FaceTime: Voice over IP Technology

Table of Contents

I.I An Introduction to Facetime and VoIP

FaceTime is a proprietary Voice over IP service developed by Apple Inc. which establishes an end-to-end encrypted connection between two devices with many levels of networking abstraction. The connection must be secure and must allow clear conversations to occur on minimal bandwidth to support cellular data limits. This discussion will be focused on Facetime Audio for computers running Mac OS X 10.9.2 and later, which is the oldest OS X version that supports Facetime audio-only version[1].

II.I Boot Loader: System Initialization and Loading the Operating System

The operating system provides an interface that abstracts the computer's hardware from the user. When the computer is powered on, the firmware initializes the hardware, locates the boot loader from a data carrier based on its signature, and starts the system. Since the program counter indicates the predetermined address at which a program is executed, the boot loader in ROM must instruct the CPU to transfer the operating system from the hard drive to that location in RAM. The issue with storing the operating system in RAM is that it is volatile and thus erased on power off. Moreover, the issue with storing the entire operating system in ROM and eliminating the boot loader is due to the unalterable nature of ROM and the amount of security updates and device driver updates required to maintain current hardware needs. Once the operating system has been loaded into volatile memory, the boot loader will instruct the CPU to execute a jump instruction to give control to the operating system.

II.II Operating System: Kernel and Graphical User Interface

The operating system is composed of two main components: the kernel, which handles memory and processes management, I/O device drivers, and the Graphical User Interface, which enables the user to interact with the application.

II.III Kernel: Memory Management

The role of the memory manager is most apparent when there are concurrent applications running. If I am running Facetime in addition to Spotify and Google Chrome, then the memory manager must allocate the memory addresses to these processes to support multitasking, and remove unused segments of memory when processes terminate. Further, if a user decides to open many concurrent programs which exceed the main memory capacity, then the memory manager has to use paging to shuffle frames from main memory to mass storage to create the impression of additional memory - known as virtual memory. Another issue with concurrent processes addressed by the memory manager is processes accessing memory outside of its address space: each process requires a translation from virtual address to physical address using a translation table in order to be aware of the physical addresses it is allocated. If a process can alter the translation table, then it can inappropriately access other memory spaces. The memory manager is supported by dual-mode, in which hardware support requires that the CPU be in kernel mode in order to modify the translation table, extend time slices, or otherwise execute privileged code.

II.IV Kernel: Process Management

The scheduler and dispatcher are critical components of process management. When I open my Facetime application and request its execution, the scheduler will add an entry to its process table. It works in tandem with the memory manager to determine the memory space allocated to it, and assigns it a priority. The scheduler has a prioritization algorithm that determines which process can execute on the CPU. When the scheduler gives priority to Facetime, the dispatcher will allocate a time slice during which the process may execute so long as there are no interrupts that will result in a context switch.

II.V Graphical User Interface

After the Facetime application has successfully been opened, the allocation of screen space for the application is managed by the window manager of the operating system. Each of the widgets on the screen (i.e. end call, take screen capture, mute self, and effects) in addition to my user-specific settings are non-code resource files that come with the application when downloaded and are all stored in mass storage with the main executable. The majority of Facetime's functionality is implemented in libraries storing reusable program parts such as templates or subroutines. In OS X, these are dynamic libraries that are not part of the main executable. Since a single copy of the library's file is required at compile time, launching the app is responsive because storing the dynamic library separately reduces the executable file size. However, run-time execution speed is fast for static libraries since many calls to functions can occur readily as the binaries are within the executable file, instead of having to be called externally.

II.VI I/O Device Management and Concurrent Processes

The iOS kernel manages device drivers. Drivers are software units that interact with hardware known as controllers which in turn operate on peripheral devices. Each controller is responsible for a specific device type and has a local buffer to store data. If Facetime is requesting a user to enter their password or some sensitive data, an Interrupt Service Routine is generated by the dispatcher which will context switch away from the current process to the highest priority process based on the scheduler. In this way, the Facetime I/O device request causes the Facetime process to enter a waiting state (as it is being serviced by an I/O device, i.e. keyboard) and a high priority process is run concurrently on the CPU as the input request in the I/O device queue is being processed. Once the user has entered data that is stored in the local buffer of the keyboard controller, the controller informs the CPU that it has completed its task by causing an interrupt. Now, the CPU will transfer the data in the I/O device's local buffer to a memory location that can be accessed by Facetime which requested the I/O. Following this, the Facetime process is marked by the scheduler as a ready state, and can be run on the CPU again.

Other peripheral devices that are required for Facetime include a network card, mouse, microphone, and speakers. Some I/O devices interact with the GUI directly: for example, a mouse click within the Facetime window will have its cursor position determined by the window manager, and will route this information to the Facetime application program. If the mouse click was on a button that reduces the volume of output, Facetime would make a system call to the audio output I/O device to queue this request to be serviced (and similarly enter a waiting state). In modern computers, Direct Memory Access (DMA) is common, which means that data can be transferred between memory and peripheral devices without involving the system processor. This improves the device's throughput by reducing overhead of the CPU from data transfer[2]; however, this requires synchronization between the DMA controllers and RAM to ensure that outdated data is not accessed.

III.I Networks: IP Protocol

Facetime uses the existing Internet infrastructure to transmit IP packers over networks using packet switching, and is thus scalable because it scales with the hardware of the Internet. IP protocol uses the IPv6 address obtained through DNS to route packets based on the destination address in the packet header.

III.II Networks: TCP Transport Layer Protocol

When a user starts a Facetime call, an initial TCP connection over Port 5223 occurs to set up the call. TCP uses a three-way handshake from the application to the Apple server. The client will send a TCP SYN packet in addition to a sequence number. The server will reply with a SYN-ACK that acknowledges the sender's packet by incrementing the client's sequence number by one and sending its own sequence number. The client will then reply with a SYN message with an

acknowledgement number which is the server's sequence number increment by 1: this establishes a stable connection to begin data transfer.

III.III Technical Implications of TCP

Since reassembling packets at the destination requires time, minimizing reordering and reducing network congestion is critical to reducing the required to set up a Facetime call. To minimize reordering, a hash algorithm such as LAG or CEF can be used to ensure that packets corresponding to the same connection are sent out identical paths to minimize reassembling at the destination[3]. TCP slow start is a congestion control algorithm that is used to regulate the amount of data flowing through a network in order to prevent packet loss due to timeout[4]. Both of these optimizations are critical to implement since TCP is inherently slow due to handshaking.

Network security is another implication of TCP: since packets must be reassembled when they arrive at the destination address indicated by the header, this can be exploited by users with malicious intent. These users simply need the correct sequence number and the sender's IP address and can send incomplete or overlapping packets, which makes reassembly impossible. The solution is to treat these offending packets using the same sixty second timeout mechanism that is applied to packet fragments. TCP was optimized to perform better on Wide Area Networks (WANs) rather than Local Area Networks (LANs)[6]; this has the unintended drawback of increased packet loss due to more routers and thus more rerouting and the increased potential for overload.

III.IV Networks: STUN

iOS devices within Local Area Networks (LANs) require Session Traversal Utilities for NAT (STUN) in order to obtain a public IP address[7]. Since having a public address is a requirement for TCP, this enables Network Address Translation Traversal (NAT), meaning IP addresses of the caller and callee hidden behind a firewall or within a local network can establish peer-to-peer communication through the Facetime application.

III.V Networks: UDP Transport Layer Protocol and RTP Transport Protocol

When engaging in a Facetime audio call, users are more concerned with call latency rather than the minutiae of audio detail. UDP fits the criterion for a real-time communication service: this protocol optimizes the timely arrival of data packets by foregoing connection establishment, error control, and packet ordering or loss. It would benefit the user experience to have minor packet loss - resulting in 10 to 30 milliseconds of disrupted audio - rather than retransmitting audio at each instance of packet loss resulting in many milliseconds of silence or lag[8]. UDP uses Apple's port 16386 and 16386, which are also associated with the Real Time Processing (RTP) protocol which is designed to reduce latency and compensate for the lack of error detection of UDP.

III.VI Technical Implications of UDP and RTP

Although RTP has mechanisms to detect packet loss, packet loss can occasionally be a design of Facetime calls. In order to increase perceived responsiveness to the user, Facebook may intentionally limit throughput during periods of high network congestion. Datagrams will invariably pass through many routers along its route: UDP makes message interception by unintended recipients difficult because individual datagrams carry little meaning. Port 5223, used for TCP, is also used to support XMPP connections. Although Facetime is modeled as an "open protocol" RTP is dependent on the XMPP protocol in order to provide a security verification based on mutual authentication - one certificate from the client's device and another from the Apple server[9]. This design means that non-iOS devices are barred from using Facetime; even if a user could reverse engineer the connection, the certificate would be impossible to bypass without proper licensing from Apple.

IV Data Representation

The operating system kernel has a device driver that enables a generic interface to communicate with the computer's microphones. The input data is audio generated from the user, which are sound waves that vibrate the diaphragm. The analog data is digitized by an analog-to-digital converter. This digital signal is represented as discrete patterns of bits, unlike the continuous signal of analog data. This digital data is then compressed using a codec called AAC-LD[10]. Since a broadband connection is already required, compression is needed to make bandwidth needs less demanding. The lossy compression version of AAC could reduce this further. Each data datagram is composed of binary data (i.e. 0s and 1s). Digital data is reliable since information moving through a wire is subject to degradation and binary data is discrete. Although analog signals enable much more information to be sent, degradation of the data means that it can be misinterpreted at the receiving end as it is continuous. The driver for the network card interfaces a wireless or ethernet connection to a router. The packets are routed through packet switching through many different routers from the one in the user's Local Area Network (LAN) to Apple's service provider server for Facetime through many Wide Area Networks (WANs) and perhaps across national or international ISPs, as it is being routed from the server to the recipient on the other end of the Facetime call. The binary data is stored in one of the address spaces allocated to the process as long as audio input is being detected by the driver interfacing with the microphone. The recipient's application receives the compressed binary data, stores it in memory. The codec decompresses the data, transmits the data to the driver corresponding to the speaker controller. The output driver converts the stream of discrete digital data into continuous analog data using a digital-to-analog converter (DAC) which induces the diaphragm of the speakers to vibrate, producing sound waves that are received by the callee.

V.I Design Considerations

FaceTime has proprietary standards that makes them not interoperable with other devices such as Windows OS. This design choice helps retain customers within the Apple ecosystem.

VI.I References

[1] Wikimedia Foundation. (2021, December 16). FaceTime. Wikipedia. Retrieved December 12, 2021, from https://en.wikipedia.org/wiki/FaceTime

[2] Sonya. (2021, September 23). What is direct memory access (DMA) and how does it work? MiniTool. Retrieved December 14, 2021, from https://www.minitool.com/lib/direct-memory-access.html#:~:text=DMA%20is%20also%20used%20for%20on-chip%20data%20transfer,memory%E2%80%9D%20to%20copy%20or%20move%20data%20in%20memory.

[3] Wikimedia Foundation. (2021, December 4). Ip fragmentation. Wikipedia. Retrieved December 15, 2021, from https://en.wikipedia.org/wiki/IP_fragmentation#Impact_on_network_forwarding

[4] Gibb, Robert. "What Is TCP Slow Start? How TCP Slow ... - Stackpath Blog." *STACKPATH*, 10 June 2016, https://blog.stackpath.com/tcp-slow-start/.

[5] Wikimedia Foundation. (2021, December 4). IPv6 packet. Wikipedia. Retrieved December 16, 2021, from https://en.wikipedia.org/wiki/IPv6_packet#Fragmentation

[6] Langer, Steve G, et al. "TCP/IP Optimization over Wide Area Networks: Implications for Teleradiology." *Journal of Digital Imaging*, Springer-Verlag, Apr. 2011, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3056981/.

[7] "Stun, Turn, and ICE NAT Traversal Protocols." *AnyConnect*, https://anyconnect.com/stun-turn-ice/.

[8] OnSIP. "UDP versus TCP for VoIP." OnSIP, https://www.onsip.com/voip-resources/voip-fundamentals/udp-versus-tcp-for-voip#.

[9] “TCP and UDP Ports Used by Apple Software Products.” Apple Support, 14 June 2021, https://support.apple.com/en-us/HT202944.
[10] “Voice over IP.” Wikipedia, Wikimedia Foundation, 12 Dec. 2021, https://en.wikipedia.org/wiki/Voice_over_IP#Protocols.