Journal of Engineering, Electrical and Informatics
Vol.2, No.2 Juni 2022
e-ISSN: 2809-8706; p-ISSN: 2810-0557, Hal 04-22

ARM TRUSTZONE AS A SECURITY FOR USER INTERACTION ON THE MOBILE PLATFORM

Lukman Santoso
Universitas Sains dan Teknologi Komputer

Moh Muthohir
Universitas Sains dan Teknologi Komputer

Ahmad Ashifudin Aqham
Universitas Sains dan Teknologi Komputer

Jl. Majapahit 605, Semarang, telp/fax : (024) 6723456

Abstract*. Smartphones have become an important part of human life until now, apart from making phone calls and other communication functions, smartphones are also used as tools for shopping, conducting banking transactions, and so on. This makes security Software on smartphones becoming more critical. trust execution environment technology was popularized to support care for users if the Operation System is compromised namely trust execution environment ARM TrustZone. This research focuses on User-Interaction and Input/Output audio input user interaction channels and proposes an application-transparent solution to keep the user communication strategy on mobile platforms using ARM TrustZone.*

This research advantage the ARM TrustZone to bring a defended architecture for User Interface input for user interaction channels (T-User-Interaction) and Audio-Input/Output for VoIP calls (T-VoIP-C). The main objective is to establish that the architecture is clear for cellular utilization. All along trust execution environment-established research, one of the critical objections faced was the ability to safely prototype designs. In a trust execution environment study, it is generally necessary to cross hardware factors with the trust execution environment-Operation System, which could be impressive for non-hardware specialists, conditional on the base applicable from trust execution environment Operation System vendors. This research considers a simulation-based model (T-simulation) that cuts down the composition time and hardware background appropriate to frame a hardware situation for expectation execution situation prototyping.

The result of this research is, T-simulation can reduce the setup time required to set up a hardware test environment for conducting trust execution environment research. For a typical research project, using T-simulation is sufficient to calculate the usefulness of the design. Accustomed that the present emphasis of T-simulation being placed on the Hikey and the Raspberry Pi. The confluence backing one gets via the Pi board matches the low/high-speed headers. In performance evaluations for GPS and cameras, there was a lag detected among the Trust Application and the Pi board when using RPC channels. For future research, further investigation is needed on the labeling permissions of User-Interaction elements in WebView to be secure. In addition, in the VoIP Computing stage in the trust execution environment, there is a need for light audio computing channels in trust execution environment which should be able to achieve sufficient audio quality

Received April 07, 2022; Revised Mei 2, 2022; Juni 22, 2022
* Lukman Santoso

improvement, so it is necessary to discuss acceptable TCB exchange in the trust execution environment vs acceptable audio quality for VoIP calls.

***Keywords**: Android, Smartphone, ARM TrustZone, Securing User Interaction.*

INTRODUCTION

The widespread use of smartphones makes mobile Operation System security very important, but the level of vulnerability in Android remains high. Attacks smartphones (android) can accomplish capricious code beheading in powerful processes by adopting custom image files. Android brings various synergy channels that can be concerned by a negotiating mobile Operation System (Fig. 1), counting user interaction oversight (User interface and audio-Input/Output), and situation-based oversight dial-to-phone. A negotiated mobile Operation System can alter different use cases for this channel. Can steal user secrets on user communication oversight such as User-Interaction input and audio-Input/Output, can spoof environments on context-based channels such as camera and location, and can steal data exchanged between devices on lines between phones such as NFC and Bluetooth, and can kidnap data that is posted into the slave or deliver fake data to the server on the back-end channel with the server.

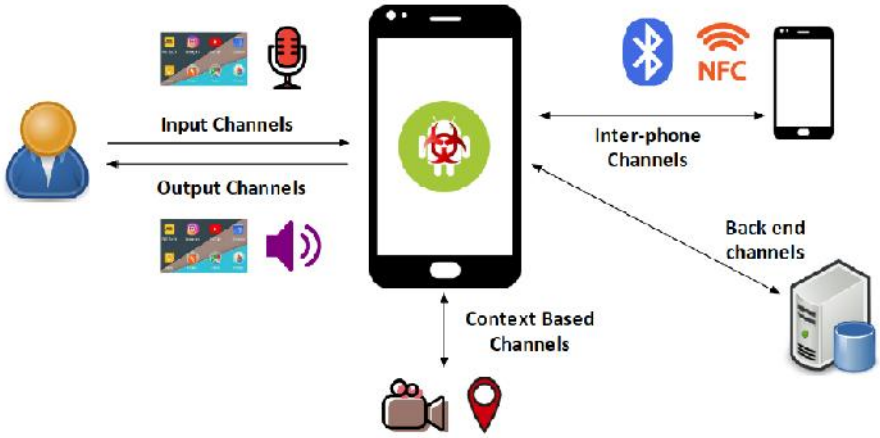

Figure 1. Smartphone Interaction Channels

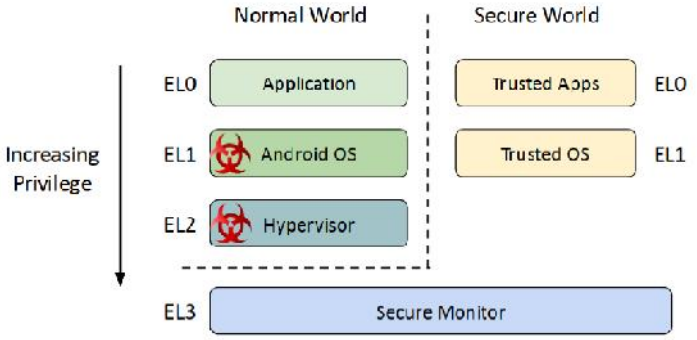

Figure 2. ARM Architecture Privilege Levels

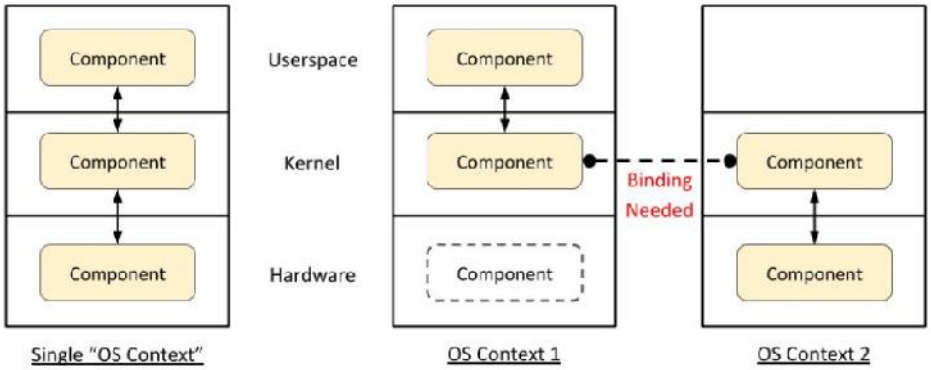

Figure 3. Operation System Context and Conclusions

This research focuses on User-Interaction input and audio-Input/Output user interaction channels. Input User-Interaction is used to grant users to get in secrets or accept actions in cellular applications. A compromised mobile Operation System can listen in on a user's VoIP calls. Bring the exposure of UI-oversight, here is a use to architecture explanation with mobile architectural features that can provide security even if the mobile Operation System is compromised. The best part of mobile devices uses the ARM design which is split into 2 worlds (Figure 2). The first is the general world of normal apps and mobile Operation Systems (Android). Research by Nordholz et al (2016) shows that, on the ARM platform, there are vulnerabilities in the hypervisor. Second is the protected world, also indicate as the trust execution environment, controlled by a hit situation that is confined from the general world. The most used trust execution climate on cellular material is ARM TrustZone. A compose Operation System in cellular is cannot access data in the protected world and cannot access hardware protected by the protected world. The protected world runs an independent trusted Operation System with its own set of trusted applications. In a specific aggregation arrangement, the entrails in the user space, the core, and the hardware are connected to form a single Operation System context. At the user-space matches, elements could add development, components can include various factors worn by the rule. In the context of OS, this study uses the term conclusive which refers to the interaction between two components through Operation System support.

BACKGROUNDS

TrustZone technology is a system-wide security model that enables establishing protected endpoints with a root of trust. Using TrustZone, System-on-Chip (SoC) hardware and software resources are partitioned to provide security. General world software is not allowed to access protected world assets. The concept of a general and safe world is tested in different elements of the SoC. The two worlds are partitioned using hardware logic implemented in the bus, peripheral, and processor structures. to get For testing, this research relies on Hikey Board 620. Operation systems especially android, grant end-users the to install various keyboard applications to use. The InputMethodManagerService (IM-MS) system service will run when a user has a specific keyboard. When a user combines “EditText” in the operating system of Android, InputMethodManager will send an IPC request for the keyboard User-Interaction to the IM-MS and then the IM-MS will then request the selected keyboard app to display the User-Interaction board. EditText posted API after input is done. To obtain user

acceptance, the application action can call an alternative action that contains the authorization for User-interferences. Based on the confirmation result, the corresponding code can be triggered in the calling Activity. The calling activity defines a confirmation message as part of the Intent and uses the startActivityForResult() API for the confirmation User-Interaction call. The User-Interaction can run in the same or different processes and is called via ActivityManagerService. After the user interacts with the acceptance actions of User-Interference, the result (finish()) is sent to the calling application by the IPC Intent scheme. The calling app gets a response from the confirmation User-Interaction through the onActivityResult() callback.

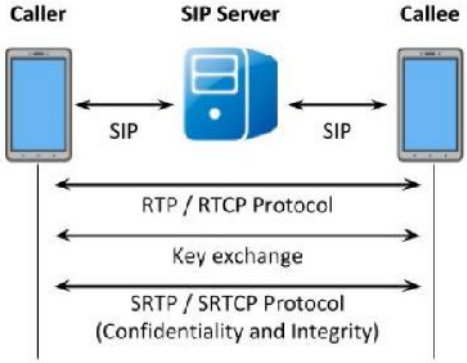

Figure 4. VoIP-C Flow

UI interaction events are collected by the hardware and authorized to Linux equipment drivers in the core. The Android system server process will receive events in the integral “InputReader” and passes them to the “InputDispatcher” which sends events in utilization through the InputChannel layer in the operation process. From the available, a popular protocol for open source implementations of VoIP is SRTP/Secure-Real-time-Transport-Protocol adopting SIP Session-Initiation-Protocol for call induction. Figure 4 provides high-tier aspects of the outflow accuracy. Once the connection is established, the RTP (Real-Time Protocol) protocol is used to transmit audio between the two endpoints. RTP was adopted in partnership with Real-Time ControlProtocol (RTC-P) to auditor the communication data and aspects of service. SRTP (located between the RTP application and transport layers) is the figuration of RTP to bring converted, directive verification, and return conservation to RTP traffic. SRTP and SRTCP require a key for encryption and HMAC. The code adopted to manage master keys consist of DTLS and ZRTP.

RELATED WORK

VeriUI (Liu et al, (2014)) “protects web login pages through retreating the WebKit engine and GUI libraries to TrustZone, VeriUI is designed to protect entire web pages”. Nonetheless, T-User-Interaction goals the granular User-Interaction show symbols that make up the entire Activity. Existing work required this research to add the Trust Application code and change the application to call the Trust Application code. The concept of protected VoIP calling on a trustless Operation System is analyzed in the work by Ender et al., (2014) "A Hardware-Assisted Proof-of-Concept for Secure VoIP Clients on Untrusted Operating Systems"). Ender et al., (2014) used a Linphone application (Open Source VoIP) to test and modify it in such a way as long as approaching SRTP file and T-VoIP-C: (1) Commercial phones do not rely on FPGAs; instead, they ship with

ARM boards that have TrustZone. Existing work does not address any of the challenges associated with leveraging TrustZone for secure VoIP. (2) Xillinux Operation System does not reflect mainstream mobile Operation Systems like Android. The existing work does not address leveraging TrustZone on the mobile Operation System audio stack to allow the use of the existing Audio API. (3) VoIP applications have streams to handle the audio file. in the current study, RTP has been removed from the general world of application flow by design to forward hardware-secure headers/payloads at the layers of SRTP. TruzCall's goal is to maintain the relative structure of the critical parts of the software stack for VoIP applications and avoid moving components unnecessarily into the trust execution environment. In short, Ender et al., (2014) research is unclear and has a high TC-B.

DR-M keeps audio/video playback secure with a trust execution environment, but the reference design and management of trust execution environment data adopted in DR-M cant apply to VoIP. Rio et al (2014) provide “Input/Output allocation between mobile devices by parting the Input/Output heap at device file boundaries”. Kuo et al, (2016) “Semantics-Aware Design for Mounting Remote Sensors on Mobile Systems frame an unknown suppress Input/Output heap that is adequate in details of power costs and communication time”. Mobile Plus (Yoo et al, (2017)) “enables operation system (Android) operations to leverage system functionality beyond devices by approaching the Android binder inter-process communication (IPC) mechanism”. TrustUI (Li et al, (2014)) adopted a discrete equipment driver design to grant TAs in a trust execution environment to deploy hardware using general world drivers. This approach is unfeasible as factors like GPS sensors and cameras because the general world would allow meddling by the raw data fed to the protected world. Nothing study can solve the trust execution environment problem to provide Trust Application transparent hardware access for different hardware categories. This research today translucent architecture that grants the general-world applications to bargaining chip TrustZone over existing Operation System APIs to protect user interactions without requiring application-specific Trust Application code within a protected world.

LITERATURE REVIEWS

TrustZone can grant the end users to characterize their confidential information in the right applications after cracking them into the trustless general world Operation Systems. For apps to protect user input interactions with minimal change, developers need to allow to adopt current operation system (Android) elements and APIs still allow to take advantage of the trust execution environment post. In case the application must have to change Android basics to accommodate trust execution environment support, this will result in significant changes to the application. In the input interaction threat model to the user’s utilities is secured. The general world admits Android apps and Operation System is unbelievable. It’s may try to divert confidential user data and fake unauthorized responses. A protected world has secure trust Applications and a trust execution environment Operation System. This aim covers user privacy and principles during the general world is compound.

Journal of Engineering, Electrical and Informatics
Vol.2, No.2 Juni 2022
e-ISSN: 2809-8706; p-ISSN: 2810-0557, Hal 04-22

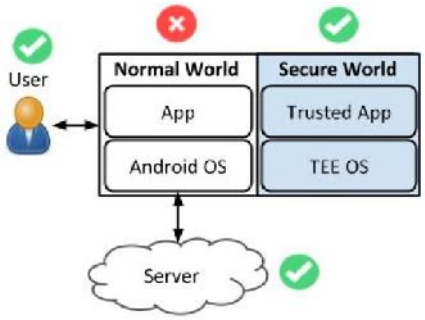

Figure 5 Input Interaction Threat Model

That the model in this research is to bring about the necessary conservation to change the conscious User-interactions to a protected and control the conclusive between the User-Interaction interactions and the general world application code across the OS. This cross-OS conclusive allows applications to take advantage of the User-Interaction in a trust execution environment by adopting current application-programming interfaces. The app developer calls the User-Interaction and gives the identical code to bring about user interferences. To protect user interactions at the machine reversal, these factors only can be accessed in safe world. Users also need indicators to identify whether they are interacting with the general world or the safe world. Indicators should be controlled exclusively by a safe world. The proposed design leverages the TZPC to enable the protected world to have absolute control over Input/Output and indicators.

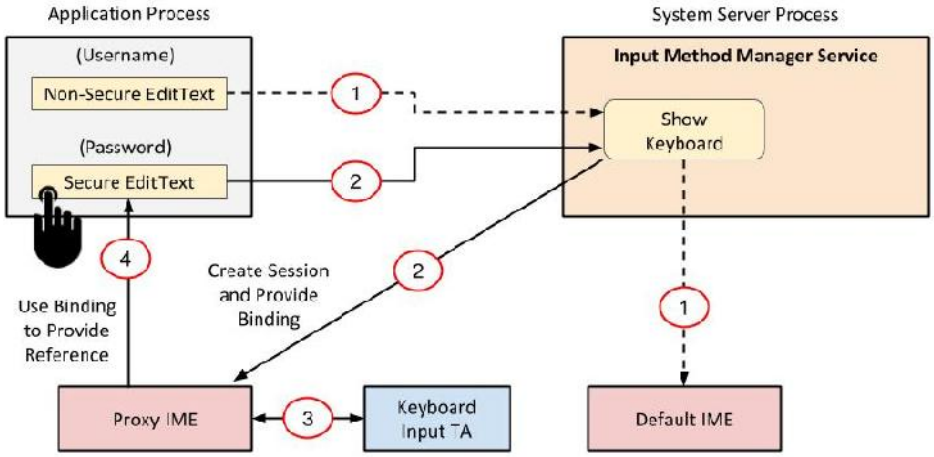

Figure 6 Switching to I-ME Proxy for Secure EditText

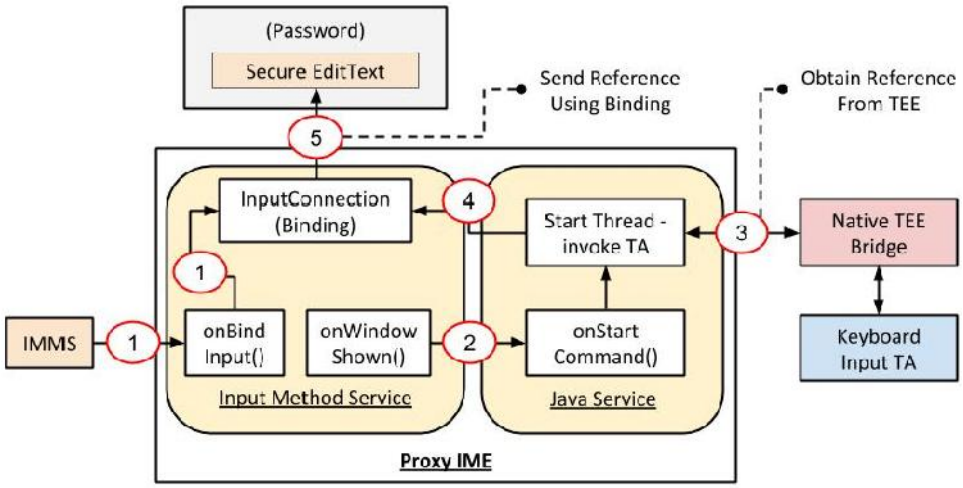

Figure 7. Intermediary I-ME Committing Allusion Collected from Keyboard Input Trust Application

To modify “InputMethodManager” on Android Application sends a keyboard view request to IM-MS via InputMethodManager. With a modification to the InputMethodManager, during the UI is in protected EditText, IM-MS will prompt the various EditText. The IM-MS use to connect with the I-ME intermediary operations to grant secure text input. During the IM-MS accepted a protected (secure) request, the actual I-ME that the IM-MS knows is modified to the application name of the intermediary IM-E. IM-MS (figure 7) establishes the period with the I-ME intermediary utilization. IM-MS takes it by “invokingbindInput()” on “theInputMethodService” of the app's IME. This allows the proxy I-ME to obtain an InputConnection type conclusive using the getCurrentInputConnection() function in onBindInput().

To trigger a keyboard input Trust Application call as long as protected text input, the model is adopted “onWindowShown()” in I-ME proxy's InputMethodService.onWindowShown() to be called before the I-ME wins is displayed to the user. Trust Application get user input, stores it in the trust execution environment, and returns a reference (according to the user secret) to the Java service. The reference is then sent to the EditText in the application using conclusive. In the T-User-Interaction threat model, the general world's Operation System declined to bring similar safety measures to the trust execution habitat for users during agreement. The general world can't see user secrets and aim to gain the code packets generated in a protected world. Packets aim to server by the general world TCP/IP bundle (figure 8), the secure User-Interaction also displays the hostname.

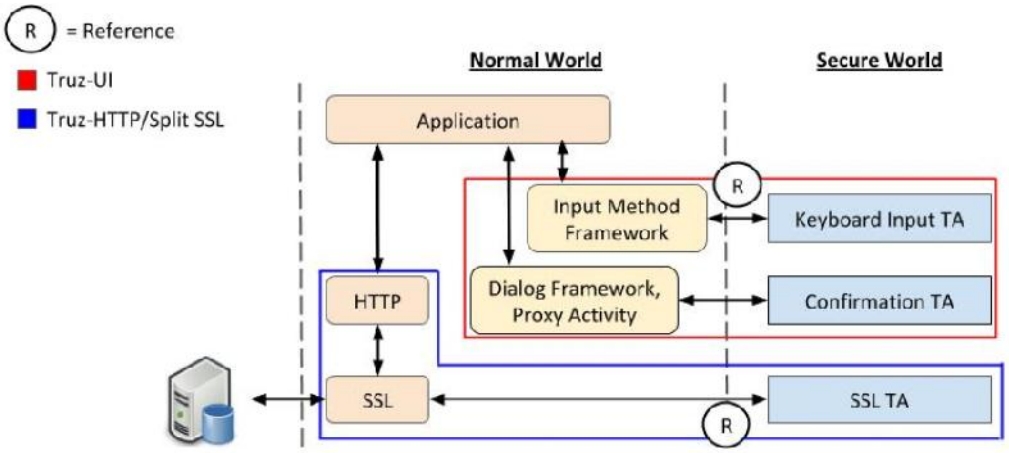

Figure 8 network T-User-Interaction Jobs and Truz-HTTP/Split-SSL

The prototype was built using the HiKey board development. The prototype uses a TFT LCD panel as the screen. The display uses an HDMI interface for display and a USB interface for touch control. The hardware application brings remoteness for user input and display. Nonetheless, while the protected world dominates it, the general world cannot access screen Input/Output. Isolation is achieved at the circuit level. The display Input/Output is linked to the compositor/un-compositor (figure 9). The compositor catches HDMI signals to the screen. The demultiplexer takes touch input from the screen and feeds it to one of the worlds. Switches are used to control the multiplexer/demultiplexer. Each world has a separate Input/Output port connected to the multiplexer/demultiplexer. Switch control is accessible only by protected world Input/Output ports. To express the user, the safe world turns on LED when the device is in the safe world. The TrustZone Protection Controller (TZPC) is configured to allow the protected world to have exclusive control over protected world switches, LED indicators, and Input/Output ports.

USB and HDMI connected to Hikey provide the general world display outputs and inputs. Due to the other vendor driver backing, the protected world cant precisely bring the input/output for touch screens. For this reason, a protected world build on the Raspberry Pi board (connected via UART). UART can only be accessed in a protected world adopting TZPC. High-definition multimedia interface and universal serial bus connection to the Pi board bring display output and input for a protected world. The User-Interaction for the keyboard input Trust Application and confirmation Trust Application is provided by the Python code running on the Pi board. TAs running in the protected world s get results via UART. The HDMI input switch and display switch are controlled by the Pi board, which in turn is controlled by the safe world. There is an LED on each switch indicating which world is in control. For the Trust Application to interact with the Pi board, the driver is must be easy to access. To transfer approach, the OP-trust execution environment Operation System kernel calculates different system calls so that the Trust Application could take advantage of the driver.

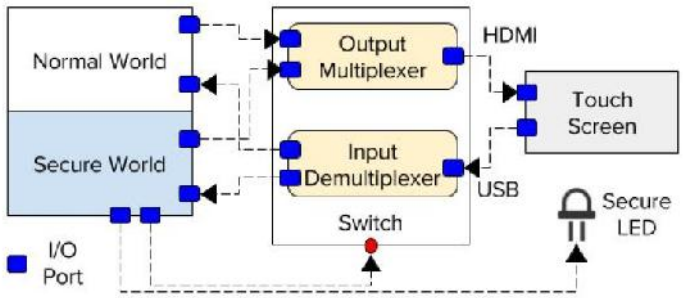

Figure 9 Hardware Setup Overview for T-User-Interaction

Figure 10 Overview of VoIP Calls

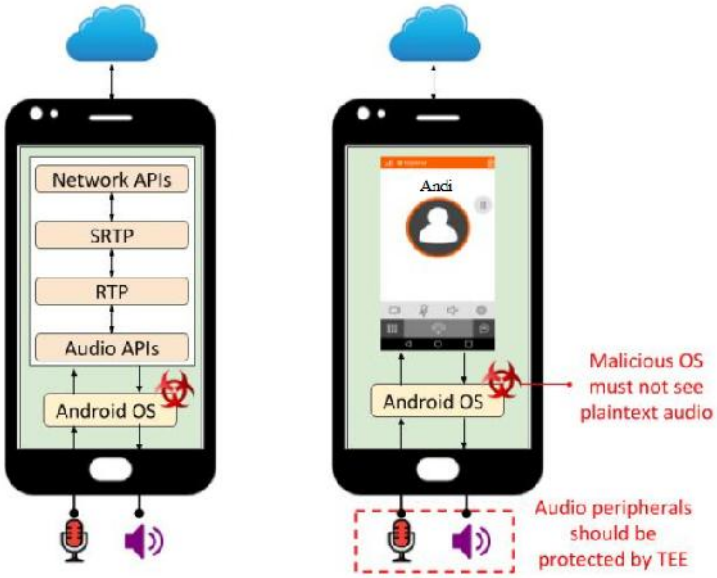

Figure 11 Stages of VoIP Applications and Secure VoIP Needs

After a VoIP- C is placed, the caller brings the input of audio and accepts the output of audio by the audio properties factors. A compromised Operation System can listen in on a user's conversations during a VoIP call. TrustZone can be leveraged to protect user voice interactions due to the hardware level isolation it offers. Since the design will utilize the TrustZone the TCB in a protected world must be minimized. During a trust execution environment-secured VoI-C, the audio factors must be disciplined by the protected world and the user's conversational audio must be protected from the general world's Operation System (Figure 11 (right). The challenge faced in designing a system like T-VoIP-C is latency. Because VoIP is a real-time system, if the general world stack calls trust execution environment in more points it will give to collect the time. Another challenge is the hardware setup to perform prototype evaluation. In trust execution environment research, connecting hardware peripherals such as microphones and speakers with the trust execution environment Operation System on development boards can be a challenge.

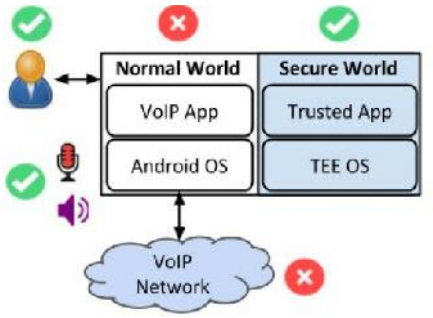

Figure 12 Voice Interaction Threat Model

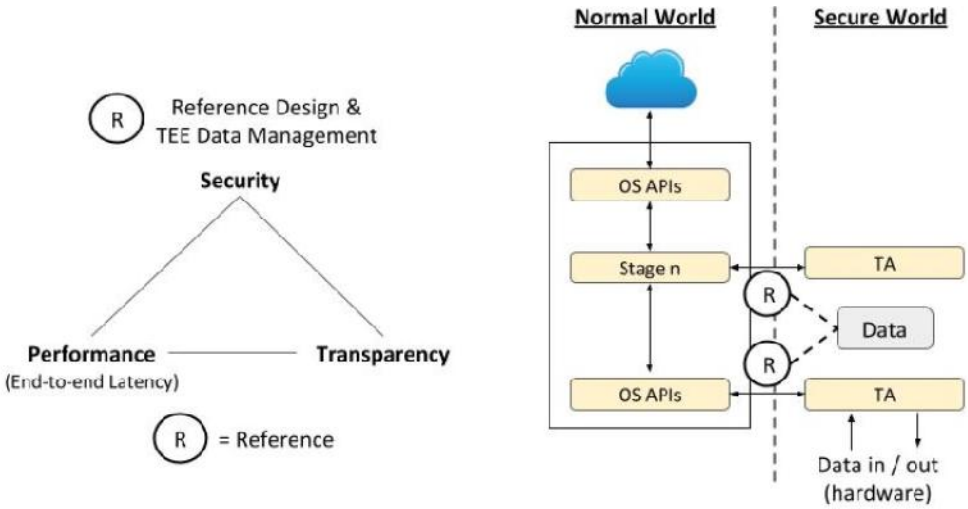

Figure 13 trust execution environment Integration Design Factors

VoIP networks cannot be trusted, although T-VoIP-C does not attempt to secure contra network-based attacks. T-VoIP-C has conducted the users that want to call someone in safely. T-VoIP-C could be continued to calculate key network adopting trust execution environment by sharing protocols such as DTLS. During scheming a security result on a mobile platform by integrating the trust execution environment into the general world heap in Figure 13 (right), and be balanced in Figure 13 (left). This is accomplished

by using references. Reference design (structure) affects clarity in the general world heap because the general world heap stages achieved on the mentioned data trust execution environment are unified and notified at a different level (general world bundle). The general world's number of trust execution environment alliance score and the step of data that managed in the trust execution environment affect overall latency.

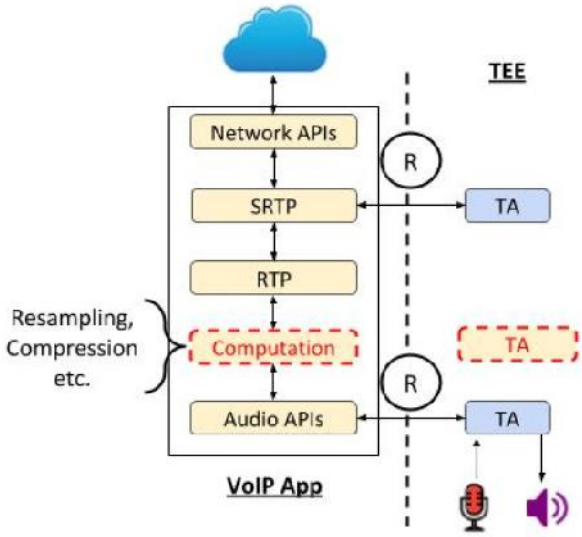

Figure 14 Stages of VoIP Applications

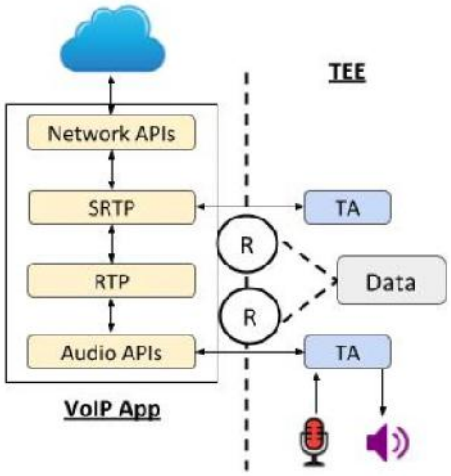

Figure 15 T-VoIP-C Design Summary

VoIP applications could be recorded to make calls in the transparent text (adopting protocols). This research targets the capacity of the problem of determining trust execution environment support for applications that previously bring end-to-end encryption but aspect privacy risks as long as negotiated by the Operation System. For VoIP applications, processing audio here is carried out in different level (Figure 14). The stages consist of audio-Input/Output, audio computation (such as resampling, and compression), RTP, SRTP packet construction/parsing, and network I/O. One of the stages in Figure 14 is marked as computation. To improve audio quality and reduce

bandwidth requirements, VoIP applications apply several types of additional computations to the audio data. For audio data read from the mic, the computations applied can include read resampling (downsampling), volume adjustment, equalization, and compression. Before playing the received audio, the computations that are applied may combine uncompress, volume modification, stabilization, and instances. For the T-VoIP-C scope of the problem. This model is targeting on the critical level of audio-Input/Output, RTP, SRTP, and network Input/Output packet construction/parsing. Architecture deduction audio aspect for guarantee. The main idea of T-VoIP-C (Figure 15) is to have the different levels in a VoIP heap functioning lateral as a channel, with each level biting data into the next. To allow the VoIP channel to control flows during retaining audio of user conversations in a trust execution environment, the design is called a trust execution environment in stages for the use of the audio and APIs. That grant the use of the analogous construction of existing software stacks. The design calls up the trust execution environment get by the reference data audio in the trust execution environment according to the reference and returns the encrypted payload and HMAC to the SRTP layer to allow the VoIP application move to carry on. The T-VoIP-C architecture proved on Linphone's open-source VoIP application. The trust execution environment Bridge app enables Java code to run Crypto and Playback Trust Application which is important for cryptographic operations in the trust execution environment.

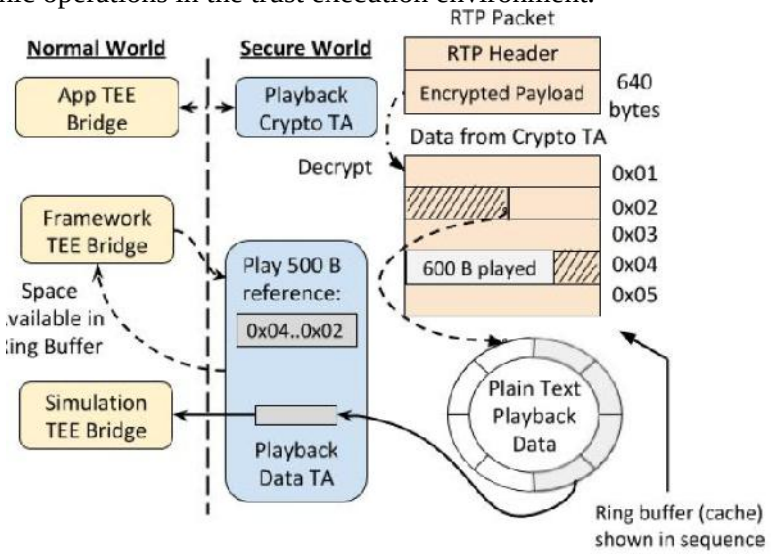

Figure 16 Management of Reference Data for Playback

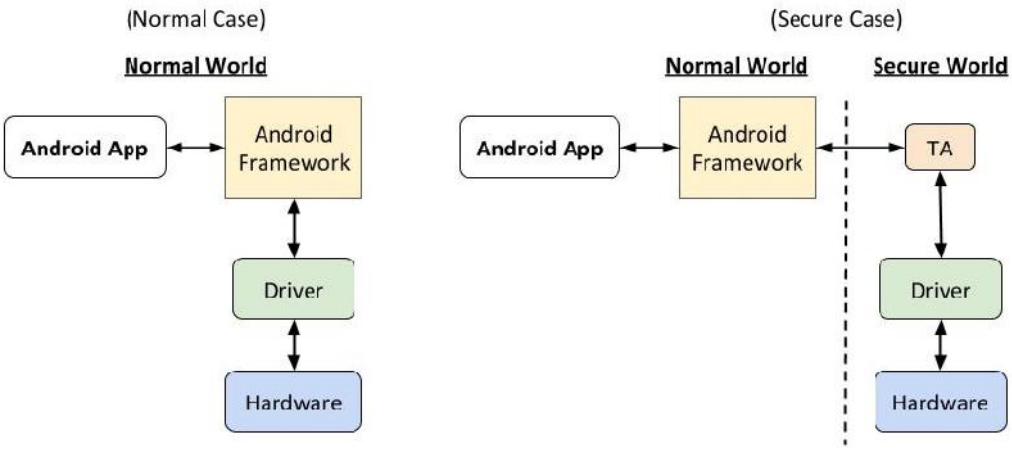

Figure 17 Accessing Hardware in Normal vs Safe Cases

TruzCall design limitations

Reducing latency requires a time-saving approach in a trust execution environment to develop the screen to be encoded. To achieve this, the SDP's in-memory cache is set up by storing audio plaintext in various packetization encode sizes. Whenever the requests a trust execution environment audio file previously revolving around the quotation copied into the cache. The cache always prepares the next buffer for RTP. Because the reference returned by the trust execution environment is apparent to be the same length as the number of bytes requested, Trust Application gains expertise that was starting to take the “memset()” and loops through the index in the cache. When the SRTP library calls the trust execution environment. The different 2 versus 1 “memcpy()” may seem insignificant, but it's worth noting that trust execution environment invocations occur multiple times per second during the call. T-VoIP-C matches VoIP jitter buffer behavior in the trust execution environment. This allows playback on accepted audio back of being reset by the general world. Specific hunting is performed for the total of audio played on the trust execution environment for each encoded RTP payload.

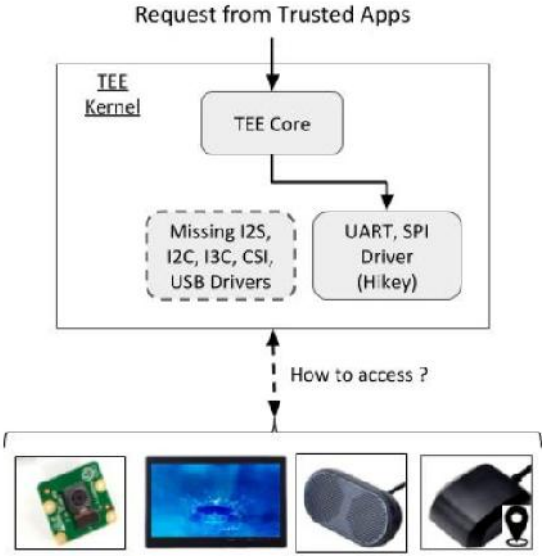

Figure 18 Driver Support in the trust execution environment Kernel

During SRTP Library accept a packet it will be forwarded to the Crypto Trust Application Playback for HMAC verification and decryption (figure 16). Since there is no adequate driver support in the trust execution environment OS, designs need to apply the external drivers of the trust execution environment to interact with the hardware. To control clarity assimilation apps will still use the current Android framework APIs trust execution environment's assimilation by the Android structure will appeal to Trust Applications in a protected world. In T-simulation, the architecture attempts to bring the action/service of the trust execution environment-mounted hardware by taking advantage of the hardware installed in the second Operation System and using the appropriate drivers.

EXPERIMENTS, RESULTS, AND EVALUATION

T-USER-INTERACTION

Security Analysis

Hardware invasion, crypto rape, side-channel invasion, and DOS invasion are treated out of reach. The enemy's purpose is to control the various secret by worming into displayed content, and reading secrets stored in a protected world. The protected world presents a secure User-Interaction. The TrustZone Protection Controller (TZPC) is configured to allow the protected world to have exclusive control over protected world switches and Input/Output ports. T-User-Interaction guarantees the investigation of safe and protected text input connect to 3 properties. The first security plot (secret) an be managed by the general world's Operating system, secondly, the details shown by the safe world can be accessed by the general world's Operating system, and lastly is the secrets various are never exposed to the general world. In addition, there are three other properties for security analysis, namely the authentication authored is constantly related to the pop-up memo in protected world, the general world can't falsify the user's consent to the message advertised operating any kind of key injection, and finally, the authentication arising in safe world cant left by the general world's Operation system.

Evaluation and Effectiveness

T-User-Interaction architecture is calculated by persuasiveness, easy to use, and completion. Designs are tested on various use cases using real-world applications. Ease of adoption is measured for developers to evaluate the complete use case. Seven open-source applications have been modified, including Elgg (Open Source Social Networking Engine) and Drupal (Open Source CMS). To calculate the strength of closed-source applications, the Operation System is modified for evaluation purposes only. Drupal (on the Ubuntu server) post content type handling is to be modified to verify authentication so it has proven post performance, with grant the users to approve posts on the protected world. Proxy activity is used for this test to be integrated with the confirmatory trust application. The application sends a protected world acknowledgment along with a postal message to the server. The Drupal server verifies authorization before publishing a post.The client in open-source is downloaded by the public GitHub repository and in closed source, the application is downloaded from Google Play. The time spent on modifications and the lines-of-code (L-OC) converted for each application was recorded. For the application to cover login security. In terms of authentication, the authentication logic varies depending on what is being proved. The overall change on the server side is less than 20 lines of code.

Table 1. Results: Open Source Applications (Evaluation)

Test Case	Client	Server	Time Spent
Drupal Attested Post	4 LOC	20 LOC	1 hour
Elgg Attested Payment	4 LOC	12 LOC	30 mins
Elgg Authenticator	3 LOC	4 LOC	30 mins
Drupal Login	3 LOC	4 LOC	30 mins
GNUSocial Login	3 LOC	4 LOC	40 mins
Kandroid Login	3 LOC	4 LOC	30 mins
Redmine Login	3 LOC	4 LOC	30 mins
Owncloud Login	3 LOC	4 LOC	40 mins
Seafile Upload	3 LOC	4 LOC	50 mins

To protect user secrets in a safe world, the application is modified to protect sensitive user data include and other secret files. Closed-source applications are re-bundle by constructing some selected EditTexts in their layout files, so when sensitive data needs to be provided by the user, the TruZ User-Interaction keyboard is invoked and the data is typed in in a protected world. To protect user confirmations in a safe world, the confirmation User-Interaction name (Activity or Activities that contains the AlertDialog) and associated messages are hard-coded file structures.

Table 2. Result: Closed-Source Applications (Evaluation)

Test Case	Login	Payment	Upload	Attestation
Success/Total	13/15	5/5	2/2	9/9

Results

Configuration files are constructed to instruct the HTTP and SSL. 31 applications were pooled with a share of 15 applications used for trust execution environment-protected login, 5 applications for trust execution environment-protected payments, 2 applications for trust execution environment-protected file uploads, and 9 applications for authentication. The reason for the failure, the application is not representative. T-User-Interaction keyboard integration adds 123ms of overhead. Confirmation User-Interaction integration adds 53 ms overhead. In the T-User-Interaction the upward is acquired by the intercommunication among the proxy I-ME application. In the confirmation User-Interaction combination, the upward is due to the communication between the trust execution environment bridge service and the confirmation trust application. Overall, the delay caused by the overhead for T-User-Interaction is barely noticeable when users are interacting with T-User-Interaction.

T-VOIP-C

Security Analysis

The security investigation concludes that TrustZone load sincerity-comfirmable the operatinal processing-trust execution environment oerating systems. The goal of the general world's dangerous Operation System is to get transparent text audio. The Operation System may try to change different stages of the VoIP-C. In the different stages, the scenarios described will not work due to various design properties. During secret phrase entry, the trust execution environment controls the User-Interaction and is notified of this using a secure LED. The Operation System can shoot a trick into that a safe call is proposed, but pass no control to the trust execution environment and emulate a secure User-Interaction shown by the trust execution environment. The Operation System cannot to access the safe LED, which is used to tell the user whether the audio peripheral is controlling the trust execution environment. Therefore, the Operation System cant tricks the user into respecting the initiation of secure calls.

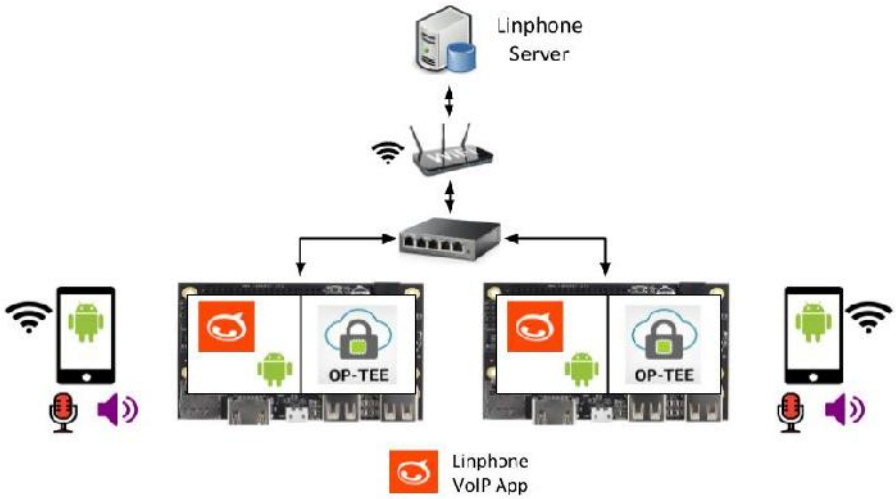

Figure 19 Simulation Setup

The general world's Operation System can affect the counter because the sequence number is sent by the general world. If the same key and counter are used, the XOR of the ciphertext can give a plaintext XOR. The counter calculated in the trust execution environment is copied by the package. Trust Application chwck that the package indicator increases each time. Bit-flipping requires knowledge of parts of plain text. The Chosen-IV attack relies on selecting a particular IV and analyzing the resulting key stream. A general world's Operation System can't observe the keystream because it's in trust execution environment memory. A general world's Operation System could try to replay the sound payload for outgoing packets preserving the previously seen references. The SDP's in-memory audio cache size brings a summary delay since the double index is used again due to index rollover. Trust Application frees memory after the data is used. Reusing old indexes will not result in data resubmission.

Simulation Test Environment

In this simulation test environment is used to test T-VoIP-C. Figure 16 shows how the Simulation trust execution environment platform brings data to report and collects data to repeat. The mix of a bridge and an out of telephone replaces the need for a driver within the trust execution environment Operation System Trust Application. The simulated bridge sends/receives transparent text audio between the external phone and the Trust Application Data trust execution environment, but these components are used for easier prototyping. If the vendor adopts T-VoIP-C, there will be no need for a simulated bridge as the Trust Application will precisely use the audio driver regulated by the vendor in the trust execution environment.

Results

The results reported the moderate of twenty analysis. The overhead added in SRTP is 0.48 miliseconds for open packets and 0.54 miliseconds for approaching packets. This has little impact on overall performance as T-VoIP-C adds a quarter second of average overhead compared to C-off for the end-to-end time during the call. The end-to-end time for C-on is higher because it uses an additional computation stage in the VoIP pipeline, which is not used by secure cases. VoIP-C aspect can be altered by a lot of factors, consisting of packet loss, sound quality, delay, and variation of delay (jitter). For VoIP, 1-2.5% of packet loss is considered acceptable. The calculation consists of a 2%

packet loss in sound quality testing. The mean-opinion-score (MO-S) is good analysis of voice quality. This is a subjective test in which participants rate the quality of a voice transmission system by rating sound quality on a scale of 1 to 5.

Table 3 T-VoIP-C Performance Evaluation

	Non-Secure	Secure
SRTP Time per Outgoing packet (ms)	0.16	0.64
SRTP Time per Incoming packet (ms)	0.12	0.66
End-to-End Time (seconds)	C-off: 4.27 C-on: 5.6	4.51
Audio Input Time (ms / KB)	16.95	18.45
Audio Output Time (ms / KB)	14.31	32.96

Table 4 T-VoIP-C VoIP Quality Evaluation

		C-on	Secure
MOS (no packet loss)		2.1	1.3
MOS (2% packet loss)		2.0	1.2
Correct Answer (no loss)		95%	95%
Correct Answer (2% loss)		98%	81%
	C-off	C-on	Secure
JBM (ms)	55	211	207
IAJ (average)	26.41	27.38	26.12
IAJ (median)	26.5	27.3	26.6
JB (ms)	67.5	89.06	79.26

In T-VoIP-C's calculation, the applicable lags consist of clarification lags and packetization lags. Because the other audio computing level is disabled in safe cases, the delays that occur for these stages are unsystematic. The packetization lag is linked to audio buffering. The average time required for each case is as follows: (1) C-On: 19.98 ms, (2) C-off: 18.08 ms, (3) Secure: 21.23 ms. During a VoIP call, RTP packets may arrive out of sequence and/or at varying intervals. VoIP applications such as Linphone use a jitter buffer to hold incoming packets before the appropriate audio is played, which adds some delay. Because T-VoIP-C needs a trust execution environment at multiple layers of the VoIP heap, trust execution environment calling could count the estimate distortion and commit to jitter. The value corresponds to fifteen minutes call. Secure cases add an average overhead of one point twenty-five milliseconds.

CONCLUSIONS AND RECOMMENDATIONS

This research provides a transparent solution for the operation to cover user interaction on cellular platforms using ARM TrustZone. This research targets audio input User-Interaction and Input/Output user interaction channels. First, this research proposes T-User-Interaction, a clear architecture that grants general-world applications to advantage TrustZone over current Operation System APIs to secure user interactions through User-Interaction input. The architecture uses a cross-Operation System compelling between User-Interaction interactions in the general world of apps, allowing app developers to call a safe version of the User-Interaction and serve code to bind to this User-Interferences. This research further proposes T-VoIP-C, a clear architecture to cover user audio-Input/Output during VoIP calls by integrating a trust execution environment at a critical stage in the audio pipeline of VoIP applications. The design enables VoIP applications to advantage TrustZone during the use of actual operation system APIs and VoIP protocols and brings general Trust Application support so application-specific Trust Application code is not required. Finally, this study proposes T-simulation, an

architecture for a simulation-based trust execution environment prototyping environment. The design uses cross-OS conclusive between trusted applications in the trust execution environment. The hardware is devoted to various operating systems on various boards such as the RaspberryPi and all of the results are armed and tested on TrustZone-enabled Hikey development boards.

More analysis will be needed on how to grant the characterizing of User Interface characteristics within WebViews as safe and how to set up cross-operation systems conclusive to grant allusion results from trust execution environment to be exchanged clearly to website utilization code within WebViews. In the VoIP computing stage in the trust execution environment, there is a demand for light audio computing channels in the trust execution environment which can accomplish adequate audio quality advancement. The architecture for this needs to converse suitable TCB exchange on trust execution environment versus compatible audio quality for VoIP calls. Lastly, for future research, it is necessary to expand hardware simulation support as the current test associate apart one category of incidental at a time. More tests can be completed to establish the reproduction is balanced and adequate to backing cases where various factors need to pervade because no work exists to adopt SPI in captive mode on the Raspberry Pi.

BIBLIOGRAPHY

A. Amiri Sani, K. Boos, MH Yun, and L. Zhong. Rio: A System Solution for Sharing Input/Output between Mobile Systems. In Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services, MobiSys 14, page 259272, New York, NY, USA, 2014. Association for Computing Machinery. ISBN 9781450327930. doi: 10:1145/2594368:2594370. URLs https://doi:org/10:1145/2594368:2594370.

A. Barth, C. Jackson, and C. Reis. The Security Architecture of the Chromium Browsers. Technical report, 2008. URL http://css:csail:mit:edu/6:858/2018/readings/chromium:pdf .

D. Liu and LP Cox. VeriUI: Attested Login for Mobile Devices. In Proceedings of the 15th Workshop on Mobile Computing Systems and Applications, HotMobile' 14, Santa Barbara, CA, USA, February 26-27 2014.

D. Zhang. Trustfa: TrustZone-Assisted Facial Authentication on Smartphones. Technical report, 2014. URL http://www:donglizhang:org/trustfa:pdf .

M. Ender , G. Duppmann, A. Wild, T. Poppelmann, and T. Guneysu. A Hardware-Assisted Proof-of-Concept for Secure VoIP Clients on Untrusted Operating Systems. In Proceedings of 2014 International Conference on ReConFigurable Computing and FPGAs, ReConFig' 14, Cancun, Mexico, Dec 8-10 2014. URL https://doi:org/10:1109/ReConFig:2014:7032489 .

R. Buhren, J. Vetter, and J. Nordholz . The Threats of Virtualization: Hypervisor-Based Rootkits on the ARM Architecture. volume 9977, 11 2016. doi:10:1007/978-3-319-50011-929.

S. Oh, H. Yoo, DR Jeong, DH BUser-interference, and I. Shin. Mobile Plus: Multi-Device Mobile Platform for Cross-Device Functionality Sharing. In Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services, MobiSys 17, page 332344, New York, NY, USA, 2017. Association for

Computing Machinery. ISBN 9781450349284. doi: 10:1145/3081333:3081348. URL https://doi:org/10:1145/3081333:3081348 .

W. Li, M. Ma, J. Han, Y. Xia, B. Zang, C.-K. Chu, and T. Li. Building Trusted Path on Untrusted Device Drivers for Mobile Devices. In Proceedings of 5th Asia-Pacific Workshop on Systems, APSys '14, pages 8:1–8:7, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-3024-4. doi: 10:1145/2637166:2637225. URL http://doi:acm:org/10:1145/2637166:2637225 .

X. Li, H. Hu, G. Bai, Y. Jia, Z. Liang, and P. Saxena. DroidVault: A Trusted Data Vault for Android Devices. In Proceedings of the 2014 19th International Conference on Engineering of Complex Computer Systems, ICECCS' 14, Tianjin, China, Aug 4-7 2014. URL https://doi:org/10:1109/ICECCS:2014:13 .

Y. Jong, P. Hsiu, S. Cheng, and T. Kuo. A Semantics-Aware Design for Mounting Remote Sensors on Mobile Systems. In 2016 53rd ACM/EDAC/IEEE Design Automation Conference (DAC), pages 1–6, 2016. URL https://ieeexplore:ieee:org/document/7544382 .

Y. Li and W. Gao. Interconnecting Heterogeneous Devices in the Personal Mobile cloud. In IEEE INFOCOM 2017 - IEEE Conference on Computer Communications, pages 1–9, 2017. URL https://ieeexplore:ieee:org/document/8057083 .