HASH SA No KE ID ID ]
Jul 27 00:27:58 ServerName charon: 07[NET] sending packet:
from <StrongSWAN public IP>[500] to <remote peer
address.[500] (316 bytes)
Jul 27 00:27:58 ServerName charon: 13[NET] received packet:
from <remote peer address.[500] to <StrongSWAN public
IP>[500] (316 bytes)
Jul 27 00:27:58 ServerName charon: 13[ENC] parsed
QUICK_MODE response 688951572 [ HASH SA No KE ID ID ]
#Phase 2 (ISAKMP) Established
Jul 27 00:27:58 ServerName charon: 13[IKE] CHILD_SA
PeerProvider{1} established with SPIs X and TS <Dummy
Address>/32 == <remote server address>/32
Jul 27 00:27:58 ServerName charon: 13[ENC] generating
QUICK_MODE request 688951572 [ HASH ]
Jul 27 00:27:58 ServerName charon: 13[NET] sending packet:
```



```
from <StrongSWAN public IP>[500] to <remote peer
address.[500] (60 bytes)
```

You can verify the proper configuration of the iptables rule to mask the source IP with the dummy address. You should see the number of packets that have been applied to that rule:

```
$ iptables -t nat -L -v

Chain PREROUTING (policy ACCEPT 804 packets, 66289 bytes)
  pkts bytes target      prot opt in      out     source
  destination

Chain INPUT (policy ACCEPT 43 packets, 4165 bytes)
  pkts bytes target      prot opt in      out     source
  destination

Chain OUTPUT (policy ACCEPT 51100 packets, 3555K bytes)
  pkts bytes target      prot opt in      out     source
  destination

Chain POSTROUTING (policy ACCEPT 51076 packets, 3553K bytes)
  pkts bytes target      prot opt in      out     source
  destination
    785 63924 SNAT          all  --  any    eth0    <<local subnet>>/24
    <remote server address>          to:<Dummy Address>
```

You can use the 'ip route' command to ensure StrongSWAN has properly configured the route on the instance required to forward traffic into the VPN tunnel. The second and fourth lines of the following output are of particular importance:

```
$ ip route show table all
default via ...
<<Remote Address>>/24 dev dummy0 proto kernel scope link
src <Dummy Address>
... dev eth0
<local subnet> dev eth0 proto kernel scope link src
<StrongSWAN public IP>
broadcast .. dev dummy0 table local proto kernel scope
link src <Dummy Address>
<< deleted the rest of the output >>
```

## 3.10 Commercial Customer Relationship Management

To demonstrate integration with a commercial CRM service, ACE Direct connects to the Zendesk portal via the ESB. ACE Direct sends JSON-based messages to the RESTful Zendesk API to manage and query customer records. Zendesk is a suite of web-based products that help companies provide better customer service. ACE Direct uses Zendesk to capture Consumer complaints. When a Consumer files a complaint, the ACE Direct software uses the Zendesk web service API to create and store a complaint ticket. This API also allows querying and updating of created tickets. When an ACE Direct Agent answers a Consumer call, the application requests the ticket information from Zendesk and displays it on the ACE Direct Agent Desktop portal.

## 3.11 FenDesk Customer Relationship Management

FenDesk is a server that emulates the [Zendesk](#) ticketing system for ACE Direct or any client that needs a simple ticketing system. The software only implements the subset of Zendesk RESTful API calls that ACE Direct uses; however, it is expandable to include other API calls.

FenDesk uses a simple storage scheme. It creates, updates, and returns tickets as simple JSON text files. The filename for a ticket follows the same naming convention as Zendesk: <ticketno>.json (e.g., 322.json). FenDesk offers RESTful API calls to test connectivity, add/update/delete/retrieve tickets, and search for all tickets with a specified VRS number.

## 3.12 Enterprise Service Bus

The ESB provides a generic method to integrate with legacy database systems as well as the diverse number of databases and unstructured data repositories on the market and in use today. ACE Direct ESB integrates with a COTS CRM service (e.g., Zendesk) as a ticketing system for the Agent to document service cases.

### 3.12.1 Background

Apache ServiceMix 6.1.2 is used as the service broker. Apache ServiceMix is an enterprise-class, open source, distributed ESB based on the service-oriented architecture (SOA) model. It is a project of the Apache Software Foundation and was built on the semantics and APIs of the Java Business Integration (JBI) specification JSR 208. The software is distributed under the Apache License.

The current version of ServiceMix fully supports the OSGi framework. ServiceMix is lightweight and easily embeddable, and has integrated Spring Framework support. It can be run at the edge of the network (inside a client or server), as a standalone ESB provider, or as a service within another ESB. ServiceMix is compatible with Java SE or a Java Enterprise Edition (EE) application server. ServiceMix uses ActiveMQ to provide remoting, clustering, reliability, and distributed failover. The basic frameworks used by ServiceMix are Spring and XBean.

ServiceMix comprises the latest versions of Apache ActiveMQ, Apache Camel, Apache CXF, and Apache Karaf. Additional installation features include:

- BPM engine via Activiti
- JPA support via Apache OpenJPA
- XA transaction management via JTA via Apache Aries

The ServiceMix ESB provides:

- Federation, clustering, and container-provided failover
- Hot deployment and life-cycle management of business objects
- Vendor independence from vendor-licensed products

- Compliance with the JBI specification JSR 208
- Compliance with the OSGi 4.2 specification through Apache Felix
- Support for OSGi Enterprise through Apache Aries

## 3.12.2 Installation Overview

First install and configure ServiceMix and its prerequisites on the host machine.

### 3.12.2.1 ServiceMix System Requirements

To run Apache ServiceMix itself, you will need Java Runtime Environment (JRE) 1.8.x (Java 8), and about 100 MB of free disk space for the default assembly.

If you are developing your own integration applications and OSGi bundles, you will also need:

- Java Developer Kit (JDK) 1.8.x (Java 8)
- MySQL
- Apache Maven 3.0.4 or higher

The ACRDEMO broker application depends on MySQL for a database and Maven for building the application.

### 3.12.2.2 Installing the JDK

Issue the following commands to install the Java 8 JDK:

```
Redhat/Fedora/CentOS Systems (SystemV):  
sudo yum install java-1.8.0-openjdk
```

Set the JAVA\_HOME environment variable in the bash startup:

```
vi ~/.bashrc
```

Add the following lines to the end of the .bashrc script:

```
JAVA_HOME= /opt/jdk1.8.0_111  
export JAVA_HOME  
JRE_HOME=/opt/jdk1.8.0_111/jre  
export JRE_HOME  
PATH=$PATH:/opt/jdk1.8.0_111/bin:/opt/jdk1.8.0_111/jre/bin  
export PATH
```

### 3.12.2.3 Installing Apache Maven

To install Apache Maven, issue the following command:

Redhat/Fedora/Centos Systems (SystemV):

```
sudo yum install maven
```

### 3.12.2.4 Installing MySQL

You will be able to set the password for the root account. **Note:** There is a current issue with the ESB that also requires setting the privileges for anonymous local users; accordingly, do not disable access for anonymous users. To install MySQL, issue the following commands:

Redhat/Fedora/Centos Systems (SystemV):

```
sudo yum install unixODBC unixODBC-devel libtool-ltdl
libtool-ltdl-devel mysql-connector-odbc
wget http://repo.mysql.com/mysql-community-release-el7-
5.noarch.rpm
sudo rpm -ivh mysql-community-release-el7-5.noarch.rpm
sudo yum install mysql-server
sudo systemctl start mysqld
sudo yum update
```

### 3.12.2.5 Configuring MySQL

The ServiceMix broker application for the ACR demo connects to the MySQL database; therefore, first create and configure the demo database.

Login to the MySQL command-line tool using the root account with the password you set earlier:

```
mysql -u root -p=somepassword
```

Create a database named “broker” for the ACR demo:

```
mysql> CREATE DATABASE broker;
mysql> USE broker;
```

Create the database user named “broker” for the ACR demo and set the password:

```
mysql> CREATE USER 'broker'@'localhost' IDENTIFIED BY
'somepassword';
```

Set the permissions for the database user “broker”. **Note:** This should be tuned to only grant the necessary privileges:

```
mysql> GRANT ALL PRIVILEGES ON broker.* TO 'broker'@'%' WITH
GRANT OPTION;
```

**Note:** There is a current issue with the ESB that also requires setting the privileges for anonymous local users. Privileges must be set for anonymous users:

```
mysql> GRANT ALL PRIVILEGES ON broker.* TO '@'localhost' WITH
GRANT OPTION;
```

Create the “users” table:

```
mysql> CREATE TABLE `users` (
  `user_id` bigint(20) NOT NULL,
  `user_name` varchar(50) DEFAULT NULL,
  `user_description` varchar(45) DEFAULT NULL,
  `user_phone` varchar(20) DEFAULT NULL,
  `user_address` varchar(50) DEFAULT NULL,
  `user_account` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`user_id`));
```

Populate the “users” table with test records. **Note:** The user\_id values need to correspond to IDs of Zendesk users:

```
mysql> INSERT INTO `users` VALUES
(3770168798,'John Doe ','Some Details','555-555-
1111',NULL,'121212'),
(4060741111,'Jane Doe','No Description','222-111-1111','12341
Main Street','12345671'),
(4758821111,'Tim','No Description','555-666-7777','','5656565');
```

### 3.12.2.6 Downloading and Building Broker Application

Set up SSH keys to access the git repository according to these instructions:

- [Adding a new SSH key to your GitHub account](#)

Create a folder for cloning the broker source code and navigate to that folder:

```
mkdir ~/code && cd ~/code
```

Clone the broker git repository from esb.git.

Navigate to the broker code folder:

```
cd camel-rest-proxy-blueprint/
```

Modify the applications blueprint file for your environment. The blueprint is configured to process messages intended for Zendesk. You will need to modify the blueprint to specify your

Zendesk hostname and any proxy settings if you are accessing Zendesk from the ESB through a proxy.

Build the broker application with Maven:

```
mvn clean install
```

### 3.12.2.7 Downloading and Installing Apache ServiceMix

Apache ServiceMix 6.1.2 is available under the Apache License v2 and can be downloaded from <http://servicemix.apache.org/downloads/servicemix-6.1.2.html>.

Create and navigate to a folder where the downloaded zip file will be placed:

```
mkdir ~/dev-tools && cd ~/dev-tools
```

Download and uncompress the zip file. For example:

```
wget
http://mirror.cc.columbia.edu/pub/software/apache/servicemix/serv
icemix-6/6.1.2/apache-servicemix-6.1.2.zip
unzip apache-servicemix-6.1.2.zip
```

### 3.12.2.8 Running and Configuring ServiceMix

In a command shell, navigate to the ServiceMix bin directory (e.g., ~/dev-tools/apache-servicemix-6.1.2):

```
cd ~/dev-tools/apache-servicemix-6.1.2/bin
```

Start ServiceMix:

```
./servicemix
```

Install the following features:

```
karaf@root>feature:install jdbc
karaf@root>feature:install pax-jdbc-mysql
karaf@root>feature:install camel-jsonpath
karaf@root>feature:install camel-jetty
karaf@root>feature:install camel-jdbc
karaf@root>feature:install camel-http4
```

You may need to download the pax-jdbc artifact from the Maven repository if the pax-jdbc-mysql install does not work:

```
karaf@root>feature:repo-add pax-jdbc 0.6.0
```

Create the JDBC connection to the MySQL database:

**Note:** Depending on your version of jdbc, you will need either the command “jdbc:ds-create” or “jdbc:create”. Type <tab> to print a list of available commands and find the one you need:

```
karaf@root>jdbc:ds-create -dn mysql -url
jdbc:mysql://localhost:3306/demo?user=broker&password=somepassword
mysqlDataSource

karaf@root>jdbc:create -d mysql -t MySQL -url
jdbc:mysql://localhost:3306/broker -u broker -p somepassword
mysqlDataSource
```

Check that the JDBC connection was created:

```
karaf@root>jdbc:datasources
```

Another way to check the database connection is to issue a query:

```
karaf@root>jdbc:query jdbc/mysqlDataSource "select * from users"
```

Install the broker application. The application will be installed as an OSGI bundle:

```
karaf@root>bundle:install -s mvn:org.apache.camel/camel-rest-
proxy-blueprint/2.16.3
```

Check that the broker was installed. The bundle should be the last bundle in the list and its status should be ACTIVE:

```
karaf@root>bundle:list
```

### 3.12.2.9 Install and Start ServiceMix as a Service

Start the ServiceMix if it is not already started. Issue the following commands:

```
karaf@root>feature:install wrapper

karaf@root>wrapper:install -s AUTO_START -n KARAF -d Karaf -D
"Karaf Service"
```

A message similar to the following will be displayed:

Setup complete. You may wish to tweak the JVM properties in the wrapper configuration file before installing and starting the service:

```
~/dev-tools/apache-servicemix-6.1.2/etc/KARAF-wrapper.conf
```

Redhat/Fedora/Centos Systems (SystemV):

To install the service:

```
$ ln -s ~/dev-tools/apache-servicemix-6.1.0/bin/KARAF-service
/etc/init.d/
```

```
$ chkconfig KARAF-service --add
```

To start the service when the machine is rebooted:

```
$ chkconfig KARAF-service on
```

To disable starting the service when the machine is rebooted:

```
$ chkconfig KARAF-service off
```

To start the service:

```
$ service KARAF-service start
```

To stop the service:

```
$ service KARAF-service stop
```

To uninstall the service:

```
$ chkconfig KARAF-service --del
```

```
$ rm /etc/init.d/KARAF-service
```

For systemd compliant Linux:

To install the service (and enable at system boot):

```
$ systemctl enable~/dev-tools/apache-servicemix-  
6.1.2/bin/KARAF.service
```

To start the service:

```
$ systemctl start KARAF
```

To stop the service:

```
$ systemctl stop KARAF
```

To check the current service status:

```
$ systemctl status KARAF
```

To see service activity journal:

```
$ journalctl -u KARAF
```

To uninstall the service (and disable at system boot):

```
$ systemctl disable KARAF
```

Exit the ServiceMix/Karaf shell, by shutting down ServiceMix:

```
karaf@root>shutdown
```



Install the ServiceMix service:

```
sudo ln -s ~/dev-tools/apache-servicemix-6.1.2/bin/KARAF-service
/etc/init.d/
```

Set the service to start when the machine is rebooted:

```
sudo update-rc.d KARAF-service defaults
```

Start the ServiceMix service:

```
sudo /etc/init.d/KARAF-service start
```

To log back into ServiceMix once the service is started, issue the following commands:

```
cd ~/dev-tools/apache-servicemix-6.1.2/bin
./client
```

To exit the ServiceMix shell without shutting down the service, type ^D (i.e., Ctrl-D). **Note:** If you type shutdown in ServiceMix shell, the entire service will be shut down.

### 3.12.3 Editing blueprint.xml Application File

The blueprint.xml file is provided with placeholders that must be edited for your specific environment.

#### 3.12.3.1 Update Zendesk Hostname

The blueprint.xml file has placeholders for the Zendesk hostname. You will need to provide your Zendesk hostname wherever you see the placeholder “<insert CRM hostname>”.

#### 3.12.3.2 Enable / Disable Proxy

If there is a proxy between the ESB and Zendesk, you may need to add the following parameters to the set of parameters specified for each instance of the Zendesk endpoint. For example, you may need to replace:

```
/api?bridgeEndpoint=true&throwExceptionOnFailure=false
```

with:

```
/api?bridgeEndpoint=true&proxyAuthHost=<replace with proxy
ip>&proxyAuthPort=<replace with proxy
port>&proxyAuthScheme=http4&
throwExceptionOnFailure=false
```

substituting your proxy host and port settings for the placeholders.

After editing the blueprint file, rebuild the application using Maven:

```
cd ~/code/camel-rest-proxy-blue-print/
```

```
mvn clean install
cd ~/dev-tools/apache-servicemix-6.1.2/bin
./client
karaf@root>bundle:install -s mvn:org.apache.camel/camel-rest-
proxy-blueprint/2.16.3
```

Make sure the bundle is successfully installed with no errors and active.

### 3.12.4 Testing the Broker Application

At this point, all of the code for the Broker application is contained in the OSGI Blueprint file (i.e., blueprint.xml) that can be viewed in [Github ESB broker application](#).esb application.

If the Broker application is running in ServiceMix running, you can check the application by using curl at the command line. For example:

```
$ curl -u
username@hostname/token:hLLUnPzJtpvMZ5WnntN3wCneKHkl20kP0Hhn5NrD
http://localhost:9090/api/v2/users/me.json --insecure
```

where username is a Zendesk user account and hostname is your Zendesk host name.

You can check the status and statistics of the main Broker route in the ServiceMix/Karaf shell:

```
karaf@root>camel:route-info rest-http-zendesk-mysql-demo
```

Here is example output from the camel:route-info command:

```
Camel Route rest-http-zendesk-mysql-demo
Camel Context: camel-1
State: Started
State: Started

Statistics
Exchanges Total: 2
Exchanges Completed: 2
Exchanges Failed: 0
Exchanges Inflight: 0
Min Processing Time: 240 ms
Max Processing Time: 494 ms
Mean Processing Time: 367 ms
Total Processing Time: 734 ms
Last Processing Time: 240 ms
Delta Processing Time: -254 ms
Start Statistics Date: 2016-07-19 14:43:44
Reset Statistics Date: 2016-07-19 14:43:44
First Exchange Date: 2016-07-19 15:12:15
Last Exchange Date: 2016-07-19 15:12:3
```

## Acronyms

<b>ACE</b>	Accessible Communications for Everyone
<b>ACR</b>	Auto Call Routing
<b>ADA</b>	Americans with Disabilities Act
<b>AMA</b>	Automatic Message Accounting
<b>AMI</b>	Asterisk Management Interface
<b>API</b>	Application Programming Interface
<b>ASL</b>	American Sign Language
<b>AWS</b>	Amazon Web Services
<b>CA</b>	Communication Assistant, Certificate Authority
<b>CAMH</b>	CMS Alliance to Modernize Healthcare
<b>CDR</b>	Call Detail Record
<b>COE</b>	Center of Expertise
<b>COTS</b>	Commercial Off-the-Shelf
<b>CRM</b>	Customer Relationship Management
<b>CSR</b>	Customer Service Record
<b>CSV</b>	Comma Separated Value
<b>DNS</b>	Domain Name System
<b>DVC</b>	Direct Video Calling
<b>EIP</b>	Elastic Internet Protocol
<b>ENUM</b>	E.164 Number to URI Mapping
<b>ESB</b>	Enterprise Service Bus
<b>FCC</b>	Federal Communications Commission
<b>FFRDC</b>	Federally Funded Research and Development Center
<b>GUI</b>	Graphical User Interface
<b>HSTS</b>	HTTP Strict Transport Security
<b>HTTPS</b>	HyperText Transport Protocol Secure
<b>iTRS</b>	Internet Telecommunications Relay Service
<b>I/O</b>	Input/Output
<b>IP</b>	Internet Protocol

<b>IPSec</b>	Internet Protocol Secure
<b>Java EE</b>	Java Enterprise Edition
<b>JB1</b>	Java Business Integration
<b>JDBC</b>	Java Database Connectivity
<b>JDK</b>	Java Developer Kit
<b>JRE</b>	Java Runtime Environment
<b>JSON</b>	JavaScript Object Notation
<b>KPI</b>	Key Performance Indicator
<b>NAT</b>	Network Address Translation
<b>ODBC</b>	Open Database Connectivity
<b>OpenAM</b>	Open Access Management
<b>OS</b>	Operating System
<b>OSGi</b>	OSGi Alliance (formerly Open Systems Group Initiative)
<b>PBX</b>	Private Branch Exchange
<b>POC</b>	Proof of Concept
<b>PSTN</b>	Public Switch Telephone Network
<b>QoS</b>	Quality of Service
<b>REST</b>	Representational State Transfer
<b>RFC</b>	Request for Comment
<b>RTCP</b>	RTP (Real-time Transport Protocol) Control Protocol
<b>RTT</b>	Real-Time Text
<b>SDP</b>	Session Description Protocol
<b>SIP</b>	Session Initiation Protocol
<b>SOA</b>	Service-Oriented Architecture
<b>SQL</b>	Structured Query Language
<b>S RTP</b>	Secure Real-Time Transport Protocol
<b>SSH</b>	Secure Shell
<b>SSL</b>	Secure Socket Layer
<b>STUN</b>	Session Traversal Utilities for NAT
<b>TCP</b>	Transmission Control Protocol
<b>TURN</b>	Traversal Using Relay NAT

<b>UDP</b>	User Datagram Protocol
<b>URD</b>	User Registration Database
<b>URI</b>	Uniform Resource Identifier
<b>URL</b>	Universal Resource Locator
<b>VPC</b>	Amazon's Virtual Private Cloud
<b>VPN</b>	Virtual Private Network
<b>VRS</b>	Video Relay Service
<b>VyOS</b>	Open source network operating system
<b>WebRTC</b>	Web Real-Time Communication

## Notice

This work was produced for the U. S. Government under Contract Number HHSM-5000-2012-000081, and is subject to Federal Acquisition Regulation Clause 52.227-14, Rights in Data—General, Alt. II, III and IV (DEC 2007) [Reference 27.409(a)].

No other use other than that granted to the U. S. Government, or to those acting on behalf of the U. S. Government under that Clause, is authorized without the express written permission of The MITRE Corporation.

For further information, please contact The MITRE Corporation, Contracts Management Office, 7515 Colshire Drive, McLean, VA 22102-7539, (703) 983-6000.

©2018 The MITRE Corporation. All Rights Reserved.