Information security of the 2016 Philippine automated elections, a case study

Jeffrey Ian C. Dy

Technical Report

RHUL–ISG–2022–4

11 April 2022

Information Security Group
Royal Holloway University of London
Egham, Surrey, TW20 0EX
United Kingdom

Student Number: 180258385
Jeffrey Ian C. Dy

Information Security of the 2016 Philippine Automated Elections, a Case Study

Supervisor: Prof. Konstantinos Mersinas, Ph.D.

Submitted as part of the requirements for the award of
MSc. Information Security at
Royal Holloway, University of London

March, 2021.

Anti-Plagiarism Declaration

I declare that this project report is all on my own words and that I have acknowledged all quotations from published or unpublished work of other people. I also declare that I have read the statements on plagiarism in Section 1 of the Regulations Governing Examinations and Assessment Offenses, and in accordance with these regulations, I submit this project report as my own work.

Jeffrey Ian C. Dy
March, 2021

Acknowledgement

This work is dedicated to God above all else. From whose Grace emanates all wisdom and knowledge. It is said that knowledge comes with purpose. This work is offered to my fellow Filipinos who will once more exercise their right of suffrage and cast their vote for the 2022 presidential elections.

My wife Merriam, my editor, who provided me the inspiration to continue with any endeavor despite the difficulties amidst the pandemic. Together with my three kids Xandy, Karl and Joanne, and our dog Rocket, they provided the necessary support system for me to persevere and soldier on. Special thanks to my project supervisor, Prof. Konstantinos Mersinas, Ph.D. for his invaluable support. It was difficult keeping in touch considering the circumstances, and yet this report was completed with his assistance.

The 23rd President of the Senate of the Republic of the Philippines, Hon. Vicente C. Sotto, III gave approval to use the log files, and other documents submitted by the Commission on Elections (COMELEC) to the Senate and the Joint Congressional Oversight Committee on the Conduct of the Automated Elections (JCOC). This project would not have been completed without the resources given to me through his office. I am thankful to his staff Atty. Marian Lucinario and Bro. Hutch Altavas. Also to the Senate Committee on Electoral Reforms chaired by Sen. Imee Marcos, and its Committee Secretary, Atty. Dana Mendiola-Alberto. Former Sen. Ferdinand "Bongbong" Marcos Jr. and his wife, Atty. Liza Marcos, also provided some valuable documents for this research.

There were like-minded individuals who I discussed the contents of these report as well: Atty. Hubert Guevara and his election source-code review team; Atty. Glenn Chong; Leo Querubin; Dr. Nelson Celis; and, Atty Ivan Uy. The informal discussions we had contributed to this work.

Table of Contents

List of Figures

List of Tables

List of Acronyms

AES	Automated Election System
APN	Access Point Name
AR	Audit Returns
BEI	Board of Election Inspectors
BOC	Board of Election Canvassers
BSP	Banko Sentral ng Pilipinas
CAC	COMELEC Advisory Council
CCS	Consolidation and Canvassing System
COC	Certificate of Canvass
COCP	Certificate of Canvass and Proclamation
COMELEC	Commission on Elections
COTS	Commercial Off The Shelf
CVL	Computerized Voters List
DepEd	Department of Education
DHCP	Dynamic Host Control Protocol
DICT	Department of Information and Communications Technology
DNS	Domain Name Service
DOST	Department of Science and Technology
DRE	Direct Recording Electronic machine
EAC	Election Assistance Commission
EDCVL	Election Day Computerized Voters List
EMS	Election Management System
EO	Election Officer
ER	Election Return
GOCC	Government Owned and Controlled Corporations
ICE	International Certification Entity

JCOC	Joint Congressional Oversight Commitee on the Conduct of the Automated Elections
MBOC	Municipal Board of Canvassers
MODEX	Modernization and Excellence Program of COMELEC
MOV	Minutes of Voting
MPLS	Multi Protocol Label Switching
NAMFREL	National Movement for Free Elections
NAT	Network Address Translation
NGO	Non-Government Organization
NLE	National and Local Elections
NPO	National Printing Office
NTSC	National Technical Support Center
OMR	Optical Mark Reader
PAT	Port Address Translator
PBR	Policy Based Routing
PCOS	Precinct Scan Optical Scanner
PCVL	Precinct Computerized Voters List
POP	Project of Precincts
PRELAT	Pre-election Logic and Accuracy Testing
PVL	Permanent Voters List
RA	Republic Act
RMA	Random Manual Audit
SAES	Smartmatic Auditable Election System
SIM	Subscriber Identity Module
SOV	Statement of Votes
TDP	Technical Deployment Package
TEC	Technical Evaluation Committee
TOCTOU	Time of Check Time of Use
TOR	Terms of Reference

TWG	Technical Working Group
VCM	Vote Counting Machine
VIN	Voters Identification Number
VIU	Voter Identity Unit
VPN	Virtual Private Network
VRR	Voter Registration Records
VVPAT	Voter Verification Paper Audit Trail
VVSG	Voluntary Voting System Guidelines
WORM	Write Once Read Many Secure Digital (SD) Cards

Executive Summary

The Philippines held its first fully automated national and local elections in 2010, after 18 years of deliberation on transitioning from manual counting of ballots. While the 2010 automated elections was a success, some criticisms emerged and points for improvement that were later applied in the 2016 Philippine presidential elections. This research examined the issues surrounding the 2016 automated presidential elections with the aim of recommending improvements for the Philippine Automated Election System (AES) for the next presidential elections in 2022.

To assess the 2016 presidential elections, the research developed and used an Automated Election System Trust Model that included the properties of: (1) voter privacy; (2) uncoercibility / receipt-freeness; (3) individual verifiability; (4) universal verifiability; (5) fairness; (6) data integrity; (7) availability; and (8) non-repudiation. Analysis of 426 log files of Vote Counting Machines (VCM) and Consolidation and Canvassing System (CCS) covering 192 clustered precincts were compared to news clippings, case pleadings, transcripts of the meetings of the JCOC and its Technical Working Group (TWG), and various laws, rules and regulations to create a more holistic picture of the 2016 automated presidential elections. In the scoring, the AES failed in 5 out of 8 properties of the Trust Model. Lack of transparency was seen as the major factor, thereby urging the adoption of a more transparent approach to certification and source code review in evaluating the Philippine AES.

Keywords: Philippines, elections, automated elections, eVoting, electoral fraud, Smartmatic, Commission on Elections.

1. Introduction

1.1. The Philippine Journey Towards Automated Elections

A comprehensive recount of the Filipino people's perception of the credibility of the government to conduct elections before and after the birth of the Commission on Elections (COMELEC) can be found in Cleo Calimbahin's dissertation [1]. Patronage politics is at the center of Philippine politics. The use of *guns, goons, and gold* is well documented. Guns and goons allude to the use of private armies to harass political opponents, while gold alludes to the use of "vote-buying" to secure election victory.

Calimbahin opined that the most fraudulent election in the country happened in 1986, particularly during the Snap Election held on January 17, 1986. The Snap Election was to decide if then Pres. Ferdinand E. Marcos will continue to serve following a 20-year term, or will he be unseated by his main rivals: United Opposition (UNIDO) candidate Salvador "Doy" H. Laurel and an independent candidate Mrs. Corazon "Cory" C. Aquino. At the last minute, the veteran politician Laurel teamed up with Cory Aquino as her Vice-President.

To improve the Snap Election's credibility, the Philippine legislature passed Batas Pambansa (BP) No. 881, the *Omnibus Election Code of 1985*. The law institutionalized the accreditation of a non-government organization (NGO) to act as an official election "citizen's arm" [2]. The citizen's arm has all the rights and privileges of a major political party. The National Movement for Free Elections (NAMFREL) took this role. On February 8, 1986, NAMFREL reported Aquino ahead by a million vote, but the official COMELEC tally showed Marcos winning by a slim margin. The following day, some members of the COMELEC Board of Canvassers (BOC) walked-out of the national canvassing. They reported that they were instructed by Col. Pedro Baraoidan, Director of the National Computing Center (NCC), to hand over the official Election Returns (ER) in exchange for alleged photocopies whose results did not match the original [3]. COMELEC continued the count despite the walk-out, and on February 15, proclaimed Ferdinand Marcos, Sr. and Arturo Tolentino as President and Vice-President-elect, respectively. Buoyed by the growing social upheaval, Defense Minister Juan Ponce Enrile and Gen. Fidel Ramos of the Philippine Constabulary led a coup d'ètat on February 22, 1986. This led to the EDSA People Power Revolution that forced Pres. Marcos to exile until his death in Hawaii, USA.

The tools of cheating during the 1986 Snap Elections included padded voters list, intimidation, vote-buying, and non-traceable disappearances of ballot boxes en route to canvassing centers. In the years that followed, the Filipino people's faith in elections and in COMELEC was low. Pres. Aquino had to make substantive changes to win back the people's trust in the electoral process. She successively appointed three COMELEC chairpersons during her term, culminating with Atty. Christian Monsod, a member of the 1987 Constitutional Convention and former chair of NAMFREL. Republic Act 6646, the Electoral Reforms Law of 1987, was passed. The new law reflected the 1987 Constitution that had the country move away from a two-party system to a multi-party system.

Immediately after the 1992 Presidential Election, COMELEC Chairperson Monsod started project MODEX[1]. The project's main objective was to push for automation of the 1998 Presidential election. Project MODEX secured legislative consent to finance the piece-by-piece automation of the different stages of election preparation. MODEX successfully led to the passage of a law mandating the computerization of the voters' list, and in starting the process for continuing voters registration [4].

COMELEC submitted a draft automated election bill in 1993, which then Pres. Fidel V. Ramos certified as "urgent". The bill became law as RA 8047 on June 7, 1995. The law authorized COMELEC to conduct a pilot testing of an Automated Election System (AES) for the 1996 special elections for the Autonomous Region of Muslim Mindanao (ARMM). The law was explicit that the Optical Mark Reader (OMR) or Precinct Count Optical Scanners (PCOS) technology will be used [5].

During the plenary hearings, the use of Direct Recording Electronic (DRE) voting machines over OMR was debated. In the end, both COMELEC and Congress felt that fewer changes (e.g., by maintaining the use of paper ballots) would make the AES more acceptable to the public.

Riding on the success of the pilot testing, COMELEC lobbied for full automation of the 1998 presidential elections. In the Senate, the Committee on Electoral Reforms was chaired by Sen. Miriam Defensor-Santiago who argued that

[1] MODEX stands for "Modernization and Excellence Program for COMELEC"

automation, if not properly done, would just replace the old with new ways of cheating [6]. The senator maintained that for automation to be successful, it should be balanced with the right policy. As COMELEC prepared for a negotiated contract for purchase of an AES, arguments were also raised on whether it would be better for the country to adopt a nationally developed open-system AES rather than a purchased or leased proprietary system.

On December 22, 1997, RA 8436 was passed. It mandated the automation of the May 11, 1998 presidential elections [7]. Despite the passage of the law, there was simply not enough time to prepare. Hence, the 1998 presidential elections remained manual.

Nine years later, on January 23, 2007, RA 9369, the Automated Election Law of 2007, was enacted. The law became the basis for the automation of the 2010 and the succeeding National and Local Elections (NLE) [8].

1.2. Issues with the Philippine Automated Election Systems

The 2010 automated election was generally lauded as a success although some critics noted that, while the event was more efficient and transparent, it failed to resolve central problems plaguing the Philippine political system. Violence, vote-buying, and personality centered politics remained prevalent [6]. The research of [9] concluded that automation did not change patronage-politics that is centered on how much a candidate spent and his/her ties with a powerful political dynasty to win an election.

The type of technology and how it was used were also heavily criticized. The principal author of the Automated Election Law of 2007, Sen. Richard Gordon filed a case before the Supreme Court against COMELEC for not including the Voter Verification Paper Audit Trail (VVPAT) functionality in the 2010 AES. COMELEC argued that the ballot already served as a VVPAT, furthering that VVPAT is not really required in a paper-based election [10].

Former COMELEC chairperson Atty. Monsod additionally observed that COMELEC made it difficult for the accredited citizen’s arm and the IT experts to get answers to legitimate inquiries [11]. He questioned why, even with automation, COMELEC policies still prohibited voters from correcting their votes [12]. In 2010, until today, voters are not allowed to get another ballot if they made a mistake filling up ballots.

In [13], Monsod said that the aim of automation was to minimize human discretion especially during the counting and canvassing of votes. He was surprised that the tiered canvassing of votes (explained in Chapter 2.1.) remained unchanged. He opined that this old practice allowed for "wholesale cheating". Vote-buying was not curbed by not allowing ERs to be directly sent to a central counting server. To aid in the automation, Monsod also proposed that the President appoints an IT professional as one of the COMELEC Commissioners.

Although Sen. Gordon won the VVPAT case in time for the 2016 presidential elections, the same issues and more were raised in the 2016 AES. Critics argued that there was a need to certify the transmission components of the AES, such as the "meet-me-room" by the Technical Evaluation Committee (TEC) certification [14] (explained in Chapters 2.4 and 4.1).

Media companies and political parties accessing the Transparency Server noticed that results were not being received on the evening of May 9, 2016. Smartmatic engineers later confessed of an error that they needed to fix. The "fix", however, was not authorized by COMELEC, which led to several foreign and Filipino Smartmatic and COMELEC personnel to be indicted for violating Sections 4.a.1, 4.a.3, and 4.a.4 of RA 10175 or the Philippine Cybercrime Law of 2012[2] [15] [16].

Despite these, COMELEC declared that the 2016 election was a success. It had the most voter turn-out at 81.95%. A total of 44.55 million Filipinos, out of the 54.36 million registered voters, voted [17]. The results of the Random Manual Audit (RMA) of the 2016 elections also showed the machines to be 99.9023% accurate [18].

But two years later, Senator Vicente Sotto III delivered two privilege speeches in the Senate alleging the following [19] [20] [21]:

1. Transmissions from the VCM to the official Municipal CCS were made earlier than the election day.
2. An individual outside the Philippines was accessing the election servers.
3. The existence of a "secret" transmission server which did not undergo the mandated certification process.

In response to the allegations, the Joint Congressional Oversight Committee on the Conduct of Automated Elections (JCOC) conducted investigations of the 2016 presidential elections. For the first time since the country's elections became

[2] These sections referred to: (1) illegal or unauthorized access to a computer system; (2) data interference or the intentional or reckless alteration of data; and, (3) system interference or the intentional or reckless alteration of how a computer system functions, respectively.

fully automated, computer logs of the AES were provided to JCOC by the COMELEC. The manner of operations of the AES seemingly remained incomprehensible to the legislators as they repeatedly requested computer logs that did not do much to help in determining what really happened during the election. COMELEC capitalized on this general unawareness and only provided specifically what was requested. When asked to liberally provide *all relevant logs*, COMELEC insisted on the need for secrecy to preserve the security of the AES [14].

The results of the JCOC investigations prompted Sen. Sotto to file Senate Bill No. 7 or the "Hybrid Election Bill". The bill proposed manual counting in election precincts, while the preparation of the ER, Certificates of Canvass (COC), and their transmission will be electronic. Sen. Sotto doubted the accuracy of computers and machines to count ballots, implying limited transparency on the protocols used by the machines to count votes [22]. On December 14, 2020, Sen. Imee Marcos, chairman of the Senate Committee on Electoral Reforms, proposed an improvement to the bill filed by Sotto [23]. If passed, Philippines will be among the countries, such as Paraguay, Germany, and Netherlands, reverting from a fully automated election to a hybrid or semi-manual election [24].

1.3. Research Objectives and Report Structure

Focusing on the 2016 presidential elections, this research aimed to determine the level of integrity of the Philippine AES by developing and using an *AES Trust Model*. The same computer audit logs submitted by COMELEC to JCOC were examined to trace what transpired during the elections. The findings were then compared to news clippings, case pleadings, transcripts of the meetings of the JCOC and its Technical Working Group (TWG) and various laws, rules and regulations to create a more holistic picture of the 2016 automated presidential elections.

Chapter 2 details the conduct of Philippine automated elections, which serve as the reference point for most of the analysis in this research.

Chapter 3 describes the methodologies employed. First was the development of an AES Trust Model for the assessment of the 2016 AES. Second was the extraction of data from the election audit logs using programs coded in python 3.

In Chapter 4, significant events of the 2016 NLE based on the extracted data were analyzed. Based on these, the integrity and security of the 2016 AES were assessed using the AES Trust Model.

Conclusions and recommendations to improve succeeding AES are listed in Chapter 5. Recommendations for further research are highlighted in Chapter 6.

1.4. Research Limitations

This research did not perform a forensic analysis of the submitted audit logs of COMELEC. The logs were analyzed as is, and no study was made as to the authenticity of the log files.

The research focused on the general conduct of the 2016 NLE. No conclusion was therefore formulated regarding the merits of specific election protests.

2. Philippine Automated Election Components and Procedures

Chapter 2.1 is dedicated to defining the terms repeatedly mentioned throughout the report. The three main components of the 2016 Automated Election System (AES) are then discussed in Chapter 2.2, followed by a description of various election preparation and voting procedures in Chapter 2.3. Chapter 2.4 discusses the statutory requirements for certification of an AES.

2.1. Definition of Terms

Prior to every election, COMELEC appoints members of the Board of Election Inspectors (BEI) that consists of a chairman, a poll clerk, and one member. They collectively decide on issues at the polling precinct level. The BEI is composed of public school teachers [25]. They are responsible for all election activities at the precinct, including the conduct of elections, printing and transmission of Election Returns (ER), and the certification of the Election Day Computerized Voters List (EDCVL).

A polling precinct is a “unit of territory for the purpose of voting” [2 Art XIII, Sec. 149]. Each precinct should have no more than 200 registered voters [26].

During the 2016 automated elections, up to seven (7) precincts with a total number of registered voters not exceeding 800 were grouped together to form a clustered precinct [27]. A precinct or a clustered precinct is situated in a public school classroom. The list of precincts, clustered precincts, and their assigned public school classroom can be found in the election Project of Precincts (POP) and is available online in [28]. The 2016 POP reported 92,509 clustered precincts.

The physical layout of a clustered precinct should be clear of obstructions. The voting booths, ballots, Vote Counting Machine (VCM), and other election paraphernalia should always be in full view of the BEI and poll watchers. Because the secrecy of the ballot is sacred by law, voting booths consider the privacy of voters.

Candidates, political parties, the citizen's arm, and regional and coalition parties may each assign a poll watcher to every precinct and every canvassing center. Only poll watchers and COMELEC-authorized personnel are allowed within 50 meters of the poll area and the canvassing centers. The police or any state troop are prohibited to enter the area unless they are called by the BEI, BOC, or COMELEC to attend to a security issue or emergency. During the 2016 NLE, taking photographs of filled-out ballots, Voter Verification Paper Audit Trails (VVPAT) or of voters including taking selfies was not allowed inside the precincts and can be considered an election offense [29].

The ER is the summary of the votes counted per candidate in one precinct or clustered precinct. It is printed and transmitted by the VCM to the Consolidation and Canvassing System (CCS) and other servers. The ER is authenticated by the BEI who can be held liable for any inaccuracies.

The consolidation of all the ER belonging to one municipality or city is called canvassing of votes. The Board of Canvassers (BOC) has complete authority during the canvassing of votes. To proclaim winners at each level of the canvassing hierarchy, the following must be printed and certified by the BOC:

1. *Statement of Votes (SOV)* - a document summarizing the total votes of each candidate per contest (e.g., presidential contest, vice president contest);
2. *Certificate of Canvass (COC)* - a document containing a summary of all the votes counted per candidate per contest per political administrative unit (e.g. precinct, city, or province)

3. *Certificate of Canvass and Proclamation (COCP)* - a document containing the summary of votes for candidates and the winner for each contest for that political administrative unit.

The composition of the BOC varies according to the canvassing level. The Municipal or City BOC performs canvassing upon receipt of ER through their Consolidation and Canvassing System (CCS). Once complete, they generate the SOV, COC and COCP, and then transmit the canvassing results to the next level CCS. Table 1 summarizes the BOC composition per level according to [30] and [31].

Table 1. Levels of Canvassing and the Board of Canvassers

Level	BOC composition	What they consolidate	What they proclaim in the COCP	What they transmit
Municipal/ City	Municipal/City Election Officer (EO), Municipal/ City Treasurer, District School Supervisor	ER from each clustered precinct	Mayor, Vice Mayor, City Councilors, Legislative District Representatives	Municipal SOV and COC
Provincial	Provincial EO, Provincial Prosecutor, District School Superintendent	Municipal/City canvassing results	Governor, Vice-Governor, Provincial Board	Provincial SOV and COC
Regional for Autonomous Region for Muslim Mindanao (ARMM)	Regional Election Director, Equivalent Rank from the Department of Justice (DOJ) and Department of Education (DepEd)	Municipal/City canvassing results	ARMM officials	Regional SOV and COC
National	Commissioners of COMELEC	Provincial or Regional canvassing results	Senators, Party-list Representatives	None
Presidential	Representatives from both the Senate and House of Representatives	Provincial and Regional canvassing results	President, Vice-President	None

2.2. Components of the 2016 Automated Election System

The Philippine AES since 2010 uses the Optical Mark Reader (OMR) technology. OMRs had been extensively used since the 1970s in education, medicine, and government census, among others. A discussion on OMR technology can be found in [32].

The AES supplier during the 2016 NLE was Smartmatic, Inc. The Smartmatic Auditable Election System (SAES) model 1800*plus* had three main components: (1)

Election Management System (EMS); (2) Vote Counting Machine (VCM); and, (3) Consolidation and Canvassing System (CCS). The system specifications of each component of the SAES discussed in this sub-chapter were based on [33].

2.2.1. Election Management System (EMS)

The EMS is an application for producing the configuration files for the VCM, CCS, and their authentication equipment (iButtons and USB security tokens). The ballot layouts per clustered precinct are also produced through the EMS.

To create the configuration files for each VCM and CCS, the Project of Precincts (POP) is first loaded. The root certificate and master keys for the election are also uploaded to the EMS. The EMS generates the session and encryption keys, digital signatures, and authentication keys for the election. Unfortunately, security protocols used in the 2016 AES, such as the cryptographic primitives, shuffling algorithm, and random key generators, were not published by COMELEC.

For each of the clustered precincts in the POP, a VCM is assigned and configured. The iButtons and their corresponding pin codes to unlock secure functions of the VCM are also configured through the EMS. The USB security tokens containing the private keys of BOC members and the master database and application to uniquely configure each CCS are produced through the EMS.

In 2016, the EMS application and databases were installed in servers inside the "memory card configuration" room where configuration files for the VCM, CCS, iButtons, and USB token authenticators were produced. The EMS application was accessed via a browser and had a dedicated LAN disconnected from other networks. There was an air-gap between the EMS and hundreds of VCM and CCS being configured just a few meters outside the memory card configuration room. The EMS was among the most guarded systems of COMELEC.

The 2016 SAES 1800*plus* EMS used Ubuntu 14.04 LTS as its operating system and JBOSS 7.1.1 for its application container. It used Java 6 system runtime environment. The database was Oracle 11 Standard One.

2.2.2. The Vote Counting Machine (VCM) and its Election States

Appendix 1 shows the various parts of the SAES1800*plus* VCM used in the 2016 presidential elections. This VCM was rated by the International Certification Entity

(ICE) to be capable of running in a 12V car battery for 14 hours and can withstand partial dusting and liquid spills.

The configuration files and the operating system of the VCM are stored in the SD card in slot A. As the VCM boots, the SD card in slot A is checked. If the SD card slot A has the correct configuration files and the SD card in slot B is empty but formatted exactly as the SD card in slot A, data from SD card A is copied to B. Thereafter, every write instruction, such as when storing ballot images scanned by the VCM, results in data being written in both SD cards A and B. If at any point data is not synchronized between the two SD cards, the VCM generates an error.

When the VCM is instructed to "lock" the SD card, the VCM attempts to lock the SD card in slot B. If the SD card in slot B is WORM[3] capable, it will be locked. Otherwise, the lock will simply fail, and the VCM will notify the BEI that the lock instruction failed.

During elections, the SD cards have a copy of the ballot images fed into the machine. To satisfy the properties of anonymity and unlinkability, the ballot images have identical timestamps. It is unclear if the VCM shuffles the images and what shuffling algorithm is used.

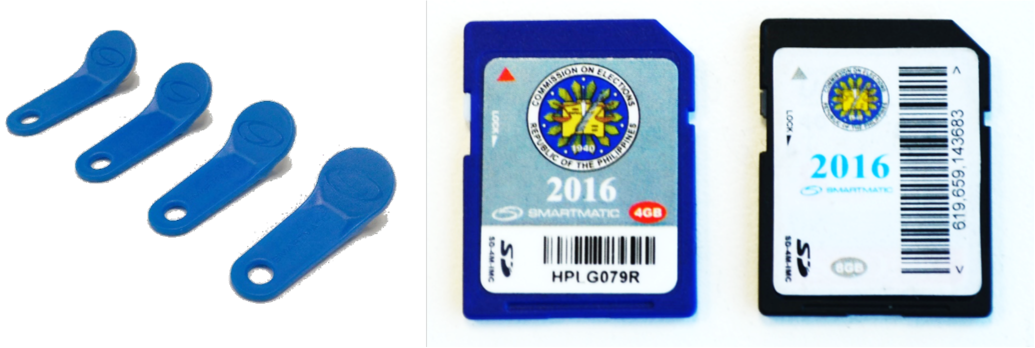
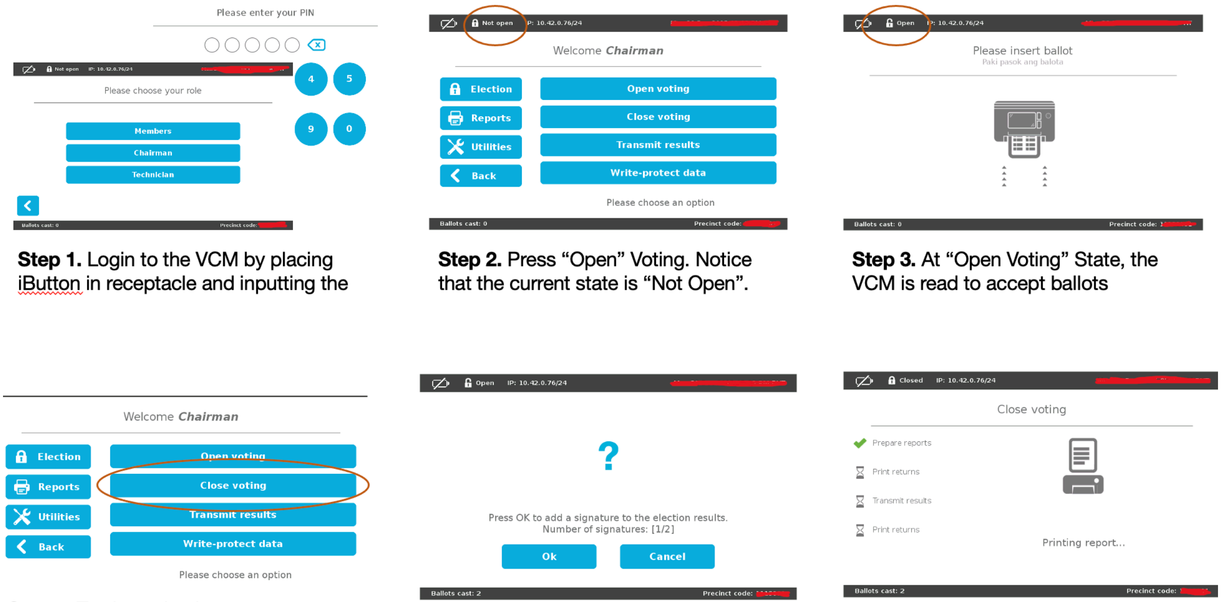


Figure 1. Official iButtons and SD Cards used in 2016 NLE (Blue SD Card is WORM)

The VCM has three election states. Changing the election state is a secure function that can only be done by the chairperson of the BEI and at least one other board member who insert their iButtons into the receptacle and enter their respective pin codes [34]. The iButtons contain the private keys of each of the BEI

[3] WORM stands for "Write Once Read Many"

members. Figure 2 shows how to change the election states from the VCM touch screen.



Press OK to add a signature to the election results.
Number of signatures: [1/2]

Step 1. Login to the VCM by placing iButton in receptacle and inputting the

Step 2. Press "Open" Voting. Notice that the current state is "Not Open".

Step 3. At "Open Voting" State, the VCM is read to accept ballots

Step 4. To close election, go to "Election" Menu and press "Close Voting"

Step 5. BEI to place iButton in receptacle to "digitally sign" the ER

Step 6. VCM will execute a series of closing procedures, then will proceed to "Close Voting" State

Figure 2. SAES 1800*plus* VCM Election States

The three election states of the VCM are:

1. *Not Open Voting* - there are no ballot images in the SD card, and the ballot count is zero;

2. *Open* Voting- the VCM is ready to accept ballots, and to perform election day related functions;

3. *Closed* Voting - the election is closed and no ballots can be inserted into the VCM. This also meant that the VCM's SD cards contain the ballot images, and the official Election Return (ER) created by the VCM when the BEI closed the election is stored in the SD cards. The ER can be transmitted or printed only at this state.

When the BEI commands the VCM to close the election, the following chain of events commences (step 6 in Figure 2):

a. The VCM prompts each member of the BEI to digitally sign the ER through the iButton.

b. The VCM prints the first eight copies of the ER.

c. The VCM gives the option to transmit the ER.

d. The VCM verifies if 22 copies of the ER will be printed.

Once the VCM election state is “closed voting” it cannot be placed in an “open voting" state. Only by performing a secure function called “re-zero” during the “closed voting” state can the VCM return to a “not open voting” state from which the VCM can then be set to an “open voting” state. When re-zeroed, the VCM will erase all ballot images in the SD card, including the ER. All ballot counters and statistics are reset to zero. However, a re-zero function does not work on a locked WORM SD card.

To transmit the ER, the BEI inserts the official USB modem. These USB modems are USB data dongles with Subscriber Identity Module (SIM) cards issued by a telecommunications provider. During the 2016 NLE, there was a contingency transmission medium available per VCM. This can be another USB modem containing a SIM card from a different telecommunications provider or a BGAN satellite modem. For the latter, the ethernet port at the side of the VCM is used.

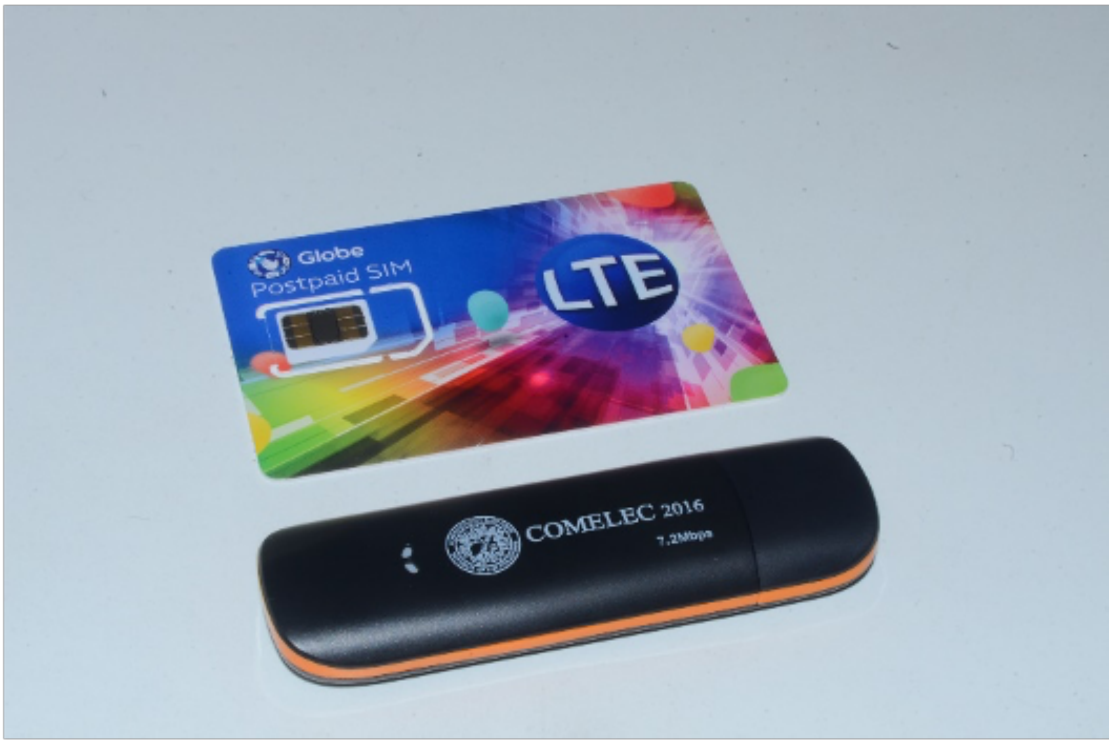


Figure 3. Official 3G USB Dongles for 2016 NLE

As part of the COMELEC continuity plan specified in [35], if the main SD card in Slot A fails, the WORM SD card is removed from Slot B and placed in Slot A. A contingency card is then inserted in Slot B. If the back-up WORM SD card in Slot B fails, it is simply replaced by the contingency WORM SD card. If both SD cards fail, another set of contingency SD cards is used. The VCM can also be replaced. Voting continues while the BEI waits for the replacement VCM. Filled-up ballots are temporarily placed in a folder. When the replacement VCM arrives, the SD cards from the defective VCM are inserted into the replacement VCM. The BEI then prints the statistics to show to the poll watchers that the ballot count is the same as that in the defective VCM. All unfed filled-up ballots are scanned to the replacement VCM by the BEI. This process concerning the VCM makes it prone to an SD card swap attack discussed in Chapter 4.3.2.

2.2.3. Consolidation and Canvassing System (CCS)

The CCS are commercial off the shelf laptops running on Ubuntu 14.0.4 LTS operating system with the SAES 1800*plus* CCS software installed. The 2016 SAES CCS software required Java 6 runtime environment. Based on submitted documents by Smartmatic, the CCS used an Oracle 11 Express Edition.

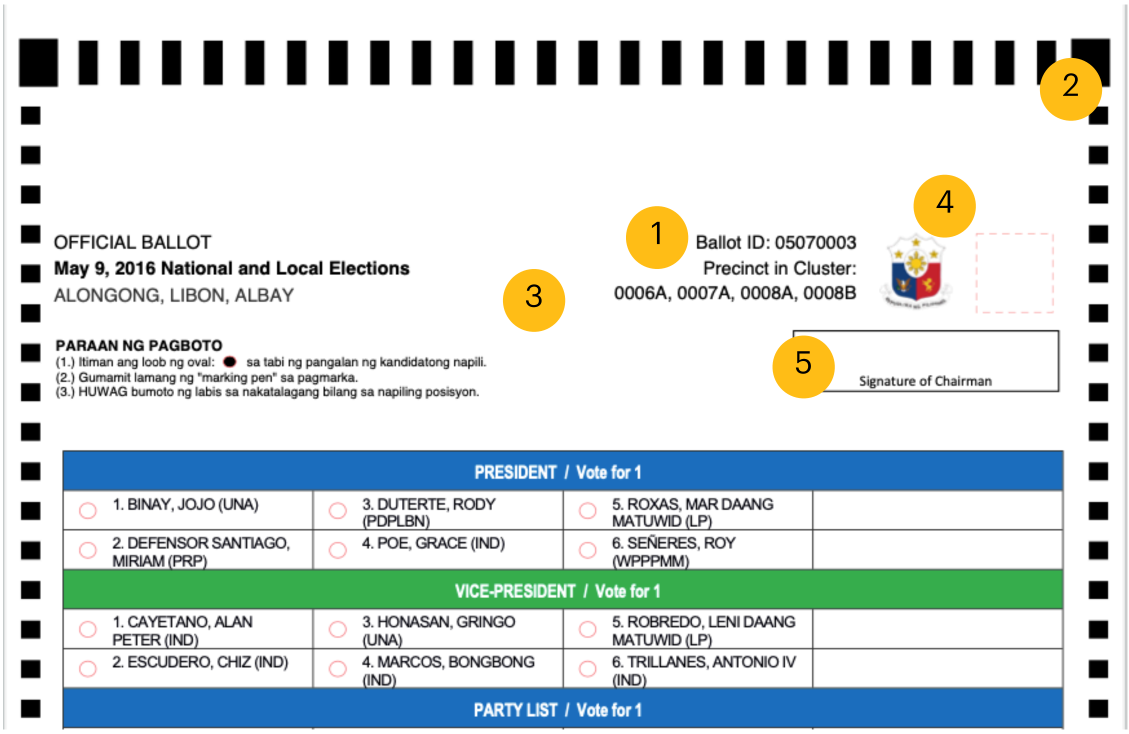
Like the VCM, a specifically configured CCS is issued to each Board of Canvassers (BOC) at various tiers or levels of the hierarchy, as shown in Table 1 (Chapter 2.1). BOC members are issued USB security tokens containing their private keys. To open the CCS, the BOC provides their passcodes. Various secure functions, such as to print the COC, SOV, and COCP, require that BOC members insert their USB tokens and input a different set of passcodes.

COMELEC has not yet declassified any documents pertaining to the security hardening procedures performed on the CCS during the 2016 NLE. The cryptographic algorithm used to sign the COC, SOV, and COCP and to secure the communication channels when transmitting the canvassing results remain confidential.

2.2.4. Ballots

Figures 4 and 5 show the top and bottom portions of the front page of the official ballot template for VCM 05070003 in 2016, as taken from the COMELEC online database in [36].

Figure 4. Top-front portion of the 2016 NLE Ballot



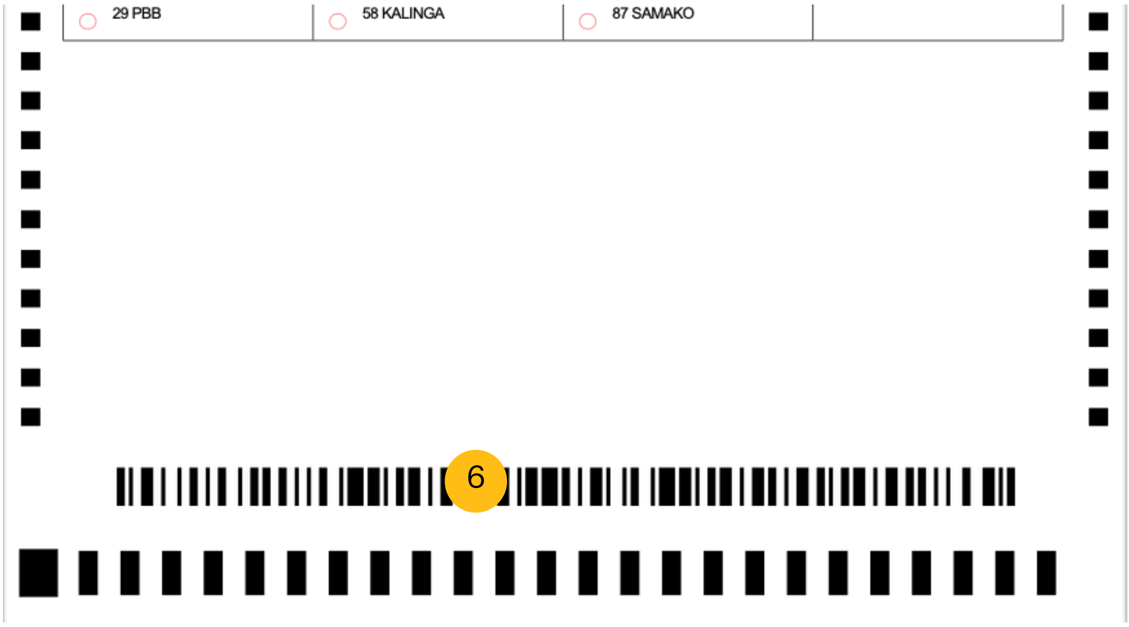
OFFICIAL BALLOT
May 9, 2016 National and Local Elections
ALONGONG, LIBON, ALBAY

Ballot ID: 05070003
Precinct in Cluster:
0006A, 0007A, 0008A, 0008B

Signature of Chairman

PARAAN NG PAGBOTO
(1.) Itiman ang loob ng oval: ● sa tabi ng pangalan ng kandidatong napili.
(2.) Gumamit lamang ng "marking pen" sa pagmarka.
(3.) HUWAG bumoto ng labis sa nakatalagang bilang sa napiling posisyon.

PRESIDENT / Vote for 1			
○ 1. BINAY, JOJO (UNA)	○ 3. DUTERTE, RODY (PDPLBN)	○ 5. ROXAS, MAR DAANG MATUWID (LP)	
○ 2. DEFENSOR SANTIAGO, MIRIAM (PRP)	○ 4. POE, GRACE (IND)	○ 6. SEÑERES, ROY (WPPPMM)	
VICE-PRESIDENT / Vote for 1			
○ 1. CAYETANO, ALAN PETER (IND)	○ 3. HONASAN, GRINGO (UNA)	○ 5. ROBREDO, LENI DAANG MATUWID (LP)	
○ 2. ESCUDERO, CHIZ (IND)	○ 4. MARCOS, BONGBONG (IND)	○ 6. TRILLANES, ANTONIO IV (IND)	
PARTY LIST / Vote for 1			



○ 29 PBB	○ 58 KALINGA	○ 87 SAMAKO	

Figure 5. Bottom-front portion of the 2016 NLE Ballot

Table 2. Description of Different Parts of the Ballot

Ref	Part Name	Description
1	Ballot ID header	The header contains the ballot ID that corresponds to the clustered precinct ID where the ballot can only be used.
2	Fiduciary marks / accuracy marks	The accuracy marginal bars are used by the machine to identify the voter's mark.
3	UV Mark	All ballots used in the 2016 NLE contain an invisible UV mark. The VCM measures the UV light and accepts the ballot if it is within the configured spectrum.
4	Coat of arms of the Republic of the Philippines	Displaying the coat of arms of the Philippines in official ballots is mandated by law
5	Chairman signature slot	The BEI Chairman affixes his signature in this box on each ballot issued to a voter.
6	Barcode	Contains the serial number of the ballot. The VCM uses this to identify if the ballot was already scanned and subsequently rejects it.

During the election, voters are asked to shade their choice of candidate. The headers of actual ballots are slightly different, but the general elements shown in Figures 4 and 5 are present. The front of the ballots shows the choices for the national electoral contests: president, vice-president, senators, and party-list representative (sectoral representatives to Congress). At the back are choices for local electoral contests: provincial and city board members/councilors, mayors, vice-mayors, governors, and legislative district representatives (congress representatives). The size of the ballot varies based on the number of local candidates per clustered precinct. The ballot sizes for the 2016 NLE ranged from 25" to 31".

Figure 4 shows the ballot ID to be 05070003. This means that the ballot was for VCM / clustered precinct ID 05070003. The first four digits of the ID, "0507", refer to CCS ID. All VCMs with ID 0507XXXX transmit their ER at the closing of election to CCS with ID 0507, which, during the 2016 NLE, referred to the town of "Libon" in the province of "Albay".

The fiduciary marks or accuracy marks are used by the VCM to identify the choice of the voter. The VCM scans the image, and the image is processed by locating the marks. The intersection of the horizontal and vertical fiduciary marks at the margin of the ballot identifies the candidate voted for. For example, a vote for Duterte, as shown in Figure 7, will appear as {11,7} or 11th horizontal mark and 7th vertical mark. Count starts at zero (0).

The VCM recognizes the allowed maximum number of marks in an election contest. For example, the VCM invalidates all votes for a contest if multiple marks are seen for a contest that only requires one mark (aka over-vote).

Printing of ballots is supervised by a committee of five appointed by COMELEC. The committee is composed of (1) two members from COMELEC, (2) one member from the Commission on Audit (COA), and (3) a member each from the dominant political party and from the dominant opposition party. Other political parties and coalition parties may assign a watcher to witness the procurement and watermarking of papers that will be used for printing the ballots.

Ballot security changes over time but the basics required by law involve watermarks, the logo of the Republic of the Philippines, and the use of special paper [37]. Ballots can only be printed either by the National Printing Office (NPO) or the *Bangko Sentral ng Pilipinas*[4] (BSP). The ballot for each clustered precinct is unique. A ballot cast at the wrong precinct will be returned by the VCM.

The number of ballots printed is strictly controlled by law. Only 120% of the total registered voters in the Computerized Voters List (CVL) are printed. Sample ballots for purposes of voter education are printed with a different configuration / layout and require a different set of SD cards for the VCM. These are also watermarked to identify them as a "test" or "sample ballot". Voters are not given an extra ballot during election, except if the BEI chairman erroneously gives a ballot intended for another precinct. Voters cannot rectify errors they made on the ballot.

2.2.5. Transparency Server, Central Server, and the Transmission Flow

The automated elections has two other servers: the Transparency Server and the COMELEC Central Server. The Transparency Server is where all accredited political parties, mass media, and the citizen's arm connect to get data for their

[4] In english meant "The Central Bank of the Philippines"

quick counts. The COMELEC Central Server is the data source for COMELEC's own quick count. The election results from both servers are considered unofficial.

The official transmission of ER and canvassing results follows the BOC hierarchy shown in Table 1, but differs slightly with the introduction of the two servers. First, the VCM transmits to three destinations: (1) MBOC CCS; (2) the Transparency Server; and, (3) the COMELEC Central Server. The MBOC CCS transmits the canvassing results both to the COMELEC Central Server and to the provincial CCS. The process is repeated for the next level of the canvassing hierarchy. Thus, all VCM and CCS transmit to the COMELEC Central Server, while only the VCM transmits to the Transparency Server. Figure 6 shows a simplified illustration of the transmission flow.

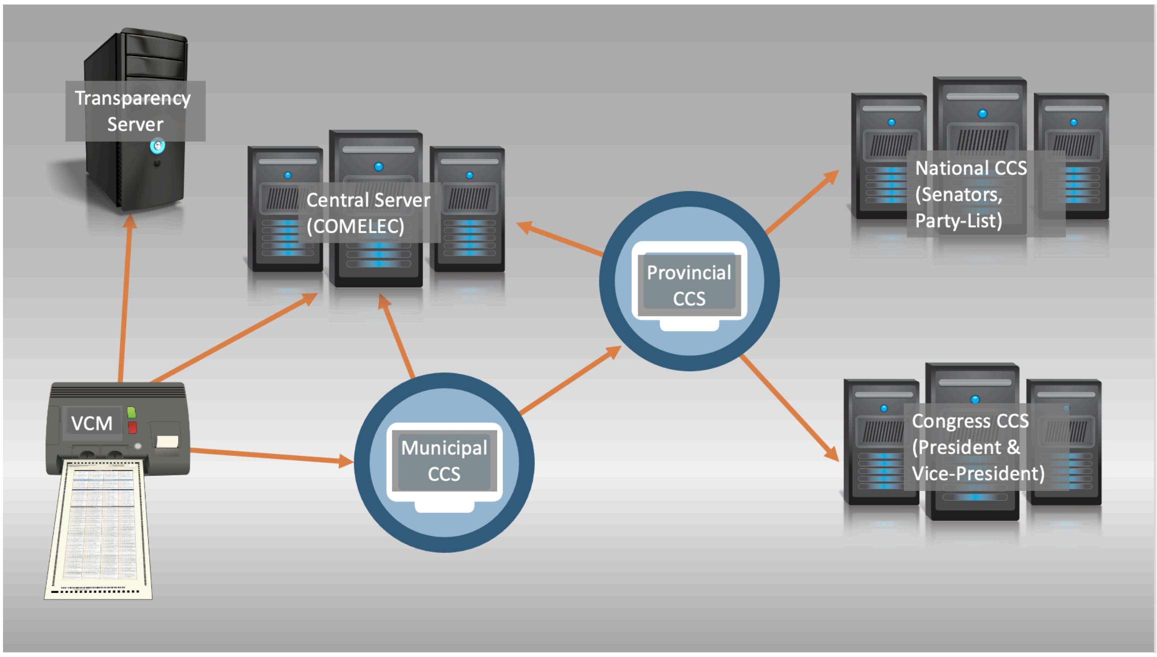

Transparency Server
Central Server (COMELEC)
National CCS (Senators, Party-List)
Provincial CCS
VCM
Municipal CCS
Congress CCS (President & Vice-President)

Figure 6. Transmission Flow

2.3. Overview of Procedures for Voting, Canvassing and RMA

NLE in the Philippines is fully automated since 2010, except for the authentication of voters, which remains manual. This sub-chapter provides an overview of the procedures of the various stages of the automated Philippine elections. Understanding the procedures is important in the analysis of the information security of the 2016 NLE.

2.3.1. The Computerized Voters List (CVL)

Non-registered voters can register anytime at COMELEC offices prior to an election. For ease of administration, any voter who failed to vote in the last two elections is deactivated from the Computerized Voters List (CVL). These voters can request for reactivation anytime.

Election registration boards at various administrative levels are tasked to approve the registration of new voters, effect any changes in the particulars of registered voters, and keep and maintain the CVL. At the national level, COMELEC is mandated to keep the CVL both in print and digital formats. A computer program with the capability to sort the CVL is required by law [4 Sec. 24].

Fifteen days before the start of the campaign period, the election registration boards notify all registered political parties and members of the BEI for each polling precinct to start the verification of the precinct-level CVL (PCVL). The PCVL becomes the *Election Day Computerized Voters List (EDVCL)* once verified. The EDCVL is delivered to the BEI on election day, and copies of the EDCVL are posted at the door of every polling precinct. Voters can check their assigned polling precinct in the CVL available online or at the EDCVL at the polling precincts.

COMELEC started the "*no bio, no boto*" campaign in 2016 [38] [39]. The 2016 AES however, had no automated authentication of voters. While the CVL was computer-aided, the authentication of whether a person is a registered voter or not remains archaic. Weaknesses in the voter verification system of COMELEC were highlighted in the election protest filed by former senator Ferdinand "Bongbong" Marcos, Jr. for his loss to Vice-President Maria Leonor "Lenny" Robredo in 2016. He cited COMELEC technical examinations that found 40,528 signatures and 3,295 thumb marks on the EDCVLs in various precincts in ARMM as not identical to those in the Voter Registration Records (VRR) in official COMELEC databases [40].

2.3.2. Pre-Election Final Testing and Sealing (FTS)

The FTS for the 2016 NLE was scheduled on May 2 to 6 [34]. The objectives of the FTS are: (1) to ensure that all election equipment and paraphernalia arrived completely at the polling precinct, and (2) to test if the VCM and its peripherals are working.

To test the VCM, the VCM goes through all election states (Figure 2) except the transmission of ER nor the printing of the second set of 22 ERs. The BEI also receives their iButtons and pin codes for the first time during the FTS. In 2016, the pin codes were pre-generated by COMELEC and were not chosen by the BEI. There was also no BEI nor BOC registration. Thus, the keys in the iButtons and USB tokens were not strictly associated to the people they were assigned to.

To test the VCM, a mock election is performed. Ten FTS participants are chosen among the poll watchers present. The participants vote. After the election is closed, the BEI poll clerk manually tallies the FTS voting results using a "manual ER sheet" and compares these with the results in the VCM-printed ER. If they match, the manual ER sheet is signed by the BEI and poll watchers. The election is then re-zeroed, as described in Chapter 2.2.2. This triggers the VCM to produce an initialization report that shows all candidates have zero votes. This report is circulated among the poll watchers. The manual tally sheets, FTS printed ER and the initialization report are placed back and kept in the VCM box.

The FTS concludes with the VCM and all paraphernalia carefully placed back in the box. The box is sealed with a green seal to indicate a successful FTS. Otherwise, the box is sealed with a red seal which meant that the VCM should not be used on election day.

2.3.3 Election Day Protocols

In the 2016 NLE, the BEI met at 5:00 am on May 9, 2016 [29] to await the official delivery of the ballots, EDCVL, and the FTS-passed VCM. Ballot covers are removed / opened in front of poll watchers. The VCMs are at "open voting" election state by 6:00 am.

A voter searches for his/her name on the EDCVL. Once he/she finds his/her name, any member of the BEI is informed of his/her intention to vote. The BEI poll clerk examines the voter's fingernails for presence of indelible ink, which would mean that he/she had already voted. The poll clerk then verifies the identity of the voter by requesting for a valid government ID. If the voter has no government ID, the poll clerk instructs the voter to take an oath, which is recorded in the Minutes of Voting (MOV) and authenticated by the voter through his/her signature and right thumb mark. Whether or not the voter is able to present a valid ID, the poll clerk is required to clearly and loudly state the name of the voter and allow anyone present, to object. If there are no objections, the voter is instructed to get the official ballot from the BEI chairman. The BEI chairman retrieves a ballot, and demonstrates to the voter that the ballot is free of any unauthorized marks and is not torn nor smudged. The chairman authenticates the ballot by signing on the appropriate box (refer to Figure 4). The voter is subsequently instructed to sign the EDCVL and affix his/her thumb mark as proof of receipt of a ballot. The ballot is then placed in a folder and given to the voter.

The voter proceeds to the voting booth to mark the ballot. Once done, the voter proceeds to scan the ballot in the VCM. If the ballot is rejected (refer to Appendix 1), the chairman inspects the ballot. If the ballot appears to be erroneous[5], another ballot is provided to the voter to fill out. Otherwise, the voter is instructed to try to have the ballot scanned again at different orientations four more times.

The voter is not given another ballot for mistakes s/he made when marking the ballot. If a voter refuses to cast the ballot, or is unsuccessful in scanning the ballot, the voter documents and certifies the incident to the MOV by affixing his/her signature and thumb mark. The used ballot is then returned to the BEI chairman who tears it in half to prevent it from being cast.

When the ballot is accepted, the VCM prints the VVPAT. If the voter believes the VVPAT to be incorrect, the voter logs the incident to the MOV, certifies it, and attaches the VVPAT to the MOV [29]. Otherwise, the VVPAT is deposited to a box in front of the BEIs. Finally, indelible ink is poured on the voters right index finger to indicate that s/he has voted. This is done regardless of whether the ballot was accepted or rejected.

At the end of the election day, the BEI changes the VCM state to "close voting" (refer to Figure 2). The BEI instructs the VCM to lock the WORM card in Slot B. After successfully locking the SD card in Slot B, the VCM shuts down. The BEI then removes the main SD card in Slot A and places the card in an officially sealed envelope affixed with the signatures of the BEI and the poll watchers. This envelope containing the VCM main SD card is sent to the municipal / city BOC.

The BEI proceeds to box the VCM. If the VCM or SD cards did not malfunction, the VCM Slot B (refer to Appendix 1) should remain sealed with a red plastic tie

[5] The ballot could be incorrectly assigned to a different precinct as indicated by the ballot precinct ID (refer to Figure 4), or the ballot may contain printing machine errors.

containing the COMELEC logo. The BEI can be held liable if the Slot B seal is broken without following protocol. A copy of the ER, the MOV, and VCM reports (e.g. initialization, diagnostic, statistics, etc.) are placed in another envelope and placed inside the ballot box. The ballot box is sealed with a yellow sticker indicating that the VCM is in a "close voting" election state and therefore contains official election results.

2.3.4. Canvassing of Votes Procedures

In 2016, the MBOC met at 3 pm on election day. Some higher-tier BOC (e.g. provincial, regional, national) may meet later to receive transmissions of canvassing results from lower-level BOC (refer to Table 1).

The procedures to operate the CCS are outlined in [31]. The BOC opens the laptop and encodes the laptop username and password. The CCS application login page where he/she types the application username and password. The CCS then prompts that the election will be activated, during which, two members of the BOC insert their USB security tokens and input the token's 6-digit key. The CCS initializes and an initialization report is printed showing that the system has not accepted any ER (for Municipal CCS) or canvassing results (for provincial or higher level CCS) yet. The CCS is now in an "active election" state.

The CCS contains different modules: (1) for printing COC, SOV, and COCP; (2) for results monitoring; (3) for manual uploading of results; and, (4) for other functions. The BOC is expected to continually monitor the transmission of results until all VCMs or canvassing results have been received.

At the municipal level, the main SD card in the VCM Slot A is delivered from every precinct. Whether or not the VCM successfully transmitted their ER, the MBOC uploads the ER from the received SD card to the CCS. The CCS has an anti-replay mechanism and knows if the uploaded ER was received earlier. In this case, the ER is rejected as a duplicate ER. After all SD cards are received and the ER uploaded, the MBOC CCS generates and prints the COC, SOV, and the COCP. Canvassing is considered finished when the three official canvassing documents are printed. The MBOC then attempts to transmit the canvassing results to the next-level CCS (provincial / PBOC). The canvassing result from the municipal / city CCS will be the basis of the provincial CCS for generating the COC, SOV, and COCP at that level.

The MBOC saves the canvassing result files to a CD-ROM and places these in an envelope. If the transmission is successful, the envelope is labeled "transmission successful". They next perform a back-up of the audit logs of the CCS. The MBOC finally prints a post-election report. The MBOC members affix their signatures on the post-election reports.

The next level of canvassing is at the provincial BOC or PBOC. Similar to the MBOC, the PBOC also monitors the transmission of the canvassing results from the municipal CCS, and receives the CD-ROM containing the canvassing results from every municipality or city in their jurisdiction. Once received, the PBOC examines the markings on the envelope. If the mark said "transmission unsuccessful", the PBOC uploads the results manually to the provincial CCS. After receipt of all canvassing results, the PBOC proceeds in a similar fashion as the MBOC. They print the provincial COC, SOV, and COCP. They then attempt to transmit the canvassing results. The canvassing result files are saved to the provided CD-ROM and sent to the next level in the BOC hierarchy. A back-up copy of the canvassing result is also saved. This process is repeated at every level of the BOC hierarchy enumerated in Table 1.

2.3.5. Procedures for Random Manual Audit (RMA)

A Random Manual Audit (RMA) is done by randomly selecting a clustered precinct per legislative district [8 Sec. 24]. The random selection of precincts for RMA is performed by the national RMA Committee which, in 2016, was composed of the Deputy National Statistician of the Philippine Statistics Authority (PSA), two Certified Public Accountants (CPA) from the citizens' arm, NAMFREL, and a lawyer from COMELEC [41]. The source code of the random precinct selector software used was reviewed by registered political parties and citizen's arm in 2016 [42]. Seven hundred twelve (712) clustered precincts were randomly selected, but 28 were not audited because the ballot boxes were not retrieved [18]. Thus, only 684 comprising 0.74% of the total 92,509 clustered precincts were audited.

An RMA Committee is formed for every legislative district. Members of this committee possess the same qualifications as the BEI [43]. The district committees perform the RMA at 7:00 am of May 10, 2016 which is basically just publicly conducting a manual tally of the ballots and comparing these with the official ER [44]. Recall that at the end of the election, the ER printout is also in the ballot box along with all the ballots (Chapter 2.3.3). The RMA Committee does not examine anything else. They do not boot the VCM nor do they review audit logs. The sole purpose of the RMA is to determine the accuracy of the VCM to read ballots. RMA results do not affect the outcome of the election even if discrepancies between the manual tally and the official ER are found. The 2016 RMA reported that the VCM used were 99.9023% accurate in reading ballots.

2.4. Certification of the Automated Election System

The Automated Election Law of 2007 enumerates information security requirements for the AES. Requirements for certification are also mandated. Although COMELEC may use an uncertified AES, they should justify such decision to Congress.

To review the implementation of the law, the JCOC is formed. It is composed of 14 members with seven members each from the Senate and the House of Representatives[6]. They perform a comprehensive assessment of the performance of the AES used in every election.

Although not part of the certification process, a significant feature of the law is the requirement for a a public source code review [8 Sec. 12], during which all political parties, accredited citizen's arm and IT groups may examine and test the hardware and software of the different components of the AES.

2.4.1. Information Security Requirements of the AES

According to [8 Sec. 6], the AES minimum capabilities are as follows:

"Sec. 6 Minimum System Capabilities - 'The automated election system must have at least the following functional capabilities:

(a) Adequate security against unauthorized access;

(b) Accuracy in recording and reading of votes as well as in the tabulation, consolidation/ canvassing, electronic transmission, and storage of results;

(c) Error recovery in case of non-catastrophic failure of device;

(d) System integrity which ensures physical stability and functioning of the vote recording ad counting process;

[6] The Philippine legislature is bicameral composed of the Senate (aka the upper Chamber), and the House of Representatives (aka lower chamber).

(e) Provision for voter verified paper audit trail;

(f) System audibility which provides supporting documentation for verifying the correctness of reported election results;

(g) An election management system for preparing ballots and programs for use in the casting and counting of votes and to consolidate, report and display results in the shortest time possible;

(h) Accessibility to illiterates and disabled voters;

(i) Vote tabulating program for election, referendum or plebiscite;

(j) Accurate ballot counters;

(k) Data retention provision;

(l) Provide for the safekeeping, storing and archiving of physical or paper resource used in the election process;

(m) Utilize or generate ballots herein defined;

(n) Provide a voter system of verification to find out whether or not the machine has registered his choice; and,

(o) Configure access control for sensitive system data and functions. "

The law also recognized the need to secure electronic communication. Sec. 7 of the law read:

> "Sec. 7 Communication Channels for Electronic Transmissions. - All electronic transmissions by and among the AES and its related components shall utilize secure communication channels as recommended by the Advisory Council, to ensure authentication and integrity of transmissions."
> Since the information security requirements in section 6 were vague, the law tasked COMELEC through the aid of a COMELEC Advisory Council (CAC) to create and implement an "evaluation system" containing more detailed regulations and/or standards for the AES. More than a decade since the law was passed, there is still no published standard evaluation criteria, nor any Implementing Rules and Regulations (IRR) that detailed these information security requirements."

In addition to the provision on secure communication channels, the law explicitly states that election results should be "digitally signed" by the BEI and the BOC and that the authentication of transmitted results follows the Philippine Electronic Commerce Act of 2000. This law defines electronic signatures as equivalent to handwritten signatures only if there are sufficient protocols present to ensure that the digital signature came from the person for which the signature was publicly known to be associated with [45]. This security service is referred to as non-repudiation.

2.4.2. The COMELEC Advisory Council (CAC) and the TEC

A COMELEC Advisory Council (CAC) is formed not later than 18 months prior to any automated election, and they remain active until six months after the completion of the canvassing of votes [8 Sec. 8]. The members of the CAC are as follows:

1. Chairman of the Commission on Information and Communications Technology[7] as chairperson;
2. One member from the Department of Science and Technology (DOST);
3. One from the Department of Education (DepEd);
4. One member from the academe chosen among the nominees by the country's academic institutions;
5. Three members nominated by the country's ICT professional organizations; and,
6. Two members from NGOs championing electoral reforms.

The CAC recommends the appropriate technology that is "secure, practicable, and cost-effective" for use in the election [8]. They serve as the ICT expert panel who evaluates and reports the risks and technical issues and provides advice on inadequacies that can surface in the course of procurement, storage and disposition of various components of the AES. The CAC submits a written assessment of the performance of the AES within six months after the conduct of an automated election to the JCOC.

2.4.3. The TEC and the AES Certification Process

[7] CICT is now the Department of Information and Communications Technology (DICT) headed by a Cabinet Secretary.

A Technical Evaluation Committee (TEC) is created with representatives each from COMELEC, DICT, and DOST. The DOST representative is the chairperson of the TEC.

The most important function of the TEC is to certify that the *"AES is secure, accurate, and operating as per specification" [8 Sec. 11]*. The TEC can request an established ICE to assist in the certification.

The TEC certification should be completed not later than three months before the elections. The certification's minimum requirements are:

1. Successful conduct of field tests and mock elections;
2. A comprehensive audit on the accuracy, functionality and security of the AES;
3. A source code review, usually done by the ICE;
4. Certification that the reviewed source code of all the components of the AES is securely deposited with the BSP;
5. An assurance that the compiled code in escrow with the BSP and those deployed in the field are one and the same; and,
6. An approved continuity and risk management plan for the AES.

For item 5, the TEC completes a ceremony of compiling the code and witnessing the completion of the AES "trusted build". The hash of the trusted build is then collected and also deposited with the BSP.

For the 2016 NLE, the ICE was SLI Gobal Solutions based in Denver, Colorado, USA. The TEC certified the EMS on January 26, 2016, and the CCS and VCM on February 10, 2016. The source codes were received by BSP and escrowed on February 12, 2016. The system's extracted SHA 256 hash codes were published in COMELEC websites. The ICE's final report on the three major components was submitted to COMELEC on April 14, 2016.

3. Methodology

The objective of this research is to assess the information security of the 2016 Philippine automated election. To perform an effective assessment, the appropriate automated election "trust" model was first developed. The model included certain properties considered to be universal for an AES. This model serves as the reference for the evaluation of the 2016 NLE.

The second objective is to recreate what transpired during the 2016 elections. Few academic literature on the event exists, and none were found to focus on the technical aspects of the Philippine AES. To objectively recreate election events, computer audit logs submitted to JCOC in 2018 were examined[8].

3.1. The Automated Election System Trust Model

The AES information security requirement, often referred to as a "trust model", is an expanded version of the standard information security properties of Confidentiality, Integrity and Availability (CIA). Various models were considered in defining the automated election trust model appropriate for the Philippines. The research borrowed heavily from the e-Voting Trust Model proposed by [46]. The model was improved by incorporating elements from similar works particularly by [47] [48] and [49]. Common properties that should be present in any AES were found. Though not usually present in other models, "non-repudiation" (discussed in Chapter 2.4.2) was added in the Philippine AES Trust Model because it is required by law. Therefore, this research developed a Philippine AES Trust Model that contains the following properties:

1. *Privacy of Voters* — this property appeared in various literature using different semantics such as "secrecy of the voter / ballot" or "unlinkability". In this research, privacy refers to both the anonymity of the voter and unlinkability of the voter to the records of votes cast.
2. *Uncoercibility / Receipt-Freeness*: The properties of "uncoercibility" and "receipt-freeness" were expounded by [48] as a requirement for any AES. To maintain a coercion-free environment, the voter must not be given the opportunity to provide any legitimate proof of how s/he voted to another person. While similar to the first property, uncoercibility is different because it removes the opportunity for voters to find their record of vote in plaintext, which they can voluntarily present to another party as proof of how they voted.
3. *Individual verifiability* — pertains to the capability of the system to prove to a voter that his/her vote was processed. This includes providing feedback to the voter if the votes cast are as s/he intended and if the votes are counted accurately. Some votes may not be counted due to errors by the voter. In these

[8] The logs were used with the permission of the Senate President and the Senate Committee on Electoral Reforms. The audit logs were already discussed in a session of Senate and thus are now public records.

cases, the voter must be informed and be given the opportunity to react appropriately.

4. Universal verifiability — This property is one of the most documented AES properties. Universal verifiability is mostly about transparency. The AES' design, implementation, and usage must be well documented, reproducible, and auditable from voting, to counting, canvassing until determination of winners [47]. This property may be challenging to achieve in the present canvassing procedure discussed in Chapter 2.2.5, where votes are stored, sent, and processed multiple times in different data formats at different stages.

5. Fairness — in election literature, fairness often refers to the absence of the capability of anyone, including election authorities, to gain knowledge of who is winning before the the election is officially closed. This is to avoid giving candidates the opportunity to plan legal or extra-legal contingencies which include derailing proclamation of winners if they knew they are losing.

6. Integrity — As with the classical definition of integrity in information security, this property is ambiguous. Integrity is the assurance that the election was not manipulated. This includes the AES' capability to detect illegitimate voters and ballots. This also includes error-detection, error-recovery, authentication and identification of election authorities and voters. While verifiability pertains to votes and voters, integrity is an assurance that the entire system is tamper-proof. This implies that only the certified software and hardware are used in the election[9], and the encryption keys used were kept confidential.

7. Availability — The system must be resilient against attempts to sabotage it.

8. Non-repudiation — This property is a requirement of the Philippine Automated Election Law of 2007. The requirement is a stronger version of entity authentication. Non-repudiation is an assurance that any transactions made, such as, but not limited to, the generation of election documents (e.g., ER, COC, SOV, and COCP) and the transmission of election results, are undeniably performed by the person recorded to have processed them.

3.2. Extracting Data from the Audit Logs

[9] In the 2016 NLE, this is called the Trusted Build.

This research obtained some audit logs of the components of the AES. The VCM and CCS logs for the municipalities of Libon, Albay and Tugaya, Lanao Del Sur were provided by COMELEC to JCOC. VCM logs for the municipality of Angono, Rizal were also provided, but no CCS logs. Table 3 lists the VCM and CCS election logs provided to JCOC. The logs were compared with the expected number of VCMs deployed per municipality according to the POP.

Table 3. Number of Log Files submitted to JCOC

Municipality / Precinct	***.pdf" VCM logs**	**"*.log" VCM logs**	**# of clustered precincts per POP**	**Number of CCS Log Files**
Libon, Albay	71	74	74	20
Angono, Rizal	91	93	93	0
Tugaya, Lanao Del Sur	25	25	25	27
Total	187	192	192	47

The log files came in two file formats. One set of 187 audit logs was in portable data format or PDF. The other set of 192 logs was in plaintext format with the filename extension ".log".

The plaintext format of the logs of the Transparency Server, Central Server, and the Domain Name Service (DNS) Servers was also provided. Due to the size and number analyzed, three python3 programs were created for data extraction. The programs were:

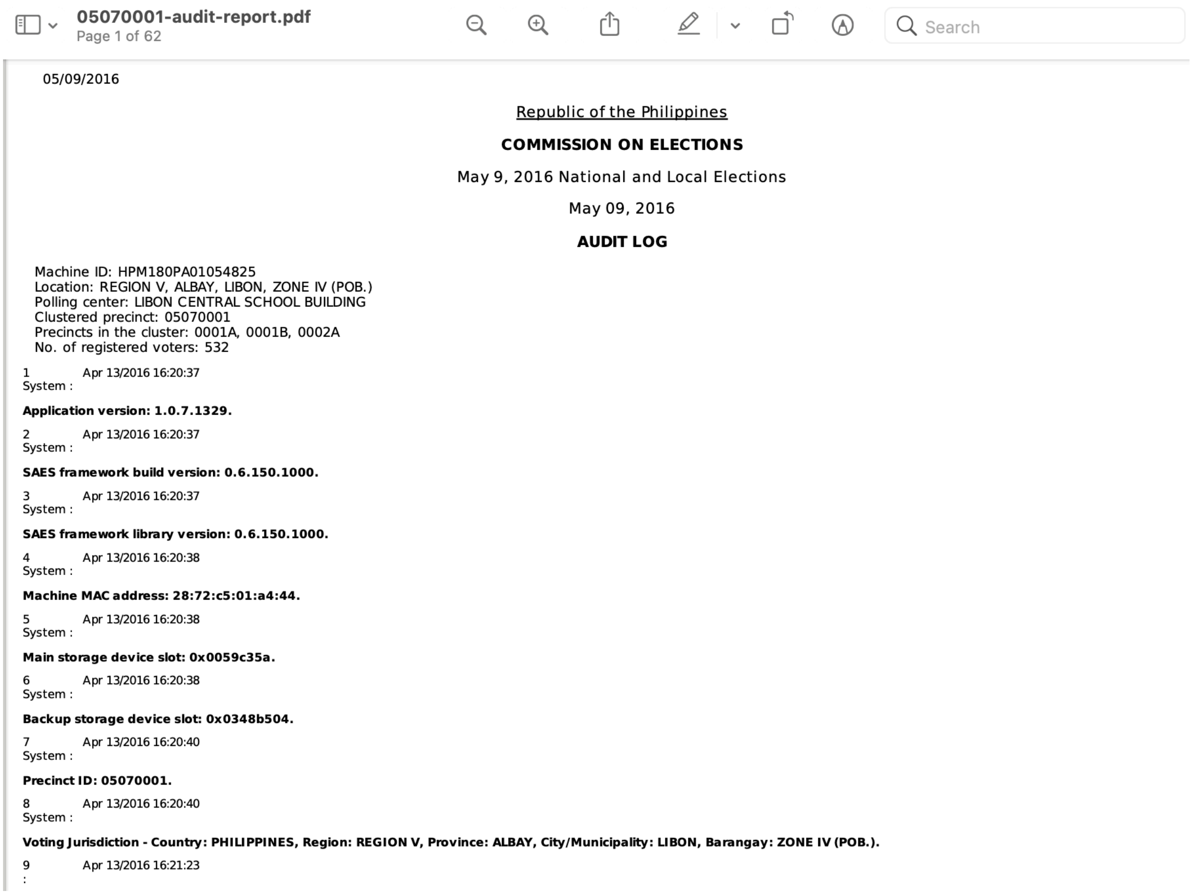
(a) *ElectionPdfLogParser.py* — This program converts the PDF audit logs to plaintext format and reformats the file line-per-line to assure a more consistent file structure. The output has a filename extension of ".parsed"

(b) *ElectionPdfLogSummarizer.py* — The program reads and extracts data from the ".parsed" files.

(c) *ElectionTxtLogSummarizer.py* — This is similar to (b), except that it collects data and performs initial analysis of the plaintext audit log files.

3.2.1. Python 3 Code Program Flow Logic

A screenshot of the PDF file audit log submitted by COMELEC is shown in Figure 7. The first python3 program, "ElectionPdfLogParser.py", converted each PDF log to text files with filename extension of ".parsed". The program contained multiple parsing logic to ensure the output was consistently formatted. A copy of the python3 code for “ElectionPdfLogParser.py” is in Appendix 2.



05/09/2016

Republic of the Philippines

COMMISSION ON ELECTIONS

May 9, 2016 National and Local Elections

May 09, 2016

AUDIT LOG

```
  Machine ID: HPM180PA01054825
  Location: REGION V, ALBAY, LIBON, ZONE IV (POB.)
  Polling center: LIBON CENTRAL SCHOOL BUILDING
  Clustered precinct: 05070001
  Precincts in the cluster: 0001A, 0001B, 0002A
  No. of registered voters: 532

1           Apr 13/2016 16:20:37
System :

Application version: 1.0.7.1329.

2           Apr 13/2016 16:20:37
System :

SAES framework build version: 0.6.150.1000.

3           Apr 13/2016 16:20:37
System :

SAES framework library version: 0.6.150.1000.

4           Apr 13/2016 16:20:38
System :

Machine MAC address: 28:72:c5:01:a4:44.

5           Apr 13/2016 16:20:38
System :

Main storage device slot: 0x0059c35a.

6           Apr 13/2016 16:20:38
System :

Backup storage device slot: 0x0348b504.

7           Apr 13/2016 16:20:40
System :

Precinct ID: 05070001.

8           Apr 13/2016 16:20:40
System :

Voting Jurisdiction - Country: PHILIPPINES, Region: REGION V, Province: ALBAY, City/Municipality: LIBON, Barangay: ZONE IV (POB.).

9           Apr 13/2016 16:21:23
:
```

Figure 7. Screenshot of the PDF file audit log file of the VCM.

The second python3 program, “ElectionPdfLogSummarizer.py”, is similar to the third program “ElectionTxtLogSummarizer.py”, except that the former processed 182 “.parsed” files, while the latter processed 192 plaintext files with file extension “*.log”. Minor differences in the syntax and the log formats for the two types of audit logs were observed, but the program flow logic of both codes was the same. Hence for brevity, only the code for “ElectionTxtLogSummarizer.py” is attached in this research in Appendix 3.

These programs used the os.walk() function to read each audit log file and processed the files as illustrated in the program flow diagram in Figure 8. The outputs were "report files" containing important findings. One report file was produced per audit log file. Thus, a total of 379 report files was produced for this research.

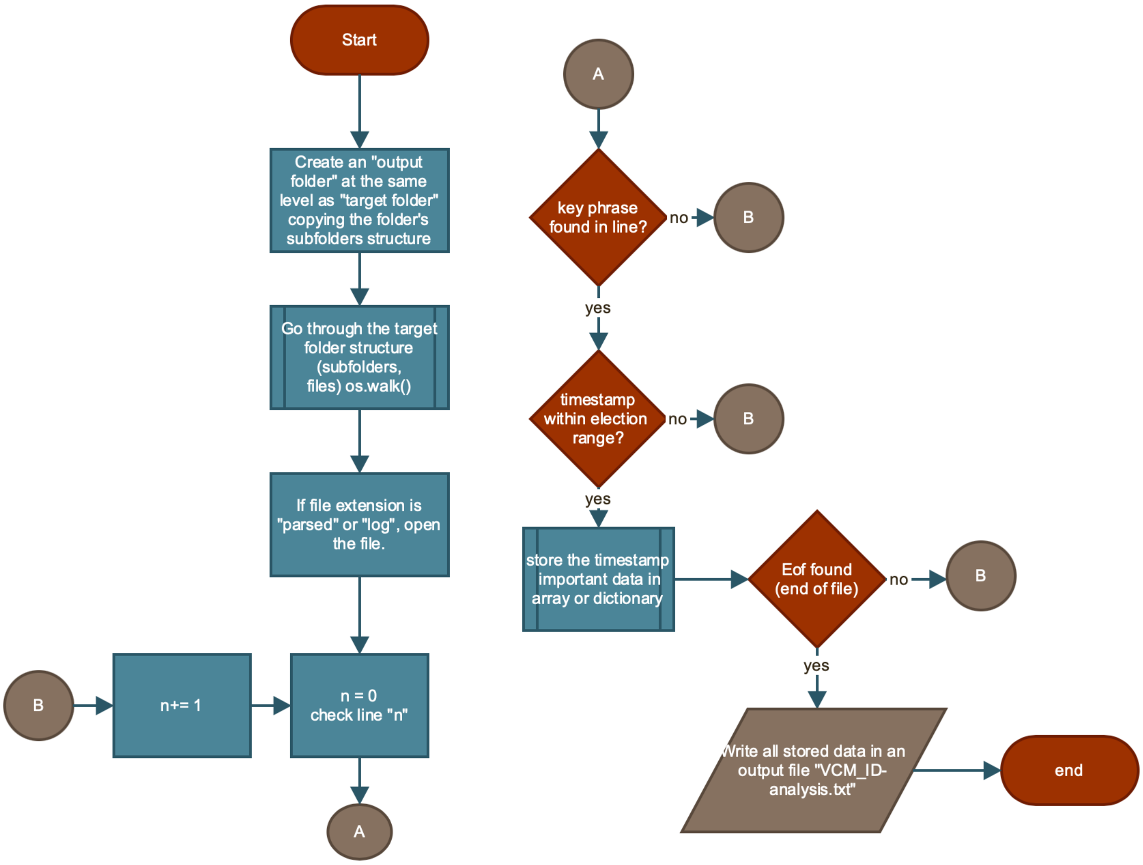

Start
Create an "output folder" at the same level as "target folder" copying the folder's subfolders structure
Go through the target folder structure (subfolders, files) os.walk()
If file extension is "parsed" or "log", open the file.
B
n+= 1
n = 0 check line "n"
A
A
key phrase found in line?
no
B
yes
timestamp within election range?
no
B
yes
store the timestamp important data in array or dictionary
Eof found (end of file)
no
B
yes
Write all stored data in an output file "VCM_ID-analysis.txt"
end

Figure 8. Program Flow Logic for analyzing the VCM audit log files

The program searched for key phrases at each line of the file. Below is a code snippet from Appendix 3, showing how the per line key phrase search, and data collection, worked:

```
_findPhrase = [['Vote cast completed'],
               ['Ballot returned'],
               ['Cleaning process started.'],
               ['A ballot was inserted']
          ]

with open(_sourceFile) as _source:
     for _line in _source:
          for _phrase in _findPhrase[3]:
```

```
if _phrase in _line:
    _array = _line.split('.')
    _insertTime = harvestTimeLog(_array)
    if _lastBallotAccepted < _insertTime:
        _ballotStats[3] += 1
        _timeStamps[0].append(_insertTime)
```

Because the research aimed to recreate the events of the election, from the conduct of the FTS to the canvassing of votes, the program checked the timestamps to determine whether the event logged found should be skipped. The code snippet below from Appendix 3 shows the time references.

```
  #as per Comelec Res 10088 amending 10057, the election day starts 6AM.
_electionTime = datetime.datetime.strptime('May 9/2016 06:00:00', "%b %d/%Y %H:%M:%S" )

 #time we do not expect any transmissions. If there are any, it can't be from PRELAT
_lockDownTime = datetime.datetime.strptime('May 1/2016 00:00:00', "%b %d/%Y %H:%M:%S")
```

After all audit log files were processed, the events gathered were written in one summary file. The summary file for all processed ".log" audit log files is in Appendix 4. The summary file contains the file name of the report file where the events were initially found. For example, the summary file reported four VCMs were rezero-ed during election day (Appendix 4):

```
Total Rezeros that affected actual votes: 4, recorded in 4 log files.
   * The machine was rezeroed 1 times affecting actual ballots fed for 36290005-audit
   * The machine was rezeroed 1 times affecting actual ballots fed for 36290016-audit
   * The machine was rezeroed 1 times affecting actual ballots fed for 36290014-audit
   * The machine was rezeroed 1 times affecting actual ballots fed for 36290019-audit
```

To study the rezero done on VCM ID 36290005, its report file "36290005-audit" was opened to reveal the following:

```
REPORT FILE GENERATED: Sat Jan 30 16:43:45 2021
     ############################################################
     #      GENERATED BY ElectionPdfLogSummarizer.py        #
     #                              #
     # this file contains the summary of important lines in the  #
     # VCM *.log audit logs. Analysis is made by the author     #
     # using relevant election CCS MBOC Logs.              #
     #                              #
     # Author: Jeffrey Ian Dy. 2020.                 #
     # for MsC Information Security. University of London      #
     ############################################################
```

```
SUMMARY OF FINDINGS for 36290005-AUDIT VCM LOG FILE

________________________________________________________________________________

Time stamp of last ballot counted is 2016-05-11 03:10:14
Total ballots casted is 329
Total ballots returned is 23
Total times election was (re-)opened is 3
    * 2016-05-09 01:54:35
    * 2016-05-09 09:55:50
    * 2016-05-10 21:54:33

Total shutdown during election : 4
    * 2016-05-09 20:45:50
    * 2016-05-10 13:55:08
    * 2016-05-10 14:58:24
    * 2016-05-10 21:46:15

Total Rezeros after voting started : 1
    * 2016-05-10 21:30:06

**************** Results Transmission Analysis *********************

VCM successfully transmitted on the following timestamps:
    * from IP 10.11.4.8 on 2016-05-11 03:29:03

[Alert!!!] The CCS did not process the transmission sent by this VCM.
```

To check the accuracy of the report files and summary files, a random audit was performed. Fifty report files were compared with the original audit logs. If a discrepancy in the data was found, the python 3 program was debugged and revised. The codes shown in Appendices 2 and 3 passed this audit.

3.2.2. Transmission Log Data Extraction Program Flow Logic

The code for extracting transmission of the VCM and the CCS was slightly more complex. This was handled by the transmissionInfo() function (Appendix 3). Unlike the other functions, transmissionInfo() read the VCM audit logs and compared the found transmission logs with the CCS logs to check if the latter processed the ER. The program flow logic of transmissionInfo() is as follows:

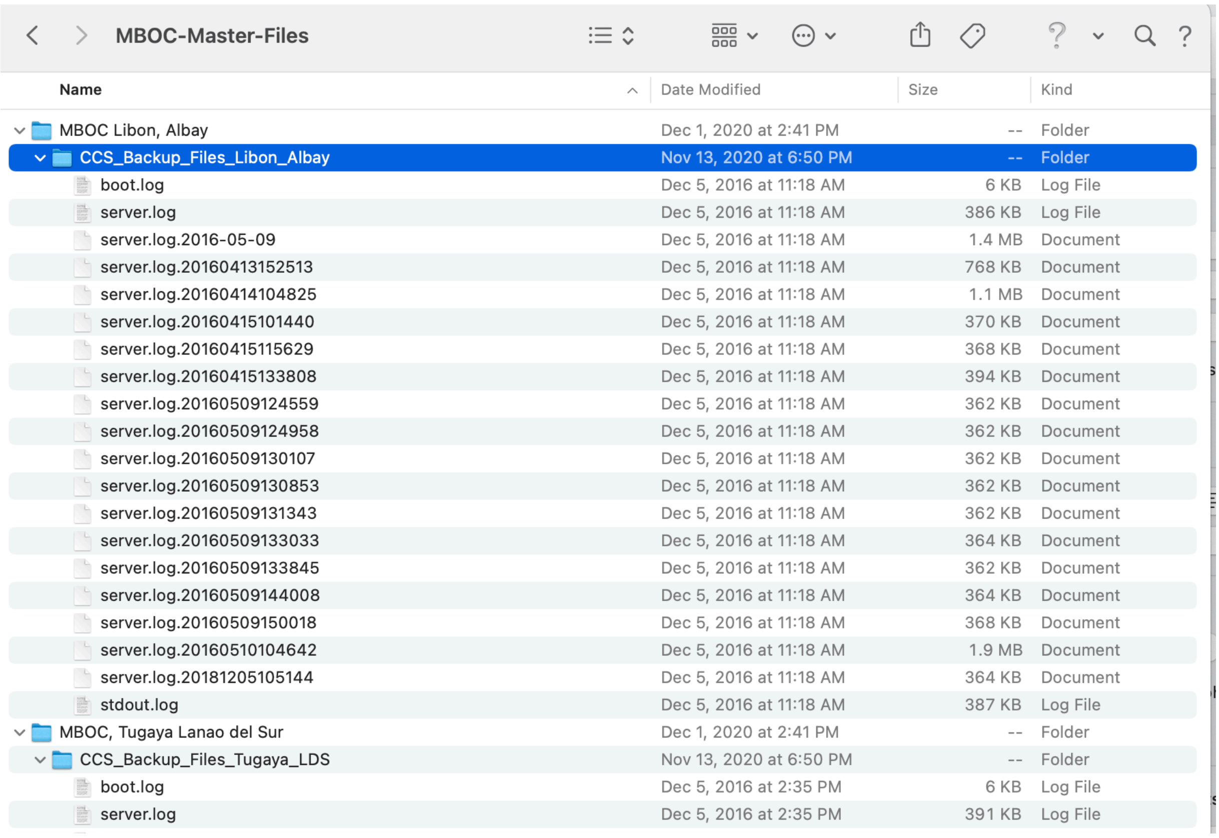
1. The VCM ID of the transmitting VCM was extracted from the VCM log filename The code snippet below shows how regex and the python3 “Pathlib.path().stem” function were used for this process.

```
_vcmIdregex = re.compile(r'(\d{4})(\d{4})')
_vcmId = _vcmIdregex.search(Path(_sourceFile).stem)
_transSearchPhrase = [["The information from results report [" + str(_vcmId.group()) + ".0"],
```

2. The function searched for the phrase, *"The election results were sent successfully to 'MBOC:"* at every line fo the VCM audit log.
3. The phrase in Step 2 indicated the successful transmission of the VCM. If found, the function searched for the phrase *"The information from results report ["* concatenated with the extracted VCM ID, in every line of each of the CCS logs.
4. If the phrase was found in any of the CCS logs, the timestamp between the transmission of the VCM and the time of processing of the MBOC CCS was compared. To be considered as a matching "transmission", the CCS log timestamp must be within +/- 3 hours of the timestamp of the VCM transmission. The process of identifying the timestamp of a line in the CCS logs is discussed in Chapter 3.2.3.
5. Five initial analyses were performed by transmissionInfo(). They were:
    - If the IP address of the sending VCM was different than what was reflected in the CCS;
    - if the transmission was an "early transmission" defined as a transmission made between May 1, 2016 at 12:00 AM to May 9, 2016 at 3:00PM[10];
    - If the CCS received the VCM transmission or not;
    - If the CCS processed a transmission, yet the VCM logs did not show any transmission; and,
    - If the time difference between the sending of the VCM and processing of the CCS was reasonable (within a +30 minute range).

The findings are mentioned both in the report files and in the summary file.

[10] Any transmission made before May 1 was considered a "test" transmission, and is therefore valid. Since the FTS was scheduled for May 2-6, 2016 (refer to Chapter 2.3.2), the VCM should have been boxed on May 1, and no transmissions were made until the close of the election. The BEI were barred by COMELEC to to transmit during FTS.



Name	Date Modified	Size	Kind
MBOC Libon, Albay	Dec 1, 2020 at 2:41 PM	--	Folder
CCS_Backup_Files_Libon_Albay	Nov 13, 2020 at 6:50 PM	--	Folder
boot.log	Dec 5, 2016 at 11:18 AM	6 KB	Log File
server.log	Dec 5, 2016 at 11:18 AM	386 KB	Log File
server.log.2016-05-09	Dec 5, 2016 at 11:18 AM	1.4 MB	Document
server.log.20160413152513	Dec 5, 2016 at 11:18 AM	768 KB	Document
server.log.20160414104825	Dec 5, 2016 at 11:18 AM	1.1 MB	Document
server.log.20160415101440	Dec 5, 2016 at 11:18 AM	370 KB	Document
server.log.20160415115629	Dec 5, 2016 at 11:18 AM	368 KB	Document
server.log.20160415133808	Dec 5, 2016 at 11:18 AM	394 KB	Document
server.log.20160509124559	Dec 5, 2016 at 11:18 AM	362 KB	Document
server.log.20160509124958	Dec 5, 2016 at 11:18 AM	362 KB	Document
server.log.20160509130107	Dec 5, 2016 at 11:18 AM	362 KB	Document
server.log.20160509130853	Dec 5, 2016 at 11:18 AM	362 KB	Document
server.log.20160509131343	Dec 5, 2016 at 11:18 AM	362 KB	Document
server.log.20160509133033	Dec 5, 2016 at 11:18 AM	364 KB	Document
server.log.20160509133845	Dec 5, 2016 at 11:18 AM	362 KB	Document
server.log.20160509144008	Dec 5, 2016 at 11:18 AM	364 KB	Document
server.log.20160509150018	Dec 5, 2016 at 11:18 AM	368 KB	Document
server.log.20160510104642	Dec 5, 2016 at 11:18 AM	1.9 MB	Document
server.log.20181205105144	Dec 5, 2016 at 11:18 AM	364 KB	Document
stdout.log	Dec 5, 2016 at 11:18 AM	387 KB	Log File
MBOC, Tugaya Lanao del Sur	Dec 1, 2020 at 2:41 PM	--	Folder
CCS_Backup_Files_Tugaya_LDS	Nov 13, 2020 at 6:50 PM	--	Folder
boot.log	Dec 5, 2016 at 2:35 PM	6 KB	Log File
server.log	Dec 5, 2016 at 2:35 PM	391 KB	Log File

Figure 9. Screenshot of MBOC CCS Log files submitted to JCOC

3.2.3. Methodology Assumptions and Constraints

There is no international standard defining what data should be included in a log file. Any vendor is free to create their own log file in a format that they see fit. This was the case with the SAES 1800 *plus*. The VCM log file-naming convention was <VCM_ID>-audit.log, while the PDF-formatted VCM log files were named <VCM_ID>-audit-report.pdf. The timestamp format was also different for different log types of the SAES 1800 *plus*:

- *PDF VCM logs* — The timestamp format was <MM><DD>/<YYYY> <HH>:<MM>:<SS> (shown in Figure 7).
- *Plaintext log files* — The timestamp format was 2016-05-09 11:18:25.819 or <YYYY>-<MM>-<DD> <HH>:<MM>:<SS>.<MS>.
- *CCS log files* — The date was not included in the timestamps. The timestamp format was 18:51:06,720 or <HH>:<MM>:<SS>,<MS>.

The filenames of the CCS logs appeared to contain the date of the last log entry. The year, month, and day of the log file were indicated in the filename either in the format <YYYY>-<MM>-<DD> or <YYYY><MM><DD> (Figure 9).

The function convertTimeMBOCLog() found in Appendix 3 used regex to extract the required string from both the log line and filename and converted the extracted string to a date variable.

3.2.4. Resolving Conflicts between the Different VCM Log File Types

Some differences were noted between the data found in the PDF and plaintext audit logs. The former had a hash of the VCM trusted build at the end of the audit log file, whereas the latter did not. The first thirteen lines of the PDF logs were general information such as the total registered voters for the clustered precinct, among others. The plaintext audit logs did not contain these. The plaintext audit logs were, at the average, 100 lines longer than their corresponding PDF logs because the plaintext logs were more raw and contained more "step-by-step' information than the PDF files. Overall, "incomplete" and inconsistent log entries were found in the PDF logs but not in the plaintext logs. Table 4 summarizes the differences between the two audit log formats.

Table 4. Incomplete Entries count: PDF Audit Logs vs Plaintext Audit Logs

	PDF audit logs	**Plaintext audit logs**
Total Logs submitted to JCOC	187	192
# of logs that didn't start in line '1'	17	0
# of logs with no entry (blank log file)	1	0
# of logs where the ballot count did not start at '1'	14	0

Table 3 in Chapter 3.2.1 also shows that the number of plaintext audit logs submitted were equal to the expected number of VCMs deployed in each municipality, as per the official POP.

Therefore, the ".log" plaintext audit logs were considered complete and more accurate. Whenever there were discrepancies between the two audit log file types, the plaintext files were considered with more weight than the PDF files. This is why the summary file of the PDF audit logs is not included in the appendices.

4. Analysis

As discussed in Chapters 1.1 and 3, the analysis of the logs are the more objective way to recreate the election events. The VCM and CCS audit logs submitted to JCOC covered 120,740 registered voters (Table 5) or only 0.22% of the 54.36 million registered voters [50]. The findings in the audit logs may not be a statistical representative of what really happened in the entire election. However, these were the only logs provided to the JCOC. The transcript of the meeting of the JCOC TWG even revealed that the logs were first seen and analyzed by members of the CAC present in the TWG [14]. Absent any other formal verification opportunities, the research proceeded to make conclusions based on the analysis of this set of logs provided.

Table 5. Total Registered Voters for the Submitted VCM and CCS Logs

Municipality	Registered Voters	MBOC Code (first 4 digits of VCM/Precint ID)
Libon, Albay	41,117	0507
Angono, Rizal	68,317	5801
Tugaya, Lanao Del Sur	11,306	3629
Total	120,740	

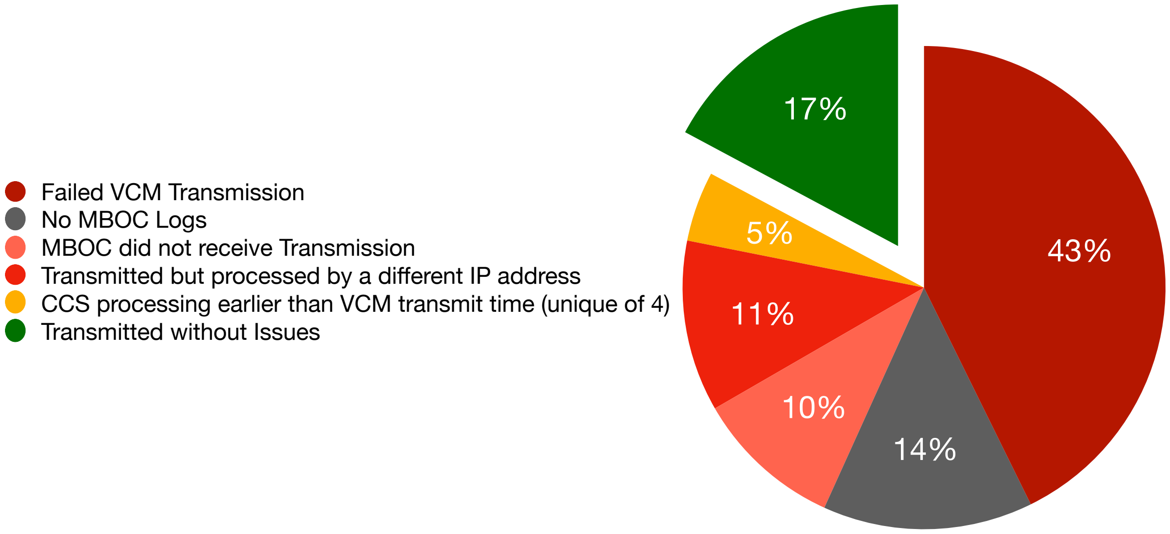


Figure 10. Transmission Statistics

Of the total 192 unique VCM audit logs, 82 (43%) logs indicated failed transmission of ER, while 110 (57%) indicated successful transmission. Among the successful transmissions, 27 (14%) logs belonged to the municipality of Angono, Rizal whose CCS logs were not submitted by COMELEC. The MBOC CCS of Tugaya, Lanao Del Sur did not receive the ER that the 19 VCMs transmitted (10%). Twenty-two transmissions were processed by a different IP address, while nine other transmissions were processed by the CCS earlier than the VCM transmission time. These meant that only 33 or 17% of the samples transmitted without issues.

4.1. The 2016 NLE Internetwork Architecture

Based on the IP addresses in the logs, the election transmission was apparently through a "private" network. All IP addresses seen in the logs were IETF RFC 1918 private IP addresses. Private IP addresses are not routable in the internet. According to COMELEC resolutions [34] and [31], data transmission of ER and CCS used USB modems with telco provider SIM cards. For this to work over private IP addresses, COMELEC would have subdivided the election transmission network into subnetworks. Each subnetwork is assigned to a different telecommunications provider. While in the Philippines different telco providers rarely share last mile circuits, the only way this can be done was for COMELEC to have a multi-layer switch or router where all telco providers can interconnect. The data center where all the gateway routers of the telecommunications providers were interconnected for the purpose of the election was coined the “meet-me-room” and was an integral element of the 2016 election transmission network.

Owing to the differences in the network architecture of each individual telco provider, Network Address Translation (NAT) was apparently used to segregate the network. Figure 11 shows a sample set-up where the true IP addresses of any transmission device were hidden behind the edge / border routers of both telcoA and telcoB. If an IP address belonged to a duplicate subnet in another telco route, then both telcoA and telcoB will not be able to route packets to each other without using NAT.

As discussed in Chapters 2.2.2 and 2.2.3, the transmission network used multiple media. To comply with the COMELEC-assigned IP subnet for each telco, telco providers used a dedicated circuit for the election, perhaps using Multi

Protocol Label Switching (MPLS) or Virtual Private Network (VPN). To route the packets from a 3G network to dedicated circuits, the SIM should be configured with the appropriate Access Point Name (APN) settings.

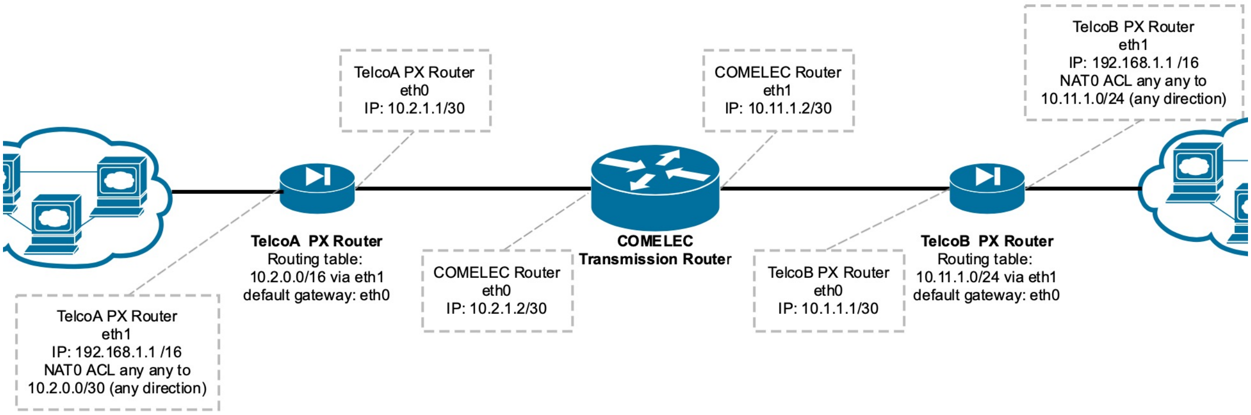


Figure 11. Sample Network Diagram Showing NAT in Action

The DNS is another interesting network component. The 2016 Automated Election System (AES) DNS logs showed that the network used the domain “pili2016pinas.net". The CCS destination was identified by its 4-digit MBOC code (refer to Table 6), followed by the subdomain “ccs” (e.g., “0507.ccs.pili2016pinas.net"). The IP addresses of approximately 1,700 CCS and more than 92,000 VCMs were impossible to know in advance. For the VCM to transmit to the target CCS using the CCS fully qualified domain name (FQDN), COMELEC needs a Dynamic DNS (DDNS) for “pili2016pinas.net" zone. The DNS logs also showed that secure DNS was not used because the name resolver requests were in plaintext.

According to [33], the ICE certification of the AES complied with the US Election Assistance Commission (EAC) Voluntary Voting System Guidelines (VVSG) version 1 (2005) framework. Although the IP addresses indicated a private network, the transmission medium was shared, given the telco circuits used were public in nature. The use of OSI layer 3 private IP addressing scheme does not automatically constitute a strictly “private” transmission. Even if the telco provider provided an OSI layer 2 MPLS type of circuit, or even a dedicated last mile connectivity, packets may have converged on a shared core network. Regardless of the network architecture that used private IP addresses, the use of a commercial network over a shared physical medium meant that Section 7.5.2 "Protection Against External Threats" of the VVSG version 1, volume 1 applied. Smartmatic and COMELEC should have submitted the documentation of all Commercial Over the Shelf (COTS) devices used in the transmission network as required by [51]. But in the list of documents enumerated in [33], these were not provided. The ICE certification report also mentioned that the telecommunications test was limited. COMELEC was even cautioned to require the vendor to provide its configuration plans to ensure that the entire AES worked in the same manner as it was tested, which would have alluded to the same provision in the VVSG version 1. There was no documentation that the configuration of the routers, switches, and other network elements in the "meet-me-room" were submitted by COMELEC to either the ICE or TEC.

4.2. Election Events Based on VCM and CCS Audit Logs

An analysis of the audit logs revealed possible irregularities during the 2016 NLE (Table 6). The summary file produced by the python programs is in Appendix 4 and contains a more detailed enumeration of the events found.

Table 6. Summary findings on VCM and CCS Log Analysis

Item #	Findings	Classification	Total Occurrences
1	**Rezeros after actual votes counted**	Critical	4 (2.08%)
2	**CCS received earlier than VCM Transmitted**	Critical	13
3	**CCS successfully processed results not from the VCM IP address, but from 10.101.1.198**	Critical	22 (11.46%)
4	**CCS successfully processed a transmitted result but VCM did not transmit**	Critical	2 (2.44%)
5	**VCM transmission successful but CCS did not receive the transmission**	Critical	19 (9.9%)
6	**Ballots inserted after the last counted ballot (uncounted ballot)**	Major	20
7	**First ballot fed on May 9, but earlier than official election time**	Major	31 (16.5%)
8	**Early transmissions**	Major	16 (8.33%)

Item #	Findings	Classification	Total Occurrences
9	**Machine changes**	Minor	7
10	**VCM transmission failure**	Minor	82 (42.71%)

The observed possible irregularities were classified by this research as follows:

1. *Critical* - This irregularity can possibly invalidate the election result and can be used as evidence for a case of electoral sabotage.
2. *Major* - While the entire results may not be invalidated, this irregularity poses serious threat to the election's information security.
3. *Minor* - This irregularity can put into question the security of the automated election, but if considered in isolation, may not be enough to be used for electoral protests or to declare a failure of elections.

4.2.1. Re-zero of VCM after the Opening of Elections

Table 7 shows the number of votes already in the four VCMs when they were re-zeroed. The timestamps also indicated that the election was redone a day later. All VCMs belonged to MBOC 3629 (Tugaya, Lanao Del Sur).

Table 7. Details of Rezero that Affected Actual Votes

VCM ID	Votes Affected	Time of last ballot case	Time of Rezero	Votes cast after Rezero	Time of last ballot cast
36290005	256	2016-05-09 20:16:34	2016-05-10 21:30:06	329	2016-05-11 03:10:14.339
36290014	156	2016-05-09 14:37:07	2016-05-10 21:28:07	218	2016-05-11 00:40:22.101
36290016	442	2016-05-09 19:24:30	2016-05-10 21:35:50	654	2016-05-11 08:26:42.948
32690019	155	2016-05-09 13:38:11	2016-05-10 21:28:43	362	2016-05-11 03:26:28

A re-zero of the VCM should require a higher-level approval, preferably from a COMELEC election officer, because it erases all ballot images and resets the election state of the VCM (Chapter 2.2.2). But the COMELEC website did not show any approved resolution for a re-election to be done in Tugaya. The timing of the re-zeros was also questionable. The re-zeros happened past 9 pm of May 9 , 2016 (election day). The election was reopened afterwards. There is a high probability that the voters did not scan the ballots themselves to the VCM because the scanning of ballots was done fin the evening of May 10 (one day after the election) to the morning of May 11. As discussed in Chapter 2.2.2, the BEI is allowed to feed the ballots on behalf of the voters under special circumstances such as if the VCM broke down and needed replacement. However, the audit logs showed that the VCMs listed in Table 8 were not replaced. There was also no circumstance outlined in official COMELEC resolutions that authorized a re-zero of votes especially when a significant number was already counted by the VCM. This is therefore a serious issue that COMELEC should investigate.

The logs additionally showed that all 25 VCMs assigned to MBOC 3269 (Tugaya, Lanao Del Sur) were in an "open voting" election state at around 1:00 to 2:00 am of May 9, 2016 and the BEI performed what appeared to be an FTS (Chapter 2.3.2). Official COMELEC resolutions scheduled FTS only from May 2 to May 6, 2016 and required the presence of poll watchers.

4.2.2. Transmission of 19 VCMs but No Record of Receipt in CCS

Still in the town of Tugaya, Lanao Del Sur, 19 out of 25 VCMs transmitted their ER successfully, but the MBOC CCS showed no record that the ERs were received. This implied either of the following:

1. The VCM or the CCS audit logs were inaccurate.
2. The VCM audit log was accurate, but the configured destination in the VCM was different.

Both have implications on the election's integrity. If the first premise is true, the SLI certification was inappropriate. Furthermore, the certification of the SAES 1800*plus* as fit for election may have been premature (Chapter 2.4.3). If the second premise is true, then a question arises on what computer received the ER and whether they were counted, double-counted, or not counted[11].

4.2.3. Processing of 22 ER by the CCS from a Different IP address

Three VCMs in Tugaya, 32690009, 32690015 and 32690020, transmitted but the CCS processed the received ER from a different IP address. The MBOC CCS processed the transmitted ER from IP address "10.101.1.198". This issue was not

[11] As discussed in Chapter 2.3.4, the ER from the VCM is manually uploaded to the CCS regardless of whether the ER was transmitted or not. The CCS knows if the ER were received prior the uploading. But in this instance, the ER could have been misrouted to a different CCS and it is not known how it could have been processed.

isolated to VCMs in Tugaya. The IP address “10.101.1.198” appeared 78x as a source IP address in the MBOC CCS logs. In these 78 instances, 22 were successfully processed by the MBOC CCS (Table 8), while the other 56 were identified by the CCS as containing duplicate ER and thus, were not processed.

Table 8. Successfully Processed ER from a Different IP: 10.101.1.198

VCM ID	VCM Transmit IP	CCS Received IP	Transmit Received (CCS)
05070002	10.11.98.105	10.101.1.198	2016-05-09 17:50:23
05070014	10.12.83.240	10.101.1.198	2016-05-09 18:39:51
05070019	10.12.60.201	10.101.1.198	2016-05-09 20:00:11
05070021	10.12.114.185	10.101.1.198	2016-05-09 23:02:11
05070024	10.19.15.59	10.101.1.198	2016-05-09 17:49:39
05070026	10.19.70.77	10.101.1.198	2016-05-09 23:17:38
05070027	10.19.51.104	10.101.1.198	2016-05-09 18:53:54
05070034	10.11.70.177	10.101.1.198	2016-05-09 17:57:27
05070041	10.12.61.116	10.101.1.198	2016-05-09 17:47:34
05070045	10.11.4.164	10.101.1.198	2016-05-09 18:57:23
05070046	10.12.111.150	10.101.1.198	2016-05-09 22:38:43
05070049	10.12.105.34	10.101.1.198	2016-05-09 22:02:44
05070053	10.11.71.142	10.101.1.198	2016-05-09 17:49:51
05070054	10.11.129.176	10.101.1.198	2016-05-09 18:48:28
05070058	10.12.24.144	10.101.1.198	2016-05-09 19:09:56
05070068	10.12.14.245	10.101.1.198	2016-05-09 18:51:06
05070070	10.11.97.102	10.101.1.198	2016-05-09 19:07:20
05070073	10.11.20.187	10.101.1.198	2016-05-09 18:17:18
05070074	10.12.94.46	10.101.1.198	2016-05-09 19:50:03
36290009	10.12.123.216	10.101.1.198	2016-05-10 07:53:48
36290015	10.11.4.194	10.101.1.198	2016-05-10 07:53:35
36290020	10.12.123.72	10.101.1.198	2016-05-10 07:55:04

Among the 22 VCMs listed in Table 8, two VCMs had the CCS receive a "retransmission" from their own VCM IP addresses (Table 9). The CCS logs of VCMs *05070034* and *05070073* revealed that the machine with IP address “10.101.1.198" beat them in first transmitting an ER to the CCS, but the CCS also received succeeding transmissions from 10.11.70.177 and 10.11.20.187 respectively albeit, containing duplicate ERs.

With the data in Table 9 showing the CCS receiving the same ER from different IP addresses including “10.101.1.198” and 56 other retransmission received by the CCS using either the VCM IP addresses or the "10.101.1.198", it can be concluded that “10.101.1.198" is not an interface NAT IP address similar to what was discussed in Chapter 4.1. Evidence indicates that the IP address “10.101.1.198” belonged to a separate computer with the ability to independently transmit an ER.

Table 9. Processed by 10.101.1.198 but CCS also received from VCM IP

VCM ID	Transmit Time	Transmit IP	CCS 1st Received IP	CCS 1st received time	CCS 2nd received IP (Not processed)	CCS 2nd received time
05070034	2016-05-09 18:30:45	10.11.70.177	10.101.1.198	2016-05-09 17:57:27	10.11.70.177	2016-05-09 18:31:05
05070073	2016-05-09 17:12:39	10.11.20.187	10.101.1.198	2016-05-09 18:17:18	N/A	N/A
05070073	2016-05-09 18:28:29	10.11.81.243	N/A	N/A	10.11.81.243	2016-05-09 18:28:49

4.2.4. VCMs did not Transmit but the CCS Processed a Transmission

Among the 82 VCM that did not transmit their ER (Figure 10), two VCMs from Libon, Albay had their ERs processed from “10.101.1.198”.

As discussed in Chapter 2.3.4, ERs from the main SD card were uploaded to their CCS by the BOC. During which, the CCS logs showed the ERs processed from the loopback IP address 127.0.0.1.

However, in the case of VCM 05070013, a transmission from "10.101.1.198" was first successfully processed on May 9, 2016, 10:14:48 pm. The ER was processed by the same CCS from the loopback IP address 38 minutes later, at 10:42:19 pm. To reiterate, VCM 05070013 did not successfully transmit its ER based on the VCM logs.

The case of VCM 05070012 was slightly different. The CCS was processing different fragments of the election results simultaneously from the 127.0.0.1 and “10.101.1.198" on 11:34 pm of May 9, 2016. Based on the CCS logs, the ER from VCM 05070012 was being processed in smaller fragments but from different sources:

```
23:34:10,869 INFO  [com.smartmatic.saes.listener.mdb.util.StationTallyValidatorHelper] (Thread-1664 (HornetQ-
client-global-threads-608921159)) The information from results report [05070012.01005079.6] transmitted by
device [05070012] from Ip [127.0.0.1] was successfully processed

23:34:11,013 INFO  [com.smartmatic.saes.listener.mdb.util.StationTallyValidatorHelper] (Thread-1658 (HornetQ-
client-global-threads-608921159)) The information from results report [05070012.00905079.9] transmitted by
device [05070012] from Ip [127.0.0.1] was successfully processed
23:34:11,080 INFO

23:34:12,352 INFO  [com.smartmatic.saes.listener.mdb.util.StationTallyValidatorHelper] (Thread-1665 (HornetQ-
client-global-threads-608921159)) The information from results report [05070012.00405009.1] transmitted by
device [05070012] from Ip [10.101.1.198] was successfully processed
```

The transmission report can be observed in batches but labelled numerically. The first ER data block processed from the loopback IP address was "05070012.01005079.6". This was followed by 05070012.00905079.9. The results report sequence 050070012.00405009.1 was processed from "10.101.1.198".

The simultaneous processing of the transmission from different sources leads to a questionable data origin authentication mechanism of the 2016 AES. But more critical is the question on how the CCS was able to process a transmitted result from “10.101.1.198” when VCM 05070012 did not transmit as per the VCM logs.

4.2.5. CCS processed ER before the VCM Transmitted

Table 10 shows instances when the CCS received a transmission earlier than the VCM transmitted the ER. Three VCMs 05070034, 36290009, and 05070049 (labelled red) had their ER processed from “10.101.1.198“ (Chapter 4.2.3), while the other ERs were received by their CCS from the VCM IP address.

Table 10. CCS Received Time is Earlier than VCM Transmit Time

VCM ID	Time VCM Transmitted	Time CCS Processed	Time Difference
05070034	2016-05-09 18:30:45	2016-05-09 17:57:27	-33m 18s
36290009	2016-05-10 08:23:33	2016-05-10 07:53:48	-29m 45s
05070001	2016-05-09 20:56:18	2016-05-09 20:49:34	-6m 44s
05070042	2016-05-09 18:51:28	2016-05-09 18:46:13	-5m 15s
05070010	2016-05-09 18:58:15	2016-05-09 18:54:27	-3m 48s
05070049	2016-05-09 22:04:24	2016-05-09 22:02:44	-1m 40s
05070038	2016-05-09 21:38:24	2016-05-09 21:38:03	-21s
05070071	2016-05-09 18:30:00	2016-05-09 18:29:46	-14s

VCM ID	Time VCM Transmitted	Time CCS Processed	Time Difference
05070043	2016-05-09 18:55:34	2016-05-09 18:55:21	-13s
05070022	2016-05-09 18:45:57	2016-05-09 18:45:47	-10s
05070067	2016-05-09 18:50:15	2016-05-09 18:50:11	-4s
05070015	2016-05-09 18:30:26	2016-05-09 18:30:24	-2s

This phenomenon indicated that the 2016 AES was not using a Network Time Protocol (NTP), nor did it use the system clock for its anti-replay mechanism. It was observed that duplicate ERs received by the CCS were rejected, indicating presence of an anti-replay mechanism, perhaps using nonces.

The problem with not using the system clock for an AES is the difficulty to ascertain its accuracy. Not using the system clock for its anti-replay mechanism is a 2016 AES critical design flaw. In elections, Time of Check and Time of Use (TOCTOU) is inimical to the AES' integrity. Votes can only be counted within a certain period. Time is also critical in many AES decisions, such as in considering which among many transmitted results are official.

4.2.6. Early Transmissions

Table 11 shows the details of the 16 VCMs that transmitted between May 1 and May 9 midnight. As discussed in Chapter 2.3.2, no transmission should be received before and during the FTS scheduled on May 2 to 6, 2016.

Table 11. Details of 16 Early Transmissions

VCM ID	Date and Time of Transmission
58010015	2016-05-03 10:24:23
58010017	2016-05-03 10:32:24
58010019	2016-05-03 10:56:25
58010018	2016-05-03 10:58:33
58010012	2016-05-03 10:59:59
58010011	2016-05-03 11:00:29
58010013	2016-05-03 11:16:57
58010014	2016-05-03 11:51:59
58010039	2016-05-03 14:40:46
58010034	2016-05-03 14:44:08

VCM ID	Date and Time of Transmission
58010048	2016-05-03 14:47:13
58010049	2016-05-03 14:53:53
58010047	2016-05-03 14:58:55
05070038	2016-05-06 11:13:31
05070035	2016-05-06 12:19:07
05070045	2016-05-06 13:49:13

During the hearings of the JCOC [52], COMELEC admitted that BEIs may have committed a mistake and attempted transmission during FTS. The MBOC logs did not indicate that the results were received from the "attempted" transmissions. Either that or some logs were not submitted to JCOC.

4.2.7. Implications of the "10.101.1.198" Findings

The presence of "10.101.1.198" as a computer that transmitted ERs to the CCS leads to a modification of the transmission flow originally shown in Figure 6. Figure 12 shows the modified transmission flow.

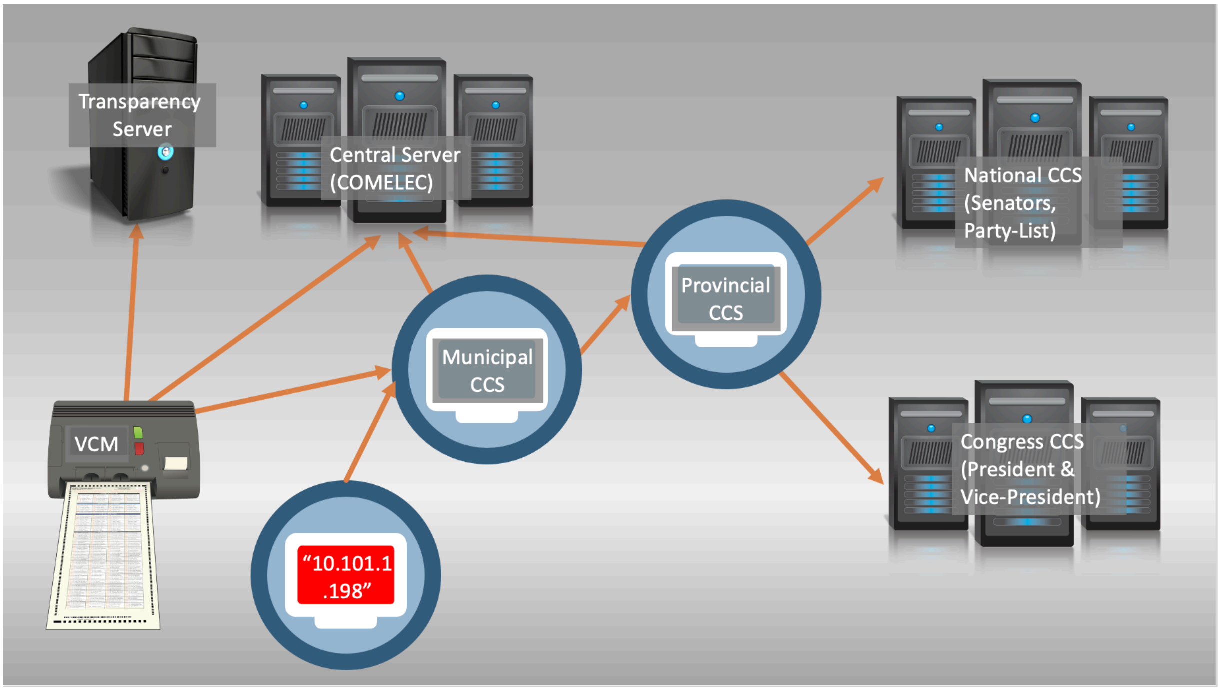


Figure 12. Modified Transmission Flow due to "10.101.1.198"

The presence of "10.101.1.198" puts COMELEC in a paradoxical position where any posited explanation leads to an unsatisfactory conclusion.

Assuming that “10.101.1.198" was known and authorized by COMELEC, "10.101.1.198" should have been subjected to the ICE and TEC certification process outlined in Chapter 2.4. As [33] claimed to have followed the US EAC VVSG testing guidelines, this computer should have been included in the system integration test as per [53]. Assuming that it was indeed tested, SLI and the TEC would then have to answer on how the 2016 AES passed the certification when clearly the audit log mechanism was inaccurate, considering the following:

1. In instances when the CCS processed an ER transmitted by "10.101.1.198" first before the VCM (Chapter 4.2.3) or when the VCM did not transmit but the CCS processed an ER (Chapter 4.2.4), where did the ER come from?
2. If the ER also came from the VCM where the ballots were scanned, why were the VCM logs not showing transmissions to 10.101.1.198?

However, the assumption that “10.101.1.198” underwent ICE and TEC certification could be wrong. The SLI certification report indicated that only the EMS, VCM, and CCS were tested. The Technical Documentation Package (TDP) also did not include any device from the transmission medium, nor was the network design / architecture included. Therefore, “10.101.1.198” most likely did not undergo the same certification as the three main components of the 2016 Automated Election System (AES). If this was the case, and in lieu of what was illustrated in Chapters 4.2.3 and 4.2.4, COMELEC should answer on how the 2016 AES was able to comply with the VVSG recommendations and with the AES Law of 2007 when an uncertified computer transmitted by assuming the machine ID of multiple VCM and, in some cases, transmitted before the VCM transmitted.

Alternatively, “10.101.1.198” can be another type of VCM. If this was true, then the configuration files used for “10.101.1.198” was clearly different than the other VCMs because unlike the others, “10.101.1.198” can assume the identity of different VCMs when it transmitted. This behavior was never recorded for a regular VCM. In this case, the ”10.101.1.198” VCM can still be classified as uncertified computer because while it could have been a VCM, it was not using the certified software / configuration. It could also be seen as a circumvention of the trusted build protocol (see Chapter 2.4.3).

Another issue that can be raised with the presence of "10.101.1.198" is its potential to weaken the information security of the 2016 AES. For example, if "10.101.1.198" can decrypt the election results and encrypt it again before it is sent to the CCS, then all the more should "10.101.1.198" be subjected to the ICE and TEC certification process because, in this case, the computer was implicitly trusted[12]. The alternative scenario could be that "10.101.1.198" was simply forwarding digitally signed results. That it neither had the encryption keys nor can it verify authenticity of transmitted results. Unfortunately, this alternative explanation did not remove the system's implicit trust on "10.101.1.198". It is undeniable that the CCS implicitly trusted "10.101.1.198" because the CCS processed ERs from it.

Lastly, COMELEC can simply deny that it authorized "10.101.1.198" to transmit. If this was the case, then COMELEC must investigate how an unauthorized computer was able to send election results that the CCS successfully processed. This can put the integrity of the entire 2016 AES at serious risk.

4.3. Theoretical Attacks against the 2016 AES

Some discussions in this chapter shed light to the creation of election security policies. The published conference proceedings of [54], despite being held in 2003, are still relevant today. No AES can be regarded as fully secure. Therefore, manual procedures to mitigate risks must be considered in assessing security of an AES. To mitigate risks, theoretical attacks must be considered.

4.3.1. Attacks against the USB port of the VCM

A working USB port can be a dangerous entry point for hackers. Early transmissions were considered a "major finding" for this research because they exposed the USB port of the VCM prematurely.

In theory, a keyboard can be inserted into the USB port. Because the VCM operating system is Linux, a Ctrl-Alt-F1 or Ctrl-Alt-F2 command can invoke the TTY (TeleTYpewriter) interface. This attack is naively simple and it is expected the TTY interface of the VCM to have been disabled. But other more complex ways to

[12] In information security, an implicit trust is a consensus that the system is secure. For example, it is generally assumed that the Access Control Server (ACS) used to authenticate card holders of an issuing bank is trustworthy. Another example is the case of a root certificate. The root certificate in a certificate chain is implicitly trusted and no further proof is required to check its authenticity. Implicit trust is a recurring theme in information security.

exploit USB vulnerabilities have been documented. Nissim in [55] enumerated a number of security attacks to the USB port. One of these attacks connected a rooted android smartphone running the “NetHunter” app to the USB port of a vulnerable machine to override the default gateway settings of the device. In a similar manner, a threat actor can use their android mobile phone running NetHunter, instead of the authorized 3G USB dongles for the election, to misroute traffic. For this attack to be successful, the attacker should know the APN settings of the SIM card used by any of the telco companies during the election (Chapter 4.1). If this attack have been employed, this potentially explains the unauthorized presence of “10.101.1.198".

USB attacks are diverse. Their capabilities range from changing network configurations to changing payloads. Various attacks can manipulate the payload of transmission devices remotely [55]. A malicious code can also be inserted remotely to change the manner by which a VCM interprets votes. This can be as simple as ignoring intersections of fiduciary marks, causing the VCM to incorrectly read marks on a ballot.

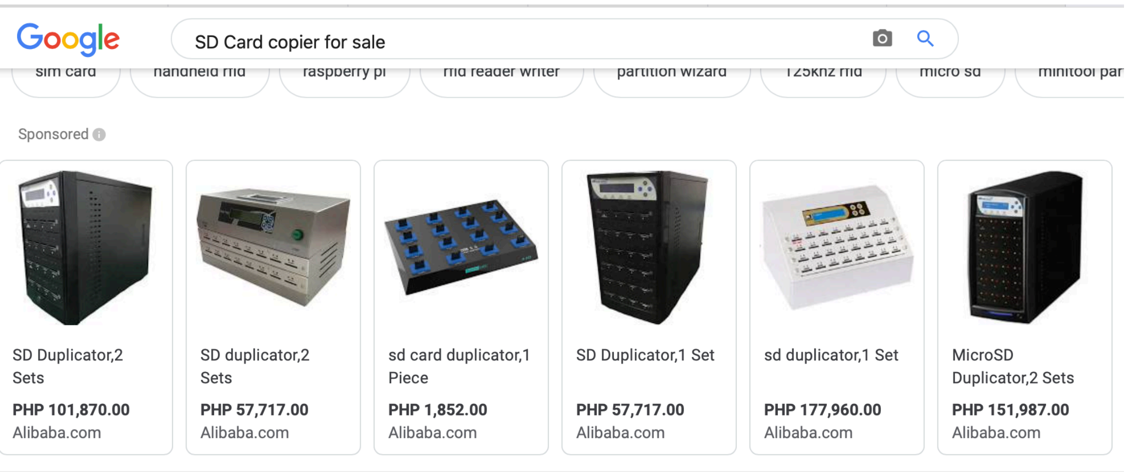
A comprehensive list of measures to mitigate the threat of USB-based attacks is provided in [56] and is summarized below:

- Enabling logical access controls such as disabling BIOS features for USB ports (e.g., limiting the use of USB for telecommunication devices as data dongles during elections);
- Enabling audit logs of USB port use;
- Disabling autorun when a USB device is inserted;
- Disabling auto-mounting features;
- Disabling auto-installation of drivers from the USB storage device;
- Not running USB devices in admin or root privileges;
- Using protection software such as anti-malware;
- Encrypting the transfer of data between the USB device and the computer;
- Using biometric and password authentication when using a USB device; and
- Restricting user access to use USB devices (e.g., putting locks to the cover of the USB port in the SAES 1800 plus VCM model, such that the BEI will have to open the lock using a key if they intend to use the USB port).

4.3.2. SD Card Swapping Attack

During a controlled demonstration of the SD cards in the source code review conducted for the 2016 NLE, tampering the results.xml or any file inside the SD card corrupted its contents and caused the VCM to fail. However, the contingency procedures of the 2016 election (Chapter 2.2.2) published in a COMELEC resolution revealed the possibility of a SD card swapping attack for the VCM by actors who has access to official election paraphernalia including the iButtons and their corresponding pin codes. The sample audit logs showed that VCMs were replaced in seven clustered precincts (refer to Appendix 4). During the VCM replacement, the VCM SD card slot B that should remain locked at all times [34 Sec. 29.c.] may have been opened, and the SD cards were replaced or moved to the new VCM. The re-zero events in Tugaya, Lanao Del Sur (Section 4.2.1) should also be investigated using this attack as a context. The fact that 25 VCMs in Tugaya were in opened election state suspiciously at around 1:00 am of May 9, 2016 should likewise be probed.

To carry out the attack, threat actors need an SD card cloning device, such as those shown in Figure 13, and an iButton cloning device. Legitimate SD Cards and iButtons need to be cloned, and a spare VCM should be available. To open the election, the 4-digit pins of the BEI members must also be known. Although, the pins can easily be discovered using 'brute force attack' because there is no maximum number of retries for pin numbers in the SAES 1800plus.



SD Card copier for sale

sim card
handheld rfid
raspberry pi
rfid reader writer
partition wizard
125khz rfid
micro sd
minitool par

Sponsored

SD Duplicator,2 Sets
PHP 101,870.00
Alibaba.com

SD duplicator,2 Sets
PHP 57,717.00
Alibaba.com

sd card duplicator,1 Piece
PHP 1,852.00
Alibaba.com

SD Duplicator,1 Set
PHP 57,717.00
Alibaba.com

sd duplicator,1 Set
PHP 177,960.00
Alibaba.com

MicroSD Duplicator,2 Sets
PHP 151,987.00
Alibaba.com

Figure 13. Varying SD Card Duplicators in the Market

To illustrate how the attack could be performed, suppose that Bob and Alice are BEI members and have access to VCM 1 and VCM 2. Both also have access to legitimate election SD cards A and B inserted in VCM 1. Both additionally have access to legitimate ballots and the iButtons. An election duplicate can thus be created as follows:

1. Before election opens at 6:00 am on election day, Alice and Bob use the SD card duplicator and any normal SD card to duplicate SD cards A and B. The iButtons are also copied. Bob now has copies of SD cards A and B. These are labelled SD cards A-1 and B-1, respectively.
2. Bob creates another copy of the empty SD card A-1 and labels it as SD card A-2.
3. Bob opens an election using VCM 2 at roughly the same time as official election opens, using SD Cards A-1 and B-1. Bob should know the pin codes for each iButton he copied.
4. Bob scans legitimate ballots on VCM 2. Bob uses spare legitimate ballots (discussed in Chapter 2.2.4) to continue the "duplicate election" in VCM 2. Bob need not use all ballots. He can continue to scan a number of ballots that can statistically allow a chosen candidate/s to win. It is rare to use all ballots because there are 20% excess ballots and rarely does a precinct have a 100% voter turn-out.
5. At midday, Bob informs Alice that he was able to scan X number of ballots into VCM 2. He then hands over SD card A-1 (only one is required) to Alice.
6. Alice watches closely as the ballots in VCM-1 are counted. Once the same ballot count as Bob's in VCM-2 is reached, Alice reports a VCM failure. Alice then restarts VCM 1 as part of the troubleshooting but, this time, replaces SD card A with SD card A-1. She inserts the contingency card into slot B. This is a troubleshooting step authorized by COMELEC as per [35]. After reboot, the empty contingency card inserted into slot B synchronizes with SD Card A-1 (discussed in Chapter 2.2.2). Election continues as usual.
7. Alice hands SD card A to Bob. While it was used earlier, SD card A already contains a certain amount of ballot images. Bob reformats SD card A by cloning it using the image of SD card A-2. Note that SD Card A-2 is a replica of SD card A when it was empty. Bob then hands over the "reformatted" SD card A back to Alice.

8. To hide her use of a copied SD card, Alice again calls a VCM failure at an opportune time. She places the contingency card to slot A, throws out SD card A-1, and inserts the now empty official SD card A in VCM card slot B. Upon reboot, the SD cards are resynchronized.

9. After resynchronization, she announces to the public that she is rebooting the VCM again to be certain that the VCM will not have any more problem, but her real agenda is to swap the SD cards in slots A and B to where they should be: SD Card A in slot A, and the contingency card in slot B.

After step 9, both SD cards would be where they should be. The audit logs would show the time Bob opened the election as the "election time". An auditor looking at the logs would simply see everything to be "normal".

To mitigate this attack, COMELEC can conduct the following:

- The audit logs of the different components of the AES should not reside solely on a removable memory. Either a copy or a small set of the audit logs should be stored in an internal memory location within the VCM or the CCS.
- A strict chain of custody rule for removable memories, ballots, and VCM should be implemented.
- Physical locks should be used for locking the SD card slots. Even if the VCM are to be replaced, the original WORM SD Card in slot B should not be removed. This way, the "original audit logs" are preserved. A new set of SD cards can be used with the new VCM.
- The BEI should be provided a facility to change the default iButton pin codes issued to them.

4.4. Assessing the 2016 NLE using the AES Trust Model

Having recreated the events of the 2016 presidential automated elections based on the audit logs, the election can now be assessed using the "Automated Election System (AES) Trust Model" developed in Chapter 3.1 using the following criteria[13]:

- <u>Fail</u> — AES trust property was not sufficiently observed and followed.
- <u>Pass</u> — Sufficient controls were in place to ensure that the AES trust model property was followed. The controls can either be technological or procedural.

[13] The criteria were developed solely for this research.

If the latter, there should be sufficient proof (i.e., the audit logs) that procedural controls were followed.

- *Excellent* — Multiple layers of controls were in place to ensure the trust model property was followed. There was strong evidence that the controls were implemented.

Table 12 summarizes the findings based on the preceding discussions. Numbers in parentheses refer to the chapters where these items were discussed.

Table 12. Assessment of the 2016 NLE using the AES Trust Model

AES Trust Model Property	Score	Positive observations	Negative observations
1. Privacy of Voters	Pass	• Law mandated the physical layout of the precinct to consider privacy of voters (2.1) • Taking photos inside the precincts were not allowed. (2.1) • The VCMs had identical timestamps of ballot images (2.2.2)	• It was unclear if the VCM employed a shuffling mechanism because the order of the ballots can still be considered when tracing voters. • The ballots had unique bar codes which can be "known" to the BEI (2.2.4)
2. Uncoercibility / Receipt Freeness	Pass	• Procedural safeguards were in place, such as not allowing VVPAT to leave the precincts and not allowing voters to take photos while voting.	• The AES was not proven to be enforcing uncoercibility such as using homomorphic encryption in transmission of data. Although, the use of homomorphic encryption may not be necessary since there was no online facility for voters to verify their votes.
3. Individual Verifiability	Fail	• Presence of VVPAT (2.3.3) • Option for voter to question their votes in the MOV (2.3.3)	• Voters were not allowed to change their ballots if they committed a mistake (2.2.4). • While voters knew how the VCM counted their votes via the VVPAT, the voters did not know how their votes were appreciated during the canvassing of votes due to the multi-tier consolidation and canvassing of election results (2.3.4).

AES Trust Model Property	Score	Positive observations	Negative observations
4. Universal Verifiability	Fail	• The RMA showed the results were 99.9023% (1.1 & 2.3.5), thus the results were reproducible; • AES certification process and local source code review were commendable. • The ballot validation procedure was standard security practice.	• Key elements of the AES, such as authentication mechanism, cryptographic protocols, random generators, and shuffling / queueing mechanisms, were not published. • Unexplained presence of "10.101.1.198" questioned the validity of audit logs (4.2.7). • Lack of timestamp validation questioned validity of results (4.2.7). • Results of Local Source Code Reviews were not published (1.2). • The cryptographic protocol used cannot be verified.
5. Fairness	Fail	• There was no partial results being transmitted. • VCM cannot print a partial ER	• BEI were able to rezero (4.2.1) • BEI can scan ballots in behalf of voters when a VCM malfunctions (2.2.2).
6. Integrity	Fail	• Publication of hash codes and printing the hash codes to every printed election results and document. • Trusted Build ceremony shows certified software was compiled and installed to machines.	• BEI were able to rezero after an election, without published resolution or notice (4.2.1) • "10.101.1.198" was able to transmit before the VCM (4.2.3) • Early use of USB ports (4.2.6) increases possibility of a USB attack (4.3.1). • There was no clear published protocol to mitigate SD Card Swapping Attack (4.3.2). • BEI iButton and BOC security token pin codes were predetermined and distributed without any facility to change them (4.3.2). • The lack of timestamp validation can increase possibility of TOCTOU attacks (4.2.5) • Voter validation was still manual. Voters without any official ID were allowed to vote (2.3.3).
7. Availability	Pass	• VCM were certified by ICE to withstand harsh environment including dust, heat, etc. (2.2.2) • The VCM can run on a 12V battery for 14 hours (2.2.2) • Signal jammers were prohibited by law. • Results can be manually uploaded (in case transmission failed)	• Availability attacks on the VCM can be done by illegitimately marking the fiduciary marks, or the bar code area of the ballots. This is a common attack in all paper-based election.

AES Trust Model Property	Score	Positive observations	Negative observations
8. Non-repudiation	Fail	• Two-Factor authentication for BEI and BOC (2.2.2 and 2.2.3)	• There were no registration of BEI to link identities with digital signatures. • There was no real-time authentication mechanism for VCM at time of entity authentication. The VCM were offline all the time (2.2.2). • Timestamps as critical element of an election were not employed as part of the authentication mechanism (4.2.5). • There was no registration of BEI and BOC to the PKI that happened. Instead, generic "identities" were used, and pin codes were given during FTS and election day to BEI and BOC members.

Of the eight listed criteria, the 2016 AES failed on the properties of universal verifiability, fairness, integrity, and non-repudiation. This failure was mostly due to the presence of "10.101.1.198" and its implications (Chapter 4.2.7) including the ability of the said machine to transmit before the VCM transmitted, and the non-use of timestamps as as an anti-replay mechanism. The re-zeros that happened in 4 VCMs in Tugaya, Lanao del Sur is also interpreted negatively against the property of fairness in the trust model.

5. Conclusion and Recommendations

The road to automation of the Philippines has been long and arduous. After election automation in 2010, the political culture remained unchanged. Automation, however, brought certain advantages. One was the significant reduction of cheating during and after elections. Voting, counting, and canvassing also became more efficient.

Due to the country's turbulent history in the conduct of elections, there remains much distrust in the electoral process, and people clamor for more transparency. Transparency should not only refer to removing obstacles to gaining information. Data that is made available should be easily recognizable and easily interpreted. This is not usually the case in an automated elections. In an Automated Election System (AES), the machine interprets how people voted, and there are cases when the machine interprets votes differently than the voter intended, or that the transmission of the votes erred.

The Random Manual Audit (RMA) was done as an assurance of the integrity of the election. However, a 99% result in an RMA does not mean the election results cannot be questioned. For example, the ballots can be switched, as illustrated in Chapter 4.3.2. The RMA only covers 0.74% of the entire clustered precincts and discussions in Chapter 4 showed that despite the RMA producing outstanding numbers, questions are still raised about the integrity of the 2016 NLE.

Thus, an alternative definition of transparency is offered. This definition is more attuned to computers and electronic data. Transparency should mean *to design, and divulge the process by which data is gathered and interpreted, such that the outcome becomes predictable and reproducible*. This definition is an improvement, although similar to that offered by [57]. Using this definition, transparency now denotes *auditability* and *reproducibility*, which is the heart of the verifiability property in the AES Trust Model used in this research.

Lack of transparency is mainly why the 2016 AES failed in the AES Trust Model. In the scoring, the AES failed in 4 out of 8 properties of the Trust Model: (1) universal verifiability; (2) fairness; (3) integrity; and (4) non-repudiation. There may be alternative explanations than the ones offered in Chapters 4.2.7 and 4.4 but due to limited literature, definitive conclusions cannot be made. Analyses made by this research brought to light an undocumented computer with IP address “10.101.1.198”. While Chapter 4.2.7 reiterated that no positive conclusion can be made with the presence of "10.101.1.198", the said computer may not have raised much controversy if it was simply subjected to a source code review.

Table 13 summarizes the recommendations to improve the AES based on the analysis made in Chapter 4. Majority of the recommendations are anchored on transparency, such as expanding the scope of the TEC certification and ensuring publication of cryptographic protocols used in the AES.

Table 13. Recommendations for Future AES

#	Applicable AES Trust Model Property	Recommendation
1	ALL	Publish in COMELEC websites all cryptographic protocols used for the AES.
2	ALL	Create the AES Law of 2007 Implementing Rules and Regulations (IRR) which includes a detailed AES technical and security selection criteria.

#	Applicable AES Trust Model Property	Recommendation
3	ALL	Amend election laws to remove the tiered process of canvassing and consolidation of election results.
4	ALL	Promulgate a chain of custody policy for all removable media that contains election results.
5	Fairness, Universal Verifiability, Integrity	Put more technology and policy controls in the re-zero capability of the BEI.
6	Non-repudiation	Use the official government PKI for the registration of BEI and BOC personnel.
7	Non-repudiation	Allow BEI and BOC to reset their pin codes and passwords.
8	Non-repudiation	Introduce a BEI and BOC registration process to create a non-repudiable relationship between the digital signatures and certificates, and people.
9	Integrity	Automate voter validation
10	Universal Verifiability, Integrity	Include audit logs in all future Random Manual Audits.
11	Universal Verifiability, Integrity	Expand coverage of the TEC certification and source code reviews to include network elements and configuration of the transmission infrastructure.
12	Universal Verifiability, Availability	Remove USB ports and instead use embedded machine transmission medium; or alternatively provide the hardening procedures to JCOC for all vulnerabilities of the AES such as but not limited to, the USB port and SD card swapping attack vectors (Chapter 4.3).

The first recommendation on the publication of all cryptographic protocols used for the AES in COMELEC websites is very important. While this may contradict some laws, such as copyright laws, privacy laws, and those that protect economic interests of private enterprises, the second principle of Auguste Kerckhoff in designing a system with the assumption that the enemy will have full knowledge of the system is worth noting. Kerckhoff's second principle states that a secure system must continue to be secure even if everything about the system, except the keys used to encrypt and decrypt data, is already known to the public. Security by obscurity is never advisable. COMELEC should adhere to this principle and develop a policy for more transparency of the AES. The AES used in Norway, for example, has their technical specifications, including the cryptographic protocols used, published and publicly available for review. The certification process for the AES in Norway also followed the Common Criteria.

The second recommendation is on the creation of an IRR of the AES Law. Since the AES law of 2007 was enacted, there had been no IRR promulgated. The IRR should include the following information security-focused recommendations:

1. The Trusted Computing Base (TCB) for each major component of the AES (VCM, CCS, EMS) and their corresponding secure elements, should be defined.
2. In relation to Recommendation #11, the IRR should identify the elements of the transmission network that must be subject to TEC certification.
    - 2.1. Configuration of off-the-shelf Customer Premise Equipment (CPE) network elements of the transmission infrastructure should be furnished to political parties, local source code reviewers, and the citizen's arm.
    - 2.2. Source code of non off-the-shelf network elements should be reviewed by TEC and the local source code review team.
    - 2.3. The entire election transmission infrastructure diagram must be published[14].
3. Define the standard for ICE and TEC certification. Currently, the certification process is at the discretion of the ICE and the AES supplier/s due to the lack of guidelines over what evaluation criteria to be followed. Models from the US EAC VVSG and the Council of Europe (CoE).
    - 3.1. The properties of the AES and their expected technical implementation must be outlined. These include:
        - 3.1.1. Definition of the privacy of voters and its relationship with unlinkability and anonymity. The technical implementation for this may include the use of proven and publicly verifiable shuffling algorithms.
        - 3.1.2.The property of uncoercibility and verifiability, including the required technical implementation, should be in the IRR. An example of a criteria for for this property is to decide if future AES requires the use of homomorphic encryption to allow voters to verify their votes online (individual verifiability), without giving voters the opportunity to prove to others who they voted for (uncoercibility). Homomorphic encryption allows a system to calculate the encrypted data, without decrypting it [58].
        - 3.1.3. Define the property of fairness and explicitly prevent any individual, including COMELEC officials, to see partial results.

[14] Care must be taken to not divulge critical information which may be used to sabotage the election. For example, the infrastructure diagram can be published, and academically reviewed without necessarily divulging the physical location of datacenters.

3.1.4. Define system “integrity” and its technical description. An example is to define whether mutual data origin authentication and machine entity authentication are required in the AES.

3.1.5.Define the SLA for availability for each component of the AES. The test requirements similar to the VVSG Vol. 2 [59] may be included.

3.1.6. In relation to Recommendation #6 and Recommendation #8 in Table 13, non-repudiation must be defined in the IRR, and the appropriate use of digital signatures including their cryptographic primitives, allowed key length, and protocol should be described.

The third recommendation about the removal of the tiered process of canvassing and consolidation of election results aims for simplicity to lessen human intervention in the counting of votes, as originally proposed by Monsod (Chapter 1.2). For each transmission of an election result from VCM to various CCS in the tier, transmission errors may be introduced. Management of cryptographic keys also becomes more complex in the order of 2^n, where n is the number of stages data was encrypted, decrypted, and processed. This multi-tiered transmission of election results should be removed. A set of fully redundant CCS can receive, and process the ER, and produce the COC, SOV and COCP necessary for proclaiming the winners at all levels. Unlike the current CCS, this new central CCS can operate as a server environment. BOCs in the municipal and provincial levels can connect to the CCS to examine and print the documents necessary to proclaim winners in various political administrative units. The central CCS can be made auditable, and its logs be included for analysis by the JCOC to ensure transparency. With only one set of CCS to receive and process ER, management of data, information security, and infrastructure of the elections becomes easier.

As discussed in Chapter 4.2.1, a re-zero is a critical function that should only be done under strict circumstances. The fifth recommendation on establishing more technology and policy controls in the re-zero capability of the BEI aims to prevent accidental or malicious re-zero of election results. For example, a technological control can be enforced where the BEI can only obtain a passcode necessary to re-zero the VCM only after it was approved by an Election Officer or other election authorities.

Non-repudiation as an information security property can only be established if there is a relationship between the person and the digital signatures. This was not the case in the 2016 AES because the BEI and BOC pin codes were generic, and there was no BEI / BOC registration done to associate each digital signature generated with the person. Recommendations #6 to #8 in Table 13 aim to change this.

COMELEC should look into the automated authentication of voters (Recommendation #9). This is a critical component of the election, which remains manual. The move for biometric authentication, however, should adhere to the requirements on voter privacy in the AES Trust Model.

The audit logs provided were useful in recreating the election. Recommendation #10 is for the audit logs to be included in the RMA. The current RMA only shows the end result and answers the question of whether the manual tabulation matches the electronic tally. The audit log can answer questions of whether the election was done properly and that no unauthorized erasures or transmissions happened. This is necessary to strengthen future AES' compliance with both individual and universal verifiability properties.

Lastly, Recommendation #11 was already discussed in Chapter 4.1, while the recommendation to mitigate USB port and SD card swapping attacks (Recommendation #12) was described in Chapter 4.3.

6. Recommended for Further Research

The Philippines is contemplating using internet voting (aka remote voting) for overseas Filipinos in the upcoming 2022 presidential elections [60]. Internet or remote automated election systems are more complex. The AES Trust Model developed in this research can be used as a guide for evaluating such systems. Subsequent studies can also examine internet voting using homomorphic encryption, efficient shuffling algorithms, and blockchain infrastructures.

Proposals on how to use homomorphic encryption in remote elections are discussed in [58] and [61]. Shuffling algorithms are also needed to ensure the privacy of voters. Kun Peng proposed a shuffling algorithm in [62].

The use of blockchain for internet voting is now gaining momentum. By combining blockchain-based election infrastructures with homomorphic encryptions and random shuffling algorithms, a blockchain-based AES that follows the proposed AES Trust Model may become mainstream. The combination of the three technologies may make it possible to administer an election without a central canvassing authority such as COMELEC. As discussed in Chapter 1.2, this was one of the original intents of the election law. The use of blockchain combined with homomorphic encryption to allow a coercion-free yet publicly transparent internet-based election that is free of a central counting authority can be found in [63] and [64].

Remote voting systems usually require a secure national voter identification system to authenticate legitimate voters. The Philippines is yet to implement a digitally secure national ID system that can be used for this purpose. The proper validation of legitimate voters before they can vote is crucial, and the non-repudiation property for this type of AES is critical.

Citations

[1] C. Calimbahin, "The promise and pathology of democracy: The Commission on Elections of the Philippines," Ph.D., The University of Wisconsin - Madison, Ann Arbor, 3367494, 2009. [Online]. Available: http://0-search.proquest.com.catalogue.libraries.london.ac.uk/dissertations-theses/promise-pathology-democracy-commission-on/docview/305035250/se-2?accountid=14565

http://resolver.ebscohost.com/openurl?ctx_ver=Z39.88-2004&ctx_enc=info:ofi/enc:UTF-8&rfr_id=info:sid/ProQuest+Dissertations+%26+Theses+Global&rft_val_fmt=info:ofi/fmt:kev:mtx:dissertation&rft.genre=dissertations+%26+theses&rft.jtitle=&rft.atitle=&rft.au=Calimbahin%2C+Cleo&rft.aulast=Calimbahin&rft.aufirst=Cleo&rft.date=2009-01-01&rft.volume=&rft.issue=&rft.spage=&rft.isbn=978-1-109-28241-2&rft.btitle=&rft.title=The+promise+and+pathology+of+democracy%3A+The+Commission+on+Elections+of+the+Philippines&rft.issn=&rft_id=info:doi/

[2] "Omnibus Election Code of the Philippines," Batasang Pambansa ed. Manila, 1985.

[3] B. Johnson, "Manila vote-count heroes fearful fugitives," in *Globe & Mail (Toronto, Canada)*, ed, 1986, p. A1.

[4] "An Act Providing For A General Registration Of Voters, Adopting A System Of Continuing Registration, Prescribing The Procedures Thereof And Authorizing The Appropriation Of Funds Therefor," ed. Philippines, 1996.

[5] "An Act Authorizing The Commission On Elections To Conduct A Nationwide Demonstration Of A Computerized Election System And Pilot-Test It In The March 1996 Elections In The Autonomous Region In Muslim Mindanao (Armm) And For Other Purposes," 9th Congress, 3rd Regular Session ed. Philippines, 1995.

[6] P. Eadie, "The 2010 Philippine National Elections: Perception and Reality," *EJEAS,* vol. 12, no. 1, pp. 55-79, 2013, doi: 10.1163/15700615-13120106.

[7] "An Act Authorizing the Commission on Elections to use an Automated Election System in the May 11, 1998 National and Local Elections and in Subsequent National and Local Electoral Exercises, Providing Funds Therefor and for Other Purposes," 10th Congress ed. Philippines, 1997.

[8] "An Act Amending Republic Act No. 8436, Entitled "An Act Authorizing The Commission On Elections To Use An Automated Election System In The May 11, 1998 National Or Local Elections And In Subsequent National And Local Electoral Exercises, To

Encourage Transparency, Credibility, Fairness And Accuracy Of Elections, Amending For The Purpose Batas Pampansa Blg. 881, As Amended, Republic Act No. 7166 And Other Related Elections Laws, Providing Funds Therefor And For Other Purposes"," 13th Congress, 3rd Regular Session ed. Philippines, 2007.

[9] V. Reyes, "The Impact of Automation on Elections," *Journal of Developing Societies,* vol. 29, no. 3, pp. 259-285, 2013, doi: 10.1177/0169796X13494276.

[10] Bagumbayan-Vnp Movement, Inc., And Richard J. Gordon, As Chairman Of Bagumbayan-Vnp Movement, Inc., Petitioners, Vs. Commission On Elections, Respondent., S. Court G.R. 222731, 2016.

[11] C. S. Monsod, "Beyond what the voters experienced," in *The Manila Times,* ed. Manila, 2010.

[12] C. S. Monsod, "Governance problems," in *The Manila Times,* ed. Manila, 2010.

[13] C. S. Monsod, "Where do we go from here?," in *The Manila Times,* ed. Manila, 2010.

[14] "Transcript of the April 8, 2019 meeting of the Technical Working Group (TWG) for the Joint Congressional Oversight Committee (JCOC) on the conduct of the elections," in *Joint Congressional Oversight Committee on the Automated Election System Technical Working Group,* 17th Congress, 3rd Regular Session ed, 2019.

[15] "Philippines: Smartmatic engineer in cybercrime case flees overseas," in *Asia News Monitor,* ed. Bangkok, 2016.

[16] "Comelec, Smartmatic officials face raps over 'alteration' in 2016 polls server," in *Philippines News Agency,* ed, 2017.

[17] "Number Of Registered Voters, Voters Who Actually Voted And Voters' Turnout By Province," COMELEC, Intramuros, Manila, Philippines, 2017. Accessed: October 16. [Online]. Available: https://comelec.gov.ph/?r=2016NLE/Statistics/VotersTurnout2016NLE

[18] "Philippines: Manual audit of clustered precincts matches machine counts --report," in *Asia News Monitor,* ed. Bangkok, 2016.

[19] V. S. III, "Privilege Speech of Senator Vicente C. Sotto III on March 6, 2018 regarding the 2016 National Elections," 2nd Session ed, 2018.

[20] V. S. III, "Privilege Speech of Senator Vicente C. Sotto III on March 14, 2018 regarding the 2016 National Elections," 2nd Session ed, 2018.

[21] M. Bueza, "'Not enough proof' yet in server logs to show 2016 poll fraud," ed, 2018.

[22] Senate, 18th Congress, 1st Regular Session Session. (2019). Providing For The Conduct Of Hybrid National, Local And Armm Elections, Through Manual Voting And Counting At The Precinct Level, And Automated Transmission And Canvassing, And For Other Purposes.

[23] "Marcos pushes for adoption of hybrid election system," in *Manila Bulletin*, ed. New Delhi, 2020.

[24] S. Risnanto, Y. B. A. Rahim, and N. S. Herman, "Preparatory Component for Adoption E-Voting," in 2019 IEEE 13th International Conference on Telecommunication Systems, Services, and Applications (TSSA), 3-4 Oct. 2019 2019, pp. 31-34, doi: 10.1109/TSSA48701.2019.8985461.

[25] "An Act Introducing Additional Reforms In The Electoral System And For Other Purposes," in *RA 6646*, 1st Congress, First Regular Session ed. Philippines, 1988.

[26] "An Act Introducing Additional Reforms In The Electoral System And For Other Purposes," 1st Congress, First Regular Session ed. Philippines, 1988.

[27] Guidelines in the Clustering / Grouping of Established Precincts, Including Accessible Precinct/s, in Connection with the May 9, 2016 National, Local and ARMM Regional Elections, COMELEC Resolution 10019, 2015.

[28] Project of Precincts for the 2016 National, Local and ARMM Elections [Online] Available: https://comelec.gov.ph/?r=2016NLE/ProjectofPrecincts/POP

[29] Amending certain provisions of resolution No. 10057 dated February 11, 2016 or otherwise known as the General Instructions for the Board of Election Inspectors (BEI) on the testing and sealing of Vote Counting Machines (VCMs), and voting, counting and transmission of election results in connection with the 09 May 2016 National and Local elections, COMELEC Resolution 10088, 2016.

[30] Guidelines On The Constitution, Composition And Appointment Of The Boards Of Canvassers, And Other Matters Related Thereto, In Connection With The May 09, 2016 National And Local Elections (NLE), COMELEC Resolution 10050, 2016.

[31] General Instructions for the Board of Canvassers on the Consolidation/Canvass and Transmission of Votes in Connection with the May 9, 2016 National and Local Elections, COMELEC Resolution 10083, 2016.

[32] S. Hussmann and P. W. Deng, "A high-speed optical mark reader hardware implementation at low cost using programmable logic," *Real-Time Imaging,* vol. 11, no. 1, pp. 19-30, 2005/02/01/ 2005, doi: https://doi.org/10.1016/j.rti.2005.03.001.

[33] S. G. Solutions, "Final Certification Test Results Comelec AES 2016 Voting System," April 14, 2016, 2016.

[34] General Instructions for the Board of Election Inspectors on the Testing and Sealing of Vote Counting Machines (VCMs), and Voting, Counting and Transmission of election Results in Connection with the 09 May 2016 National and Local Elections, COMELEC Resolution 10057, 2016.

[35] Revised Contingency Procedures in Connection with the May 9, 2016 National and Local Elections, COMELEC Resolution 10101, 2016.

[36] 2016 National and Local Elections Ballot Face Templates [Online] Available: https://comelec.gov.ph/?r=2016NLE/BallotFaceTemplates

[37] "An Act Providing For Synchronized National And Local Elections And For Electoral Reforms, Authorizing Appropriations Therefor, And For Other Purposes," 8th Congress ed. Philippines, 1991.

[38] "An Act Providing for Mandatory Biometrics Voter Registration," 15th Congress, Third Regular Session ed. Philippines: Official Gazette republic of the Philippines, 2013.

[39] R. G. A. Panaligan, Leslie Ann G., "SC upholds Comelec's 'No Bio, No Boto'," in *Manila Bulletin*, ed, 2015.

[40] S. Tomacruz, "Why Marcos wants SC to investigate election materials in 3 Mindanao provinces," October 15, 2019. [Online]. Available: https://www.rappler.com/nation/explainer-why-marcos-wants-sc-probe-election-materials-3-mindanao-provinces

[41] In The Matter Of The Composition Of The Random Manual Audit (Rma) Committee And Allocation Of Clustered Precincts Per Legislative District, COMELEC Resolution 10133, 2016.

[42] In The Matter Of The Amendments To The General Instructions For The Conduct Of Random Manual Audit (Rma) For The 9 May 2016 Automated Synchronized National And Local Elections And Subsequent Elections Thereafter, COMELEC Resolution 10090, 2016.

[43] In The Matter Of The General Instructions For The Conduct Of Random Manual Audit (Rma) For The 9 May 2016 Automated Synchronized National And Local Elections And Subsequent Elections Thereafter, COMELEC Resolution 10078, 2016.

[44] In The Matter Of Further Amending Provisions Of The General Instructions For The Conduct Of Random Manual Audit (Rma) For The 9 May 2016 Automated Synchronized National And Local Elections And Subsequent Elections Thereafter, COMELEC Resolution 10109, 2016.

[45] "An Act Providing For The Recognition And Use Of Electronic Commercial And Non-Commercial Transactions And Documents, Penalties For Unlawful Use Thereof, And For Other Purposes," 11th Congress ed. Philippines, 2000.

[46] A. Prosser, "Transparency in eVoting," *Transforming government,* vol. 8, no. 2, pp. 171-184, 2014, doi: 10.1108/TG-09-2013-0032.

[47] M. Volkamer, O. Spycher, and E. Dubuis, "Measures to establish trust in internet voting," presented at the Proceedings of the 5th International Conference on Theory and Practice of Electronic Governance, Tallinn, Estonia, 2011. [Online]. Available: https://doi.org/10.1145/2072069.2072071.

[48] O. Cetinkaya, "Analysis of Security Requirements for Cryptographic Voting Protocols (Extended Abstract)," in *2008 Third International Conference on Availability, Reliability and Security,* 4-7 March 2008 2008, pp. 1451-1456, doi: 10.1109/ARES.2008.167.

[49] V. Cortier, "Formal verification of e-voting: solutions and challenges," *ACM SIGLOG News,* vol. 2, no. 1, pp. 25–34, 2015, doi: 10.1145/2728816.2728823.

[50] List of Registered / Accredited Political Parties [Online] Available: https://comelec.gov.ph/?r=2016NLE/PoliticalParties

[51] US Election Assistance Commission. (2005). *2005 Voluntary Voting System guidelines.* [Online] Available: https://www.eac.gov/sites/default/files/eac_assets/1/28/VVSG.1.0_Volume_1.PDF

[52] "Transcript of the June 4, 2019 hearing of the Joint Congressional Oversight Committee on the Automated Election System," in *Joint Congressional Oversight Committee on the Automated Election System*, 17th Congress, 3rd Regular Session ed, 2019.

[53] US Election Assistance Commission. (2005). *2005 Voluntary Voting System Guideline.* [Online] Available: https://www.eac.gov/sites/default/files/eac_assets/1/28/VVSG1.0Vol.2.PDF

[54] A. Xenakis and A. Macintosh, "Procedural security analysis of electronic voting," 2004 2004, no. Conference Proceedings: ACM, 2004, pp. 541-546, doi: 10.1145/1052220.1052288. [Online]. Available: http://ulrls.summon.serialssolutions.com/2.0.0/link/0/eLvHCXMwbV3PS8MwFA4yQTypc-Jvctmx25pfbc51RRBhoicvI20SEbpVOuff70uarkOkvaSBR1_6yPua5H0fQpRMZtGfOUGY2JYS8oOWiZXaJIQbJRMpLC1I4Rk33p_ES8bzuVz03NG6Aqsrv6FfBsrqSWlX7eYpIIfUV_kCJnCYKHvuF1di6pCFK-TiKTQkAOPA79S1aeD5iRmfelvEa946my5Rlau9NJOfoE1XrPNRT5yulYNnEBVu-aAxbfapm2nlJTg6_ua2GmxH7AghB7EHd-8VNPb9GtP8U4_pQ3iPUzTqy_7wYpfQztCBWQ_RUXc0_hzFvlM7og68Cbp3WAVWE1xb3Gvq4J_anaceodd8_pY9RkFyIVKMkogWNqGAgQjMY0JoTrmZERMrpxqmJS3SeKaVVkoKZglRcSI4_G5ZrdMSsERBL9BgDf5fIkykspYxwhRcFPq0UFYUgiumXfX2FRrCGCy_WkqNZfD4-t-nN-i4PS9DI8Jv0eC72Zo7dLitmmpz7z_8L18nrQI

[55] N. Nissim, R. Yahalom, and Y. Elovici, "USB-based attacks," *Computers & Security,* vol. 70, pp. 675-688, 2017/09/01/ 2017, doi: https://doi.org/10.1016/j.cose.2017.08.002.

[56] A. Tetmeyer and H. Saiedian, "Security Threats and Mitigating Risk for USB Devices," *IEEE Technology and Society Magazine,* vol. 29, no. 4, pp. 44-49, 2010, doi: 10.1109/MTS.2010.939228.

[57] M. Turilli and L. Floridi, "The ethics of information transparency," (in English), *Ethics and Information Technology,* vol. 11, no. 2, pp. 105-112, Jun 2009 2020-11-17 2009, doi: http://dx.doi.org/10.1007/s10676-009-9187-9.

[58] R. Suwandi, S. M. Nasution, and F. Azmi, "Secure E-voting System by Utilizing Homomorphic Properties of the Encryption Algorithm," *Telkomnika,* Article vol. 16, no. 2, pp. 862-867, 2018, doi: 10.12928/TELKOMNIKA.v16i2.8420.

[59] US Election Assistance Commission. (2015). *2015 Voluntary Voting System Guidelines.*

[60] "Philippines: Comelec considering Internet voting for overseas Filipinos in 2016," in *Asia News Monitor,* ed. Bangkok, 2014.

[61] V. Mateu, J. Miret, and F. Sebé, "A hybrid approach to vector-based homomorphic tallying remote voting," *International Journal of Information Security,* Article vol. 15, no. 2, pp. 211-221, 2016, doi: 10.1007/s10207-015-0279-8.

[62] K. Peng, "An efficient shuffling based eVoting scheme," *The Journal of systems and software,* vol. 84, no. 6, pp. 906-922, 2011, doi: 10.1016/j.jss.2011.01.001.

[63] X. Yang, X. Yi, S. Nepal, A. Kelarev, and F. Han, "Blockchain voting: Publicly verifiable online voting protocol without trusted tallying authorities," *Future generation computer systems,* vol. 112, pp. 859-874, 2020, doi: 10.1016/j.future.2020.06.051.

[64] T. Dimitriou, "Efficient, Coercion-free and Universally Verifiable Blockchain-based Voting," *Computer networks (Amsterdam, Netherlands : 1999),* vol. 174, p. 107234, 2020, doi: 10.1016/j.comnet.2020.107234.

Appendix 1: Different Parts of the VCM

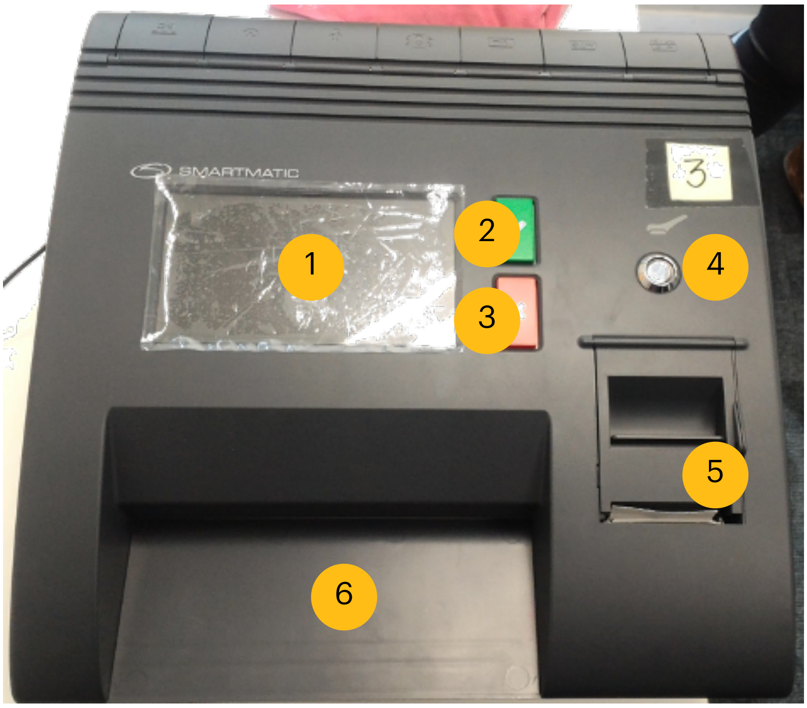


Figure A. Top View of the SAES 1800plus with back cover closed

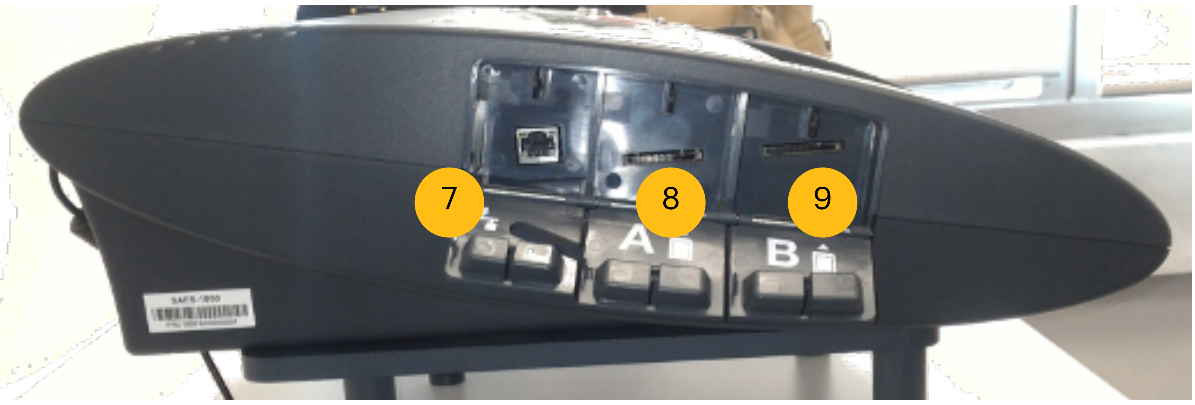


Figure B. Side view of the SAES 1800plus showing SD card slots

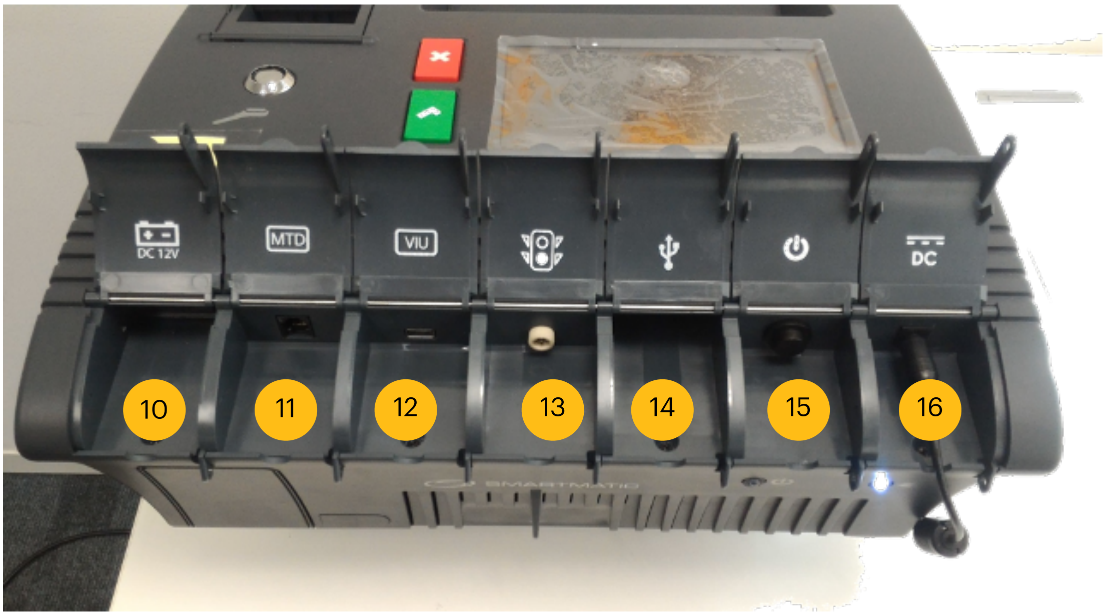


Figure C. Top view of the SAES 1800plus with back cover opened

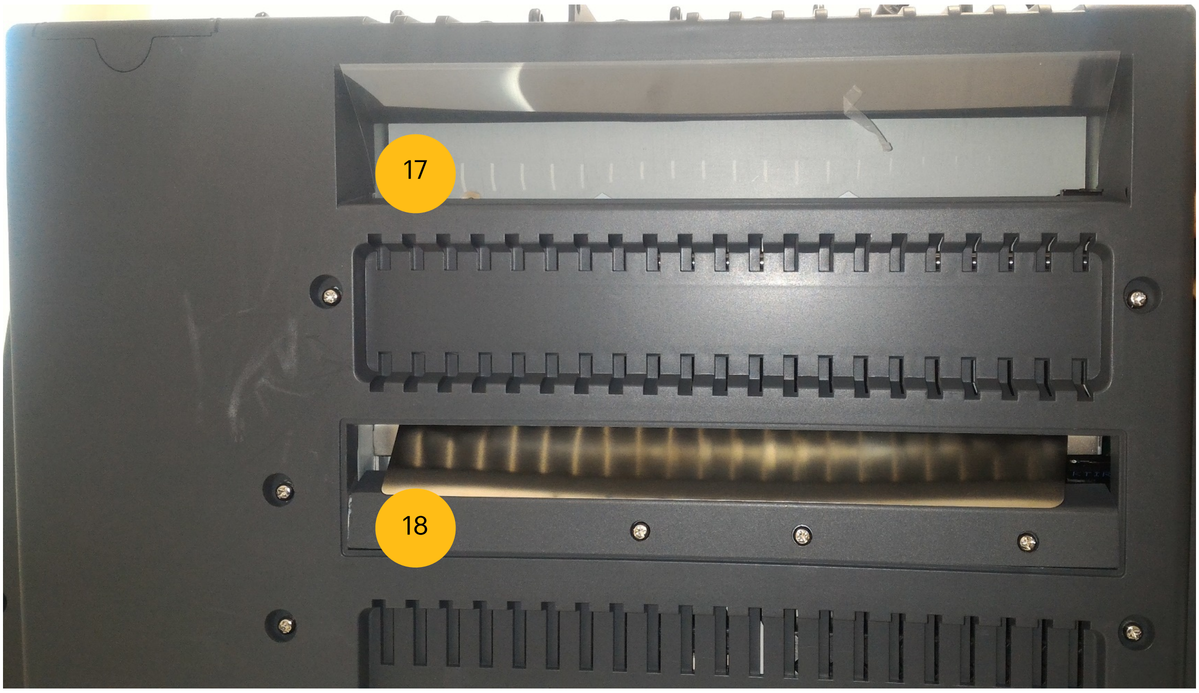


Figure 4. Bottom view of the SAES 1800plus

Table. Description of Parts of the SAES 1800plus

Ref	Part Name	Description
1	Touchscreen	A 16-bit TFT colored touch screen to capture inputs to the machine.
2	Accept button	A voter either uses the touchscreen or presses the “Accept” button to Accept a ballot positioned at the feeder,
3	Reject button	A voter can choose to “Reject” a ballot already fed into the machine if the screen shows that the votes recognized was not what the voter intended. using either the touchscreen or the reject button.
4	iButton receptacle	The receptacle is an electromagnetic reader that reads the iButton to authenticate the BEI.
5	Thermal paper slot	The VCM has a thermal paper printer to print the VVPAT and Election Returns (ER) The thermal paper is replaced or replenished by removing the cover, and placing the thermal paper roll in the compartment.
6	ballot feeder	Ballots are fed in this slot
7	Ethernet port	Used to connect ethernet cables forVSAT or BGAN satellite modems when USB modems are not working
8	Main SD Card Slot (Slot A)	SD card containing the configuration files of the VCM is inserted here. SD card A and B are synchronized and thus contains the same files.
9	WORM Card Slot (Slot B)	Same function as SD card slot A except that SD card slot B is capable of locking a Write Once Read Many (WORM Card).
10	battery cord plug	A standard vehicle battery (12V DC output) can run the VCM for 14 hours. That battery is plugged through this slot.
11	MTD Port	MTD or Multi Transmission Device Port is a serial bus used for the MTD device in the 2010 PCOS machines. The hardware bus pin map is confidential and proprietary. This port can only be used for the MTD and was never used in the 2016 election.
12	VIU Port	VIU or Voter Identification Units is a Smartmatic product for using fingerprints to identify voters. This was not used in the 2016 NLE.
13	Traffic lights	Standard LED light to show if the VCM is in use (red), or is idle (green).
14	USB port	The USB port is primarily used to insert a 3G device / dongle.
15	Power button	Physical power button switch in case the software “turn-off” does not work.
16	Accepted ballot slot	Accepted or legitimate ballots exit through this slot.
17	Rejected ballot slot	Rejected ballots exit through this slot. Ballots will only be rejected if the voter, or BEI pressed the “Reject” button (ref #3). Otherwise, a deficient ballot is always returned through the ballot feeder (ref #6).

Appendix 2: Python3 Code for Converting and Parsing VCM ‘*.pdf’ log files

```
#! /user/bin/env python3

# Script used to convert Election PDF log files to text, and parse it for easy processing
# To use this script, remember to change the file paths, and to download PyPDF2 via pip.
# The parsing logic used is specific to the Vote Counting Machine PDF log files in 2016.
# For Jeffrey Ian Dy MsC Information Security Project, University of London. 2020

import os, sys, re, PyPDF2, shutil
from tqdm import tqdm
from pathlib import Path
import pyinputplus as pyip

def remove_empty(_list):
#this function removes empty elements in  _list
  return list(filter(None, _list))

def removeDuplicateLines(_list):
    for _current, _element in enumerate(_list):
        _next = _current + 1
        if _next < len(_list) and _list[_current] == _list[_next]:
            _list.pop(_current)

def separate_linenum(_string):
#this function is useful for separating dates with time (for parsing time)
  _new_string = ''
  _list = _string.split()
  _match = re.match(r"([0-9]+)([a-zA-Z]+)", _list[0])
  if _match:
    _temp_array = _match.groups()
    _new_string = ' '.join((list(_temp_array) +_list[1:]))
  return _new_string

def introduction():
    print('''
        #################################################################################
        #                         WARNING !!!                          #
        # This Python script was intended for parsing Philippine Election 2016    #
        # PDF log files. It will convert all pdf log files to *.txt.            #
        # Then it will parse the file to allow for easy processing.             #
        # Author: jicd for MsC Infosec, University of London. 2020                     #
        #################################################################################
        ''')

    print(f'Your current directory is: {Path.cwd()}')
    print("Do you wish to continue? 'y' for yes or 'n' for no.")
    response = pyip.inputChoice(['yes', 'y', 'no', 'n'])
    if response.lower() == 'n' or response.lower() == 'no':
        print("exiting!")
        sys.exit()

def createDir():
# This function creates a folder at the current path where to store all files to be parsed.
# The intention is to put all parsed files using the same folder structure as the current folder.
# The function returns the _folder_path which is needed by other functions.

    print(f"Checking if folder 'converted_pdf' exists in {Path.cwd().parent}. ")
    _conversionPath = Path.cwd().parent / Path('converted_log_files')
    if _conversionPath.exists():
        print(f"Folder {_conversionPath} already exists.")
        print("Do you wish to clean the folder of its content?")
        response = pyip.inputChoice(['yes', 'y', 'no', 'n'])
        if response.lower() == 'y' or response.lower() == 'yes':
            shutil.rmtree(_conversionPath)
            os.makedirs(_conversionPath)
    else:
        print(f"Creating folder: {_conversionPath}")
        os.makedirs(_conversionPath)

    return _conversionPath

def garbageDeleter():
#This function deletes unwanted files if there is. These are present in the original files sent to the Senate.
#This happens when the files are moved to a removable device

    _p = Path.cwd()
    print(f"deleting '*stat*', '*DS_S*', '*MERGED*' files in current directory. Do you whish to continue?")
    response = pyip.inputChoice(['yes', 'y', 'no', 'n'])
    if response.lower() == 'y' or response.lower() == 'yes':
        _p = Path.cwd()
        for index, _element in enumerate(list(_p.rglob('*stat*'))):
            print(f"Removing {_element}")
            os.remove(_element)
        for index, _element in enumerate(list(_p.rglob('*DS_S*'))):
            print(f"Removing {_element}")
            os.remove(_element)
        for index, _element in enumerate(list(_p.rglob('*MERGED*'))):
            print(f"Removing {_element}")
            os.remove(_element)


def convertPDF(_convertPath, _writePath):
# this function simply converts all pages of a PDF file into one text file.
# I found out though that this is not simple, there is a need to reparse the files to make them more
# consistent for date analysis.
    _pdfFileobj = open(_convertPath, 'rb')
    _pdfReader = PyPDF2.PdfFileReader(_pdfFileobj)
    _pdf_pages = _pdfReader.numPages
    _increment = 0
    _content = []
    while _increment < _pdf_pages:
        _pageobj = _pdfReader.getPage(_increment)
        _extract_tuple = _pageobj.extractText().split('\n')
        for _l in _extract_tuple:
            _line = _l + '\n'
            with open(_writePath, 'a') as _f:
                _f.write(_line)
        _increment += 1


def parse1(_sourcePath, _filename):
#first part of parsing. clean up the File
```

```
    _destinationPath = Path(_sourcePath).parent / Path(_filename + '.parse1')
    _f = open(_sourcePath, 'r')
    _file = _f.read().split('\n')
    _f.close()
    _removed_spaces = remove_empty(_file)

    removeDuplicateLines(_removed_spaces) # remove all duplicate lines first.

    # basic parsing rules are shown below
    for _current, element in enumerate(_removed_spaces):
        _next = _current + 1
        if _removed_spaces[_current] == ':Report printed successfully.':
            _removed_spaces.insert(_next, '\\\\n')
        if _removed_spaces[_current] == 'information.':
            _removed_spaces.pop(_current)
        if _removed_spaces[_current] == ':The report could not be printed.':
            _removed_spaces.insert(_next, '<parser insert>')
        if _removed_spaces[_current].isdigit():
            _sequence = str(_removed_spaces[_current]) + str(_removed_spaces[_next])
            _removed_spaces[_current] = _sequence.strip()
            _removed_spaces.pop(_next)

    # after parsing, there may be duplicate lines produced. Removed them again.
    removeDuplicateLines(_removed_spaces)
    # initializes file. If it exists, previous contents are removed.
    _f = open(_destinationPath, 'w')
    _f.close()

    with open(_destinationPath, 'a') as _f:
        for _line in _removed_spaces:
            _f.write('%s\n' % _line)

    #janitorial services...
    os.remove(_sourcePath)
    return _destinationPath


def parse2(_sourcePath, _filename):
# As the name suggests, you need to first do parse1, before doing parse2.
    _tempfilePath = Path(_sourcePath).parent / Path(_filename + '.temp')
    _destinationPath = Path(_sourcePath).parent / Path(_filename + '.parsed')
    _counter = 0
    _count = 0
    _index = 0
    _file_array = []
    #initialitze the parsed file
    _f = open(_destinationPath, 'w')
    _f.close()
    _f = open(_sourcePath, 'r')
    _file = _f.read().split('\n')
    _f.close()

    #initialize the temp file. At this stage we now have two files
    _temp_f = open(_tempfilePath, 'w')
    _temp_f.close()

    # parse the log headers to make them look good (strip extra spaces)
    while _counter <=11:
        _file_array.append(_file[_counter].strip())
        _counter += 1

    with open(_tempfilePath, 'a') as _f_temp:
        for _element in _file_array:
            _f_temp.write('%s\n' % _element)
        _f_temp.write('\n')

    for _item in _file[12:]:
        _index = _file.index(_item)
        if _count < 2:
            _stringline =  _item.strip() + ' '
            _temp_file = open(_tempfilePath, 'a')
            _temp_file.write(_stringline)
            _temp_file.close()
            _count += 1
        else:
            _stringline = _item.strip() + '\n'
            _temp_file = open(_tempfilePath, 'a')
            _temp_file.write(_stringline)
            _temp_file.close()
            _count = 0

    #now, read the _file_array again, and separate the numbers from the dates
    _temp_file = open(_tempfilePath, 'r')
    _read_temp = list(_temp_file.read().split('\n'))
    _temp_file.close()
    _string = _read_temp[-2].strip()

    for _current, _element in enumerate(_read_temp):
        _next = _current + 13
        if _next < len(_read_temp)-1:
            _new_string = separate_linenum(_read_temp[_next])
            _read_temp[_next] = _new_string

    _read_temp.append(_string)
    with open(_destinationPath, 'a') as _parse_output:  #write them into our output file.
        for _element in _read_temp:
            _parse_output.write('%s\n' %_element)

    #janitorial services...
    os.remove(_sourcePath)
    os.remove(_tempfilePath)

########## start of main() ##############
introduction()
garbageDeleter()
_destination = createDir()
_total = len(list(Path.cwd().rglob('*.pdf')))
print(f"Parsing a total of {_total} PDF files")
#this is to show the progress bar. Looks more neat.
with tqdm(total = _total, desc = ' Progress') as pbar:
    for i in range(1):
        for _folders, _subfolders, _files in os.walk(Path.cwd()):
            for _file in _files:
                if _file.endswith('pdf'):
                    _convertPath = Path.cwd() / Path(_folders).name / Path(_file)
                    _writePath = Path(_destination) / Path(_folders).name / Path(_file + '.txt')

                    # this step replicates each folder name to the destination directory.
                    if not _writePath.parent.exists():
                        os.makedirs(Path(_writePath).parent)
                        #creates the file if it does not exist,
                        # or initializes the file if it exists.
                        _f = open(_writePath, 'w')
                        _f.close()
                    # Step 1: start PDF conversion
                    convertPDF(_convertPath, _writePath)
                    # Step 2: Let's parse the files stored in _writePath
                    _finalPath = parse1(_writePath, _file)
                    parse2(_finalPath, _file)
                    pbar.update(1)

########### end of main() ###############
```

Appendix 3: Python3 Code for VCM ‘*.log’ Analysis

```
#! /user/bin/env python3

# USED FOR SUMMARIZING *.LOG FILES OF THE VCM. ANOTHER SCRIPT SHOULD BE USED TO SUMMARIZE PDF FILES!

# Script used to summarize the *.log files of the Vote Counting machine used
# in 2016 election in the Philippines. This is necessary to enable quick analysis of the log files.
# For Jeffrey Ian Dy MsC Information Security Dissertation, University of London. 2020

import os, sys, time, shutil, datetime, re, calendar
from pathlib import Path
from tqdm import tqdm
import pyinputplus as pyip

# this function that harvests time per line
# unlike harvestTimePdf function, here remember to do _line.split('.')
def harvestTimeLog(_array):
    return datetime.datetime.strptime(_array[0], "%Y-%m-%d %H:%M:%S")

def showPercent(_a, _total):
    return round((_a/_total *100), 2)

# This function assumes that the MBOC CCS filename is the date of the log, and the time shown
# in the line corresponds to the timestamp for the log. IThe function extracts this information
def convertTimeMbocLog(_fileName, _line):
    _serverLogDateReg = re.compile(r'(\d{4})(\d{2})(\d{2})')
    _serverLodgDateReg2 = re.compile(r'(\d{4})-(\d{2})-(\d{2})')
    _timeRegex = re.compile(r'\d{2}:\d{2}:\d{2}')
    _time = _timeRegex.search(_line)
    if not _serverLogDateReg.search(_fileName):
        _date = _serverLodgDateReg2.search(_fileName)
    else: _date = _serverLogDateReg.search(_fileName)
    _month = calendar.month_name[int(_date.group(2).lstrip('0'))]
    _dateTime = _month + ' ' + _date.group(3).lstrip('0') + '/' + _date.group(1) + ' ' + _time.group()
    return datetime.datetime.strptime(_dateTime, "%B %d/%Y %H:%M:%S")

def intro():
    _conversionPath = Path.cwd()
    print('''

      ###########################################################################
                       WARNING !!!

        This Python script was intended for parsing Philippine
        Election 2016 text log files. Use ElectionPdfSummarizer.py
        for parsing PDF log files. The script must be ran from the same
        folder where *.log files are.

        Author: jicd for MsC Infosec, University of London. 2020
        ###########################################################################
        ''')

    print(f"Your current directory is: {Path.cwd()}")
    print("Do you wish to continue? 'y' for yes or 'n' for no.")
    response = pyip.inputChoice(['yes', 'y', 'no', 'n'])
    if response.lower() == 'n' or response.lower() == 'no':
        print("exiting!")
        sys.exit()
    else: return _conversionPath

# The createDir() function creates a folder at the current path
# from this folter, the parsed files (one file per log) will be dumped.
# Later, a "summary text file" will be created reporting all anomalies.
def createDir():

    print(f"Checking if folder 'PDF_summarizer_files' exists in {Path.cwd().parent}. ")
    _destinationPath = Path.cwd().parent / Path('Log_summarizer_files')
    if _destinationPath.exists():
        print(f"Folder {_destinationPath} already exists.")
        print("Do you wish to clean the folder of its content?")
        response = pyip.inputChoice(['yes', 'y', 'no', 'n'])
        if response.lower() == 'y' or response.lower() == 'yes':
            shutil.rmtree(_destinationPath)
            os.makedirs(_destinationPath)
    else:
        print(f"Creating folder: {_destinationPath}")
        os.makedirs(_destinationPath)
    return _destinationPath

def createHeaders(_sourceFile, _outputFile, ):
    _findPhraseList = [['Machine physical ID:'],
                        ['Voting Jurisdiction - Country:'],
                        ['Precinct ID:']]
    _lockDownTime = datetime.datetime.strptime('May 1/2016 00:00:00', "%b %d/%Y %H:%M:%S")
    _lineTime = datetime.datetime.strptime('May 1/2015 00:00:00', "%b %d/%Y %H:%M:%S")
    _machineId, _location, _precinct = '', '', ''

    with open(_sourceFile) as _source:
        for _line in _source:
            for _i, _element in enumerate(_findPhraseList):
                for _phrase in _element:
                    if _phrase in _line:
                        _array = _line.split(':')
                        _lineTime = harvestTimeLog(_array)
                        if _lineTime >= _lockDownTime:
                            if _i == 0 and _machineId == '':
                                _machineIdstr = _array[-2].split()
                                _machineId = _machineIdstr[-1]
                            elif _i == 1 and _location == '':
                                _array0 = _array[1].split()
                                _region = (f"{_array0[6]} {_array0[7]}")
                                _province = (f" {_array0[9]}")
                                _city = (f" {_array0[11]}")
                                _barangay = (f" {' '.join(_array0[13:])}")
                                _location = _region + _province + _city + _barangay
                            elif _i == 2 and _precinct == '':
                                _tempString = _array[-2].split()
                                _precinct = _tempString[-1]
    if _machineId == '' and _precinct == '': return False
    else: return True

def headers(_sourceFile, _outputFile):
    _file_array = []
    _counter = 0
    _string = Path(_sourceFile).stem
    _temp_string = (f'SUMMARY OF FINDINGS for {str.upper(_string)} VCM LOG FILE')
    _string = ('''
________________________________________________________________________
  '''  + '\n' + _temp_string + '\n') # we're just juggling through variable string ... :)
    _doc = open(_outputFile, 'w') #initializing the outputfile
    _doc.write(f'REPORT FILE GENERATED: {time.strftime("%c")}')
    _doc.write('''
      ###########################################################
      #       GENERATED BY ElectionPdfLogSummarizer.py        #
      #                                    #
      # this file contains the summary of important lines in the  #
      # VCM *.log audit logs. Analysis is made by the author     #
      # using relevant election CCS MBOC Logs.               #
      #                                    #
      # Author: Jeffrey Ian Dy. 2020.                      #
      # for MsC Information Security. University of London       #
      ###########################################################

        ''')
    _doc.write (_string.ljust(5))
    _doc.write('''
________________________________________________________________________
  ''' + '\n')
    _doc.close()
    _doc = open(_sourceFile, 'r')
    _readFile = _doc.read().split('\n')
    _doc.close()

    if not createHeaders(_sourceFile, _outputFile):
        with open(_output_file, 'a') as _output:
            _output.write('[ALERT!!!] THE AUDIT LOG FILE IS EMPTY! There are no entries in the log file.\n')
        return False
    else: return True

##### Below are Data Gathering Functions #####
## data gathering functions parse the log files and store data into lists
## and dictionaries

# This function gets the ballot statistics
# Notice that the function returns a dictionary
def ballotInfo(_sourceFile, _refTime):
    _lineTime = datetime.datetime.strptime('May 9/2016 5:30:00', "%b %d/%Y %H:%M:%S")
    _inserTime = datetime.datetime.strptime('May 9/2016 5:30:00', "%b %d/%Y %H:%M:%S")
    _lastBallotAccepted = datetime.datetime.strptime('May 1/2016 5:30:00', "%b %d/%Y %H:%M:%S")
    _ballotStats = [0,0,0,0,0]
    _timeStamps = [[],[]]
    _findPhrase = [['Vote cast completed'],
                    ['Ballot returned'],
                    ['Cleaning process started.'],
                    ['A ballot was inserted']
                ]
    with open(_sourceFile) as _source:
        for _line in _source:
            for _index, _element in enumerate(_findPhrase[:-1]):
                for _phrase in _element:
                    if _phrase in _line:
                        _array = _line.split('.')
                        _lineTime = harvestTimeLog(_array)
                        if _lineTime >= _refTime:
                            if _index == 0:
                                _arrayCast = _array[1].split()
                                _ballotStats[_index] = int(_arrayCast[-1])
                                _lastBallotAccepted = harvestTimeLog(_array)
                                if _ballotStats[0] == 1:
                                    _ballotStats[4] += 1
                                    _timeStamps[1].append(_lineTime)

                            else:
                                _ballotStats[_index] += 1

    with open(_sourceFile) as _source:
        for _line in _source:
            for _phrase in _findPhrase[3]:
                if _phrase in _line:
                    _array = _line.split('.')
                    _insertTime = harvestTimeLog(_array)
                    if _lastBallotAccepted < _insertTime:
                        _ballotStats[3] += 1
                        _timeStamps[0].append(_insertTime)

    return {'Total ballots casted': _ballotStats[0],
            'Total ballots returned': _ballotStats[1],
            'Total cleaning ballots inserted':_ballotStats[2],
            'Total ballots inserted after last counted':_ballotStats[3],
            'Total times election was (re-)opened' : _ballotStats[4]
            }, _timeStamps, _lastBallotAccepted

# This function checks the machineID of the machine and
# deduces if the machine was changed
def macIdChange(_sourceFile, _refTime):

    _findPhrase = ['Machine physical ID:']
    _count = 0
    _macChangeList, _macChangeTimeStamp, _macIDarray = [], [], []
    _macID = ''

    with open(_sourceFile, 'r') as _source:
        for _line in _source:
            for _phrase in _findPhrase:
                if _phrase in _line:
                    _array = _line.split('.')
                    _timeStamp = harvestTimeLog(_array)
                    if _timeStamp >= _refTime:
                        _macIDarray = _array[1].split()

                        # the next if statement states that if this is the
                        # first MacID list seen during election day, then this
                        # is the reference macID. If the macID changed, then the machine
                        # was replaced. We also count the number of times macID changed.
                        if len(_macChangeList) == 0:
                            _macID = _macIDarray[-1]
                            _macChangeList.append(_macID)
                        else:
                            _foundID = _macIDarray[-1]
                            if _macID != _foundID:
                                _macChangeTimeStamp.append(_timeStamp)
                                _macID = _foundID
                                _macChangeList.append(_macID)

    return _macChangeList, _macChangeTimeStamp

#The statCheck() function counts the states of the VCM
#states may be rezeroed or shutdown.
def statCheck(_sourceFile, _refTime):
    _findPhrase = [['shutting down...'],
                   ['Re-zero process will erase more than 13 votes.'],
                   ['Machine physical ID:']
                   ]
    _dictKeys = ['Total shutdown during election',
                 'Total Rezeros after voting started'
                 ]
    _statCount = [0,0]
    _stats = {}
    _shutdownTimestamps, _RezeroTimestamps = [], []

    with open(_sourceFile, 'r') as _source:
        for _line in _source:
            for _i, _element in enumerate(_findPhrase):
                for _phrase in _element:
                    if _phrase in _line:
                        _array = _line.split('.')
                        if harvestTimeLog(_array) >= _refTime:
                            if _i == 0:
                                _statCount[_i] += 1
                                _shutdownTimestamps.append(harvestTimeLog(_array))
                            elif _i == 1:
                                _statCount[_i] += 1
                                _RezeroTimestamps.append(harvestTimeLog(_array))

    for _index, _element in enumerate (_statCount):
        _stats[str(_dictKeys[_index])] = _element

    return _stats, _shutdownTimestamps, _RezeroTimestamps

# The transmissionInfo() function is complex. It first looks at the timestamps when the VCM successfully transmitted.
# It then harvests the VCM ID from the filename of the auditlog using regex, then finds if the MBOC CCS logs which is
# a different file and path,  contains a record of when the VCM ID transmitted file was received and processed.
# The function then compares if the time is within range and stores the timestamps.

def transmissionInfo(_sourceFile, _transParentPath, _refTime, _electionTime):
# The transmission logs shall be retained on the same path as they were transmitted to me:
    _findPhrase = ["The election results were sent successfully to 'MBOC:"]
    _lineTime = datetime.datetime.strptime('Jan 1/1900 5:30:00', "%b %d/%Y %H:%M:%S")
    _transmitTime = datetime.datetime.strptime('May 9/2016 5:30:00', "%b %d/%Y %H:%M:%S")

     # according to the MBOC general instructions, the MBOC opens at 3PM. that is 9 hours from the time the election
     # was authorized open
    _expectedTransmitTime = _electionTime + datetime.timedelta(hours = 9)
    _timeStamps, _transmitReceived, _transmitFileReceived = [], [], []
    _earlyTransmissionList, _sourceIP, _receivedIP  = [], [], []
    _reSendIP, _reSendTime, _earlyIP = [], [], []

    #we are looking for the VCM ID as a search parameter for the CCS logs.
```

```
_vcmIdregex = re.compile(r'(\d{4})(\d{4})')
_vcmId = _vcmIdregex.search(Path(_sourceFile).stem)
_transSearchPhrase = [["The information from results report [" + str(_vcmId.group()) + ".0"],
                      ["A tally was processed coming from device [" + str(_vcmId.group())]]

#the *DS_S* is a pain, and reappears when I move from one Mac to another (courtesy of iCloud)
#Thus, the best solution is to delete everytime the program is ran!
for index, _element in enumerate(list(_transParentPath.rglob('*DS_S*'))):
    print(f"Removing {_element}")
    os.remove(_element)

# first checks _source log file if a transmission is made by looking for p_findPhrase in every line of _source.
# If _source transmitted, it opens file _transmitFile and check if CCS received and processed the transmission.

with open(_sourceFile, 'r') as _source:
    for _line in _source:
        for _phrase in _findPhrase:
            if _phrase in _line:
                _array = _line.split('.')
                _findIpArray = _line.split()
                _lineTime = harvestTimeLog(_array)
                _sourceIPstrArray = _findIpArray[-1].split('/')
                _sourceIPstr = _sourceIPstrArray[0]
                if _lineTime >= _refTime and _lineTime not in _timeStamps: #avoiding duplicates
                    if _lineTime < _expectedTransmitTime:
                        _earlyTransmissionList.append(_lineTime)
                        _earlyIP.append(_sourceIPstr)
                    else:
                        _timeStamps.append(_lineTime)
                        _sourceIP.append(_sourceIPstr)
                    for _folders, _subfolders, _files in os.walk(_transParentPath):
                        for _subfolder in _subfolders:
                            _transmitFileSub = Path(_folders) / Path(_subfolder)
                        for _fileName in _files:
                            _transmitFile = Path(_transmitFileSub) / Path(_fileName)
                            with open(_transmitFile, 'r') as _f:
                                for _line in _f:
                                    for _index, _element in enumerate(_transSearchPhrase):
                                        for _phrase in _element:
                                            if _phrase in _line:
                                                # approximating time of transmission receipt by comparing
                                                #  MBOC log filename and line time.
                                                _transmitTime = convertTimeMbocLog(_fileName, _line)
                                                _receivedIpArray = _line.split()
                                                if _index == 0:
                                                    _receivedIPstr = _receivedIpArray[-4].strip('[]')
                                                    if _transmitTime >= _lineTime + datetime.timedelta(hours = -3) \
                                                    and _receivedIPstr not in _receivedIP \
                                                    and _receivedIPstr != '127.0.0.1':
                                                        _transmitReceived.append(_transmitTime)
                                                        _transmitFileReceived.append(_fileName)
                                                        _receivedIP.append(_receivedIPstr)
                                                elif _index == 1:
                                                    _receivedIPstr = _receivedIpArray[15].strip('[]')
                                                    if _transmitTime >= _lineTime + datetime.timedelta(hours = -3) \
                                                    and _receivedIPstr not in _reSendIP \
                                                    and _receivedIPstr != '127.0.0.1':
                                                        _reSendIP.append(_receivedIPstr)
```

```
                                                            _reSendTime.append(_transmitTime)

    return _timeStamps, _transmitReceived, _transmitFileReceived, _earlyTransmissionList, _sourceIP, _receivedIP, \
    _reSendIP, _reSendTime, _earlyIP

def CCSProcessedNotfromVCM (_fileArray, _transParentPath, _refTime):
    _timeStamp, _receivedIP, _vcmIdList = [], [], []
    for _folders, _subfolders, _files in os.walk(_transParentPath):
        for _subfolder in _subfolders:
            _transmitFileSub = Path(_folders) / Path(_subfolder)
        for _file in _files:
            _transmitFile = Path(_transmitFileSub) / Path (_file)
            for _element in _fileArray:
                _vcmIdArray = _element.split('-')
                _vcmId = _vcmIdArray[0]
                _transSearchPhrase = ["The information from results report [" + _vcmId + ".0"]
                with open(_transmitFile, 'r') as _f:
                    for _line in _f:
                        for _phrase in _transSearchPhrase:
                            if _phrase in _line:
                                _transmitTime = convertTimeMbocLog(_file, _line)
                                _receivedIpArray = _line.split()
                                _receivedIPstr = _receivedIpArray[-4].strip('[]')
                                if _receivedIPstr != '127.0.0.1' and _transmitTime >= _refTime:
                                    if len(_timeStamp) == 0:
                                        _timeStamp.append(_transmitTime)
                                        _receivedIP.append(_receivedIPstr)
                                        _vcmIdList.append(_vcmId)
                                    else:
                                        if _transmitTime not in _timeStamp:
                                            _timeStamp.append(_transmitTime)
                                            _receivedIP.append(_receivedIPstr)
                                            _vcmIdList.append(_vcmId)

    return _timeStamp, _receivedIP, _vcmIdList

##### end of fact-finding / statistics functions #####

########## start of main() program ##############
_countMacChange, _earlyTransmissionCount,  = 0, 0
_fileName = ''
_transmissionList, _earlyTransmissionTotal, _longTransmitGap, _tReceivedmismatch,  = [], [], [], []
_noTransmissionList, _ccsNotReceivedList, _noCcslogList, _IPMismatch, _reTransmit = [], [], [], [], []
_ballotStats, _isEmptyList, _noBallotOneList, _insertedAfterList, _electionReopenedList = [], [], [], [], []
_macIdChangeList, _earlyElectionList, _rezeroList, _shutdownList = [], [], [], []
_ballotSum, _statSum = {}, {}

_sourcePath = intro()
_destinationPath = createDir()

#as per Comelec Res 10088 amending 10057, the election day starts 6AM.
_electionTime = datetime.datetime.strptime('May 9/2016 06:00:00', "%b %d/%Y %H:%M:%S" )
# as per COMELEC Res 10057, FTS is May 2 -6, so no ballots should be fed after
_minTime = datetime.datetime.strptime('May 7/2016 00:00:00', "%b %d/%Y %H:%M:%S")
#time we do not expect any transmissions. If there are any, it can't be from PRELAT
_lockDownTime = datetime.datetime.strptime('May 1/2016 00:00:00', "%b %d/%Y %H:%M:%S")
```

```
## Hard coded Paths are here. The user is advised to change these Paths if s/he wants to try to run this script
_execReportPath = Path(_destinationPath)/Path('TxtLogsExecutiveSummary.txt')
_transParentPath = Path.cwd().parent / Path('MBOC-Master-Files')

_total = len(list(Path(_sourcePath).rglob('*.log')))
print(f"Summarizing a total of {_total} *.parsed files")
# tqdm is a python library for showing progress bars.
with tqdm(total = _total, desc = ' Progress') as pbar:
    for i in range(1):
        for _folders, _subfolders, _files in os.walk(_sourcePath):
            for _file in _files:
                if _file.endswith('log'):
                    _fileName = Path(_file).stem
                    _convertPath = Path(_sourcePath) / Path(_folders).name / Path(_file)
                    _writePath = Path(_destinationPath) / Path(_folders).name / Path(_fileName + '-analyzed.txt')

                    # this step replicates each folder name to the destination directory.
                    if not _writePath.parent.exists(): os.makedirs(Path(_writePath).parent)

                    # These functions are called in this order to write to the file.
                    _macIdChange = macIdChange(_convertPath, _electionTime)
                    _transmissionList = transmissionInfo(_convertPath, _transParentPath, _lockDownTime, _electionTime)

                    _ballotStats = ballotInfo(_convertPath, _minTime)
                    #adds to the total ballot statistics
                    for _key, _value in _ballotStats[0].items():
                        _ballotSum.setdefault(_key, 0)
                        _ballotSum[_key] += _value

                    _statCheck = statCheck(_convertPath, _electionTime)
                    # adds to total shutdown and rezero (statCheck) statistics
                    for _key, _value in _statCheck[0].items():
                        _statSum.setdefault(_key, 0)
                        _statSum[_key] += _value

                    _status = headers(_convertPath, _writePath)
                    if not _status: _isEmptyList.append(_fileName) #checks if the Log file is empty

                    with open(_writePath, 'a') as _f: #write them in a file.
                        #step: let's output the general ballot statistics
                        _f.write(f"\nTime stamp of last ballot counted is {_ballotStats[2]}\n")
                        for _key, _value in _ballotStats[0].items():
                            if _value:
                                _f.write(f"{_key} is {_value}\n")

                                #check for possible uncounted ballots
                                if "inserted after last counted" in _key:
                                    _insertedAfterList.append((
                                            f"{_ballotStats[0][_key]} ballot/s inserted "
                                            f"after the last was counted in {_fileName}"))
                                    for _time in _ballotStats[1][0]: _f.write(f"\t* {_time}\n")

                                #check if elections were repeated (which also requires a rezero)
                                elif "(re-)opened" in _key:
                                    for _time in _ballotStats[1][1]:
                                        _f.write(f"\t* {_time}\n")
                                        if _time < _electionTime:
                                            _earlyElectionList.append((
                                                f"Log {_fileName} indicate an election was opened early, on {_time}"))
                                    if len(_ballotStats[1][1]) > 1:
                                        _electionReopenedList.append((
                                            f"election was opened {str(len(_ballotStats[1][1]))} "
                                            f"times in {_fileName}"))

                        #Log integrity check: check for incomplete log entry
                        if _ballotStats[0]['Total times election was (re-)opened'] == 0:
                            _f.write("[Alert!!!] There is no registered first ballot entered. ")
                            _f.write("Log file lines are incomplete!\n")
                            _noBallotOneList.append(_fileName)

                        #print the values in macIdChange
                        if len(_macIdChange[1]) > 0:
                            _countMacChange += len(_macIdChange[1])
                            _f.write(f"\nThe Machine ID changed {len(_macIdChange[1])} time(s) during the election:\n")
                            _macIdChangeList.append((
                                f"The Machines were changed {len(_macIdChange[1])} times in {_fileName}"))

                            #careful, length of _macIdChange[1] is shorter by 1!
                            for _i, _element in enumerate(_macIdChange[0]):
                                _j = _i + 1
                                _f.write((
                                    f"\t* From {_element} to {_macIdChange[0][_j]} "
                                    f"on {_macIdChange[1][_i]}\n"))
                                if len(_macIdChange[1]) == _j:
                                    break
                        #print the statCheck function return values
                        for _key, _value in _statCheck[0].items():
                            if _value != 0:
                                _f.write(f"\n{_key} : {_value}\n")

                                #times a machine was shutdown during election period
                                if "shutdown" in _key:
                                    _shutdownList.append((
                                        f"The machine was shutdown {_statCheck[0][_key]} "
                                        f"times on election day for {_fileName}"))
                                    for _element in _statCheck[1]: _f.write(f"\t* {_element}\n")

                                #times when a rezero was made with actual voters being erased.
                                elif "Rezeros" in _key:
                                    _rezeroList.append((
                                        f"The machine was rezeroed {_statCheck[0][_key]} times "
                                        f"affecting actual ballots fed for {_fileName}"))
                                    for _element in _statCheck[2]: _f.write(f"\t* {_element}\n")

                    #process the transmission report
                        _f.write("\n**************** Results Transmission Analysis *********************\n")

                        # no transmission made by VCM nor processed by CCS
                        if  len(_transmissionList[0]) == 0 and len(_transmissionList[3]) == 0:
                            _noTransmissionList.append(_fileName)
                            _f.write(f"\n[Alert!!!] No transmission made by this machine\n")
                        else:
                            _f.write(f"\nVCM successfully transmitted on the following timestamps:\n")
                            if len(_transmissionList[3]) > 0:
                                _earlyTransmissionCount += len(_transmissionList[3])
                                for _i, _element in enumerate(_transmissionList[3]):
```

```
            _f.write((f"\t* from IP {_transmissionList[8][_i]} on {_element}"
                        f" [Alert!!!] Early Transmission. \n"))
            _earlyTransmissionTotal.append((
                f"{_element}: early transmission recorded in {_fileName}"))
    for _i, _element in enumerate(_transmissionList[0]):
        _f.write(f"\t* from IP {_transmissionList[4][_i]} on {_element}\n")

# VCM transmitted but not processed by CCS
if  len(_transmissionList[0]) > 0 and len(_transmissionList[1]) == 0:

    #removing all Angono VCMs from statistics because there is no MBOC log file submitted for Angono
    angonoRegex = re.compile(r'5801')
    if not angonoRegex.search(_fileName):
        _f.write(f"\n[Alert!!!] The CCS did not process the transmission sent by this VCM.\n")
        _ccsNotReceivedList.append(_fileName)
    else:
        _f.write(f"\n[Alert!!!] There is no MBOC CCS log file to match transmissions from this VCM \n")
        _noCcslogList.append(_fileName)
elif len(_transmissionList[0]) > 0 and len(_transmissionList[1]) > 0:
    _f.write(f"\nThe CCS Processed the transmissions successfully at these times:\n")
    for _i, _element in enumerate(_transmissionList[1]):
        _f.write((f"\t* from {_transmissionList[5][_i]} on {_element} found in"
                    f" {_transmissionList[2][_i]} log file.\n"))

        # mismtaching IP processing
        if _transmissionList[5][_i] not in _transmissionList[4]:
            _f.write(f"\t* IP mismatch: VCM IP is {_transmissionList[4][_i]} ")
            _f.write((
                    f"but CCS processed IP {_transmissionList[5][_i]} "
                    f"on {_transmissionList[1][_i]}\n"))
            _IPMismatch.append((
                f"IP mismatch in {_fileName}: VCM IP {_transmissionList[4][_i]} "
                f"but CCS processed {_transmissionList[5][_i]} "
                f"on {_transmissionList[1][_i]}"))

    _totalTransmit = len(_transmissionList[1]) + len(_transmissionList[7])
    if len(_transmissionList[0]) != _totalTransmit:
        _tReceivedmismatch.append((
            f"transmission count mismatch! {_fileName} transmitted {len(_transmissionList[0])} times,"
            f" but MBOC received {_totalTransmit} transmissions."
            ))
    for _i, _element in enumerate(_transmissionList[1]):
        if _element > _transmissionList[0][_i] + datetime.timedelta(minutes=30) or \
        _element < _transmissionList[0][_i]:
            _f.write(f"\t* Questionable time gap: VCM transmitted {_transmissionList[0][_i]} ")
            _f.write(f"but CCS processed on {_element}\n")
            _longTransmitGap.append((
                f"Questionable time gap: {_fileName} transmitted {_transmissionList[0][_i]} "
                f"but MBOC processed at {_element}"))

    # Retransmission happened but not processed (identical transmission)
    if len(_transmissionList[6]) >0:
        _f.write(f"\nidentical transmission happened:\n")
        for _i, _element in enumerate(_transmissionList[6]):
            _f.write(f"\t* from IP {_element} on {_transmissionList[7][_i]}\n")
            _reTransmit.append((
                f"Identical transmission in {_fileName} from IP {_element} "
                f"on {_transmissionList[7][_i]}"))
```

```
                    pbar.update(1)

### end of the summarizer (core) program.
## Let's now create the Executive Summary File ##

with tqdm( total = 1, desc = 'Creating Summary File') as pbar:
    with open(_execReportPath, 'w') as _f:
        _f.write(f"EXECTIVE REPORT GENERATED {time.strftime('%c')}\n")
        _f.write('''
      ##########################################
      #          EXECUTIVE SUMMARY         #
      #        Txt Log Files Analysis       #
      #                                    #
      #         Author: jicd. 2020.         #
      # for Msc InfoSec, University of London #
      ##########################################

         ''' + '\n')
        _f.write(f"Total log files analyzed: \t{_total}\n")

        _f.write(f"\n******** Log integrity checking ********\n")
        _f.write((
            f"\nTotal log files with empty entries:\t{len(_isEmptyList)}, "
            f"{showPercent(len(_isEmptyList), _total)}%\n"))

        for _element in _isEmptyList: _reportFile.write(f"\t* {_element}\n")
        _f.write((
            f"\nTotal log files without a first ballot entry:"
            f"\t{len(_noBallotOneList)}, {showPercent(len(_noBallotOneList), _total)}%\n"))
        for _element in _noBallotOneList: _f.write(f"\t* {_element}\n")
        _f.write(f"\n******** End of log integrity checking ********\n\n")

        for _key, _value in _ballotSum.items():
            _f.write(f"{_key}: {_value}\n")
            if "inserted after last counted" in _key:
                for _element in _insertedAfterList:
                    _f.write(f"\t* {_element}\n")
                _f.write("\n")
            elif "(re-)opened" in _key:
                _f.write((
                    f"Total clustered precincts that opened more than one election in the period:"
                    f"\t{len(_electionReopenedList)}, {showPercent(len(_electionReopenedList), _total)}%\n"))
                for _element in _electionReopenedList:
                    _f.write(f"\t* {_element}\n")

        _f.write((
            f"\nTotal Rezeros that affected actual votes:"
            f"\t{_statSum['Total Rezeros after voting started']}, "
            f"recorded in {len(_rezeroList)} log files.\n"))
        for _element in _rezeroList: _f.write(f"\t* {_element}\n")
        _f.write((
            f"\nTotal machine changes:\t{_countMacChange}, "
            f"recorded in {len(_macIdChangeList)} log files.\n"))
        for _element in _macIdChangeList: _f.write(f"\t* {_element}\n")

        if len(_earlyElectionList) > 0:
            _f.write((
                f"\nTotal instances when the first ballot accepted was earlier than official election opening time:"
                f"\t{len(_earlyElectionList)}, {showPercent(len(_earlyElectionList), _total)}%\n"))
            for _element in _earlyElectionList: _f.write(f"\t* {_element}\n")

        _f.write((
            f"\nTotal shutdowns during election:\t{_statSum['Total shutdown during election']}, "
            f"recorded in {len(_shutdownList)} log files.\n"))
        for _element in _shutdownList: _f.write(f"\t* {_element}\n")

        _f.write(f"\n******** Transmission Analysis Summary ********\n")
        _f.write((
            f"\nTotal early transmissions:\t{_earlyTransmissionCount}, "
            f"recorded in {len(_earlyTransmissionTotal)} log files, "
            f"{showPercent(len(_earlyTransmissionTotal), _total)}%.\n"))
        for _element in _earlyTransmissionTotal: _f.write(f"\t* {_element}\n")

        _f.write((
            f"\nTotal VCMs that transmitted but their CCS did not received transmission:"
            f"\t{len(_ccsNotReceivedList)}, "
            f"{showPercent(len(_ccsNotReceivedList), _total)}%\n"))
        for _element in _ccsNotReceivedList: _f.write(f"\t* {_element}\n")

        _f.write((
            f"\nTotal VCMs with mismatching transmit and received data:"
            f"\t{len(_tReceivedmismatch)}, {showPercent(len(_tReceivedmismatch), _total)}%\n"))
        for _element in _tReceivedmismatch: _f.write(f"\t* {_element}\n")

        _f.write((
            f"\nTotal VCMs with questionable transmit versus processed time gap:"
            f"\t{len(_longTransmitGap)}, {showPercent(len(_longTransmitGap), _total)}%\n"))
        for _element in _longTransmitGap: _f.write(f"\t* {_element}\n")

        _f.write((
            f"\nTotal VCMs with no matching MBOC CCS log file:"
            f"\t{len(_noCcslogList)}, {showPercent(len(_noCcslogList), _total)}%\n"))
        for _element in _noCcslogList: _f.write(f"\t* {_element}\n")

        _f.write((
            f"\nTotal VCMs that did not transmit results during election:"
            f"\t{len(_noTransmissionList)}, {showPercent(len(_noTransmissionList), _total)}%\n"))
        for _element in _noTransmissionList: _f.write(f"\t* {_element}\n")

        _f.write((
            f"\nIP Address mismatch between MBOC and processed CCS:"
            f"\t{len(_IPMismatch)}, {showPercent(len(_IPMismatch), _total)}%\n"))
        for _element in _IPMismatch: _f.write(f"\t* {_element}\n")

        _CCSProcessedElsewhere = CCSProcessedNotfromVCM(_noTransmissionList, _transParentPath, _electionTime)
        _f.write((
            f"\nTotal CCS Processed results from VCMs that did not transmit:"
            f"\t{len(_CCSProcessedElsewhere[2])}, "
            f"{showPercent(len(_CCSProcessedElsewhere[2]), len(_noTransmissionList))}%\n"))
        for _i, _element in enumerate(_CCSProcessedElsewhere[2]):
            _f.write((
                f"\t* {_element} did not transmit but CCS processed on "
                f"{_CCSProcessedElsewhere[0][_i]} from IP {_CCSProcessedElsewhere[1][_i]}\n"))

        _f.write((
            f"\nIdentical Transmission Received but not processed"
            f"\t{len(_reTransmit)}, {showPercent(len(_reTransmit), _total)}%\n"))
        for _i, _element in enumerate(_reTransmit):
            _f.write(f"\t*{_element}\n")

pbar.update(1)

## Printing on screen summary
print(f"After summarizing a total of {_total} log files, we got the following statistics.")
for _key, _value in _ballotSum.items():
    print(f"{_key}: {_value}")
print((
    f"Total times when an election was opened more than once: {len(_electionReopenedList)}, "
    f"{showPercent(len(_electionReopenedList), _total)}%"))
print((
    f"Total log files with empty entries: {len(_isEmptyList)}, "
    f"{showPercent(len(_isEmptyList), _total)}%"))
print((
    f"Total log files without a first ballot entry: {len(_noBallotOneList)}, "
    f"{showPercent(len(_noBallotOneList), _total)}%"))
print((
    f"Total machine changes: {_countMacChange}, "
    f"recorded in {len(_macIdChangeList)} log files."))
print((
    f"Total rezeros with affected counted ballots: {_statSum['Total Rezeros after voting started']} "
    f"recorded in {len(_rezeroList)} log files."))
print((
    f"Total shutdowns during election: {_statSum['Total shutdown during election']} "
    f" recorded in {len(_shutdownList)} log files."))
print((
    f"Total early transmissions: {_earlyTransmissionCount}, "
    f"recorded in {len(_earlyTransmissionTotal)} log files."))
print((
    f"Total VCMs that did not transmit results during election: {len(_noTransmissionList)}, "
    f"{showPercent(len(_noTransmissionList), _total)}%"))
print((
    f"Total VCMs that transmitted but their CCS did not received transmission: "
    f"{len(_ccsNotReceivedList)}, {showPercent(len(_ccsNotReceivedList), _total)}%"))
print()
print(f"For a more detailed summary of all the parsed log files, open {Path(_execReportPath)}")

########### end of main() ###############
```

Appendix 4: Summary Report Produced by the Log File Parser

```
EXECTIVE REPORT GENERATED Sat Jan 30 16:43:55 2021

        ##########################################
        #        EXECUTIVE SUMMARY        #
        #      Txt Log Files Analysis      #
        #                       #
        #       Author: jicd. 2020.       #
        # for Msc InfoSec, University of London #
        ##########################################


Total log files analyzed:     192

******** Log integrity checking ********

Total log files with empty entries:  0, 0.0%

Total log files without a first ballot entry:  0, 0.0%

******** End of log integrity checking ********

Total ballots casted: 90798
Total ballots returned: 5389
Total cleaning ballots inserted: 139
Total ballots inserted after last counted: 20
     * 2 ballot/s inserted after the last was counted in 05070003-audit
     * 1 ballot/s inserted after the last was counted in 05070026-audit
     * 1 ballot/s inserted after the last was counted in 36290017-audit
     * 9 ballot/s inserted after the last was counted in 36290022-audit
     * 1 ballot/s inserted after the last was counted in 36290002-audit
     * 3 ballot/s inserted after the last was counted in 36290015-audit
     * 1 ballot/s inserted after the last was counted in 36290006-audit
     * 2 ballot/s inserted after the last was counted in 36290021-audit

Total times election was (re-)opened: 224
Total clustered precincts that opened more than one election in the period: 27, 14.06%
     * election was opened 2 times in 05070010-audit
     * election was opened 3 times in 05070012-audit
     * election was opened 2 times in 36290024-audit
     * election was opened 3 times in 36290005-audit
     * election was opened 2 times in 36290011-audit
     * election was opened 2 times in 36290008-audit
     * election was opened 2 times in 36290003-audit
     * election was opened 2 times in 36290017-audit
     * election was opened 2 times in 36290022-audit
     * election was opened 2 times in 36290025-audit
     * election was opened 2 times in 36290010-audit
     * election was opened 2 times in 36290004-audit
     * election was opened 2 times in 36290009-audit
     * election was opened 3 times in 36290016-audit
     * election was opened 2 times in 36290002-audit
     * election was opened 2 times in 36290023-audit
     * election was opened 2 times in 36290007-audit
```

```
    * election was opened 2 times in 36290013-audit
    * election was opened 2 times in 36290020-audit
    * election was opened 2 times in 36290001-audit
    * election was opened 2 times in 36290015-audit
    * election was opened 2 times in 36290018-audit
    * election was opened 2 times in 36290012-audit
    * election was opened 2 times in 36290006-audit
    * election was opened 2 times in 36290021-audit
    * election was opened 3 times in 36290014-audit
    * election was opened 3 times in 36290019-audit

Total Rezeros that affected actual votes:   4, recorded in 4 log files.
    * The machine was rezeroed 1 times affecting actual ballots fed for 36290005-audit
    * The machine was rezeroed 1 times affecting actual ballots fed for 36290016-audit
    * The machine was rezeroed 1 times affecting actual ballots fed for 36290014-audit
    * The machine was rezeroed 1 times affecting actual ballots fed for 36290019-audit

Total machine changes:    7, recorded in 7 log files.
    * The Machines were changed 1 times in 05070046-audit
    * The Machines were changed 1 times in 05070026-audit
    * The Machines were changed 1 times in 05070013-audit
    * The Machines were changed 1 times in 58010024-audit
    * The Machines were changed 1 times in 58010091-audit
    * The Machines were changed 1 times in 58010012-audit
    * The Machines were changed 1 times in 36290019-audit

Total instances when the first ballot accepted was earlier than official election opening time:    31, 16.15%
    * Log 05070010-audit indicate an election was opened early, on 2016-05-08 09:36:48
    * Log 05070012-audit indicate an election was opened early, on 2016-05-08 08:26:06
    * Log 05070012-audit indicate an election was opened early, on 2016-05-08 09:00:26
    * Log 58010008-audit indicate an election was opened early, on 2016-05-09 04:44:56
    * Log 58010044-audit indicate an election was opened early, on 2016-05-09 05:59:38
    * Log 58010035-audit indicate an election was opened early, on 2016-05-09 05:57:17
    * Log 36290024-audit indicate an election was opened early, on 2016-05-09 01:47:17
    * Log 36290005-audit indicate an election was opened early, on 2016-05-09 01:54:35
    * Log 36290011-audit indicate an election was opened early, on 2016-05-09 01:52:08
    * Log 36290008-audit indicate an election was opened early, on 2016-05-09 01:50:58
    * Log 36290003-audit indicate an election was opened early, on 2016-05-09 01:51:07
    * Log 36290017-audit indicate an election was opened early, on 2016-05-09 01:52:30
    * Log 36290022-audit indicate an election was opened early, on 2016-05-09 01:52:16
    * Log 36290025-audit indicate an election was opened early, on 2016-05-09 01:44:27
    * Log 36290010-audit indicate an election was opened early, on 2016-05-09 02:05:56
    * Log 36290004-audit indicate an election was opened early, on 2016-05-09 01:47:44
    * Log 36290009-audit indicate an election was opened early, on 2016-05-09 01:54:12
    * Log 36290016-audit indicate an election was opened early, on 2016-05-09 01:50:11
    * Log 36290002-audit indicate an election was opened early, on 2016-05-09 01:51:20
    * Log 36290023-audit indicate an election was opened early, on 2016-05-09 01:48:44
    * Log 36290007-audit indicate an election was opened early, on 2016-05-09 01:58:29
    * Log 36290013-audit indicate an election was opened early, on 2016-05-09 01:55:23
    * Log 36290020-audit indicate an election was opened early, on 2016-05-09 01:49:07
    * Log 36290001-audit indicate an election was opened early, on 2016-05-09 01:52:27
    * Log 36290015-audit indicate an election was opened early, on 2016-05-09 01:46:55
    * Log 36290018-audit indicate an election was opened early, on 2016-05-09 01:57:58
    * Log 36290012-audit indicate an election was opened early, on 2016-05-09 01:54:48
    * Log 36290006-audit indicate an election was opened early, on 2016-05-09 01:55:14
    * Log 36290021-audit indicate an election was opened early, on 2016-05-09 02:21:31
    * Log 36290014-audit indicate an election was opened early, on 2016-05-09 01:44:27
    * Log 36290019-audit indicate an election was opened early, on 2016-05-09 01:57:46
```

```
Total shutdowns during election:  65, recorded in 37 log files.
    * The machine was shutdown 1 times on election day for 05070025-audit
    * The machine was shutdown 1 times on election day for 05070031-audit
    * The machine was shutdown 1 times on election day for 05070060-audit
    * The machine was shutdown 1 times on election day for 05070036-audit
    * The machine was shutdown 1 times on election day for 05070003-audit
    * The machine was shutdown 1 times on election day for 05070017-audit
    * The machine was shutdown 1 times on election day for 05070030-audit
    * The machine was shutdown 1 times on election day for 05070056-audit
    * The machine was shutdown 1 times on election day for 05070044-audit
    * The machine was shutdown 1 times on election day for 05070062-audit
    * The machine was shutdown 2 times on election day for 05070057-audit
    * The machine was shutdown 1 times on election day for 05070015-audit
    * The machine was shutdown 1 times on election day for 05070026-audit
    * The machine was shutdown 2 times on election day for 58010024-audit
    * The machine was shutdown 3 times on election day for 58010091-audit
    * The machine was shutdown 1 times on election day for 58010079-audit
    * The machine was shutdown 1 times on election day for 58010054-audit
    * The machine was shutdown 3 times on election day for 58010012-audit
    * The machine was shutdown 2 times on election day for 36290024-audit
    * The machine was shutdown 4 times on election day for 36290005-audit
    * The machine was shutdown 1 times on election day for 36290011-audit
    * The machine was shutdown 1 times on election day for 36290008-audit
    * The machine was shutdown 1 times on election day for 36290003-audit
    * The machine was shutdown 2 times on election day for 36290017-audit
    * The machine was shutdown 1 times on election day for 36290022-audit
    * The machine was shutdown 1 times on election day for 36290010-audit
    * The machine was shutdown 5 times on election day for 36290004-audit
    * The machine was shutdown 1 times on election day for 36290009-audit
    * The machine was shutdown 4 times on election day for 36290016-audit
    * The machine was shutdown 1 times on election day for 36290002-audit
    * The machine was shutdown 2 times on election day for 36290023-audit
    * The machine was shutdown 2 times on election day for 36290007-audit
    * The machine was shutdown 1 times on election day for 36290020-audit
    * The machine was shutdown 1 times on election day for 36290001-audit
    * The machine was shutdown 2 times on election day for 36290021-audit
    * The machine was shutdown 4 times on election day for 36290014-audit
    * The machine was shutdown 5 times on election day for 36290019-audit

******** Transmission Analysis Summary ********

Total early transmissions:  16, recorded in 16 log files, 8.33%.
    * 2016-05-06 11:13:31: early transmission recorded in 05070038-audit
    * 2016-05-06 12:19:07: early transmission recorded in 05070035-audit
    * 2016-05-06 13:49:13: early transmission recorded in 05070045-audit
    * 2016-05-03 11:00:29: early transmission recorded in 58010011-audit
    * 2016-05-03 14:58:55: early transmission recorded in 58010047-audit
    * 2016-05-03 10:32:24: early transmission recorded in 58010017-audit
    * 2016-05-03 14:47:13: early transmission recorded in 58010048-audit
    * 2016-05-03 11:16:57: early transmission recorded in 58010013-audit
    * 2016-05-03 10:24:23: early transmission recorded in 58010015-audit
    * 2016-05-03 14:44:08: early transmission recorded in 58010034-audit
    * 2016-05-03 10:58:33: early transmission recorded in 58010018-audit
    * 2016-05-03 14:40:46: early transmission recorded in 58010039-audit
    * 2016-05-03 14:53:53: early transmission recorded in 58010049-audit
    * 2016-05-03 10:59:59: early transmission recorded in 58010012-audit
    * 2016-05-03 11:51:59: early transmission recorded in 58010014-audit
    * 2016-05-03 10:56:25: early transmission recorded in 58010019-audit

Total VCMs that transmitted but their CCS did not received transmission:     19, 9.9%
    * 36290024-audit
    * 36290005-audit
    * 36290011-audit
    * 36290008-audit
    * 36290003-audit
    * 36290022-audit
    * 36290025-audit
    * 36290010-audit
    * 36290004-audit
    * 36290016-audit
    * 36290002-audit
    * 36290023-audit
    * 36290007-audit
    * 36290001-audit
    * 36290018-audit
    * 36290006-audit
    * 36290021-audit
    * 36290014-audit
    * 36290019-audit

Total VCMs with mismatching transmit and received data:     53, 27.6%
    * transmission count mismatch! 05070054-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070040-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070061-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070023-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070037-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070016-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070002-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070067-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070009-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070073-audit transmitted 2 times, but MBOC received 3 transmissions.
    * transmission count mismatch! 05070028-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070052-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070046-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070010-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070004-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070041-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070055-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070074-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070060-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070058-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070022-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070008-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070072-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070066-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070047-audit transmitted 2 times, but MBOC received 3 transmissions.
    * transmission count mismatch! 05070029-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070053-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070005-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070030-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070024-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070019-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070038-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070042-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070014-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070021-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070050-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070065-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070071-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070027-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070049-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070006-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070039-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070043-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070057-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070001-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070015-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070034-audit transmitted 1 times, but MBOC received 3 transmissions.
    * transmission count mismatch! 05070020-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070045-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070051-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070070-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070026-audit transmitted 1 times, but MBOC received 2 transmissions.
    * transmission count mismatch! 05070069-audit transmitted 1 times, but MBOC received 2 transmissions.

Total VCMs with questionable transmit versus processed time gap:    24, 12.5%
    * Questionable time gap: 05070054-audit transmitted 2016-05-09 18:07:15 but MBOC processed at 2016-05-09 18:48:28
    * Questionable time gap: 05070067-audit transmitted 2016-05-09 18:50:15 but MBOC processed at 2016-05-09 18:50:11
    * Questionable time gap: 05070073-audit transmitted 2016-05-09 17:12:39 but MBOC processed at 2016-05-09 18:17:18
    * Questionable time gap: 05070010-audit transmitted 2016-05-09 18:58:15 but MBOC processed at 2016-05-09 18:54:27
    * Questionable time gap: 05070074-audit transmitted 2016-05-09 18:04:23 but MBOC processed at 2016-05-09 19:50:03
    * Questionable time gap: 05070058-audit transmitted 2016-05-09 17:34:20 but MBOC processed at 2016-05-09 19:09:56
    * Questionable time gap: 05070022-audit transmitted 2016-05-09 18:45:57 but MBOC processed at 2016-05-09 18:45:47
    * Questionable time gap: 05070019-audit transmitted 2016-05-09 17:35:15 but MBOC processed at 2016-05-09 20:00:11
    * Questionable time gap: 05070038-audit transmitted 2016-05-09 21:38:24 but MBOC processed at 2016-05-09 21:38:03
    * Questionable time gap: 05070042-audit transmitted 2016-05-09 18:51:28 but MBOC processed at 2016-05-09 18:46:13
    * Questionable time gap: 05070014-audit transmitted 2016-05-09 17:49:40 but MBOC processed at 2016-05-09 18:39:51
    * Questionable time gap: 05070071-audit transmitted 2016-05-09 18:30:00 but MBOC processed at 2016-05-09 18:29:46
    * Questionable time gap: 05070027-audit transmitted 2016-05-09 17:34:41 but MBOC processed at 2016-05-09 18:53:54
    * Questionable time gap: 05070049-audit transmitted 2016-05-09 22:04:24 but MBOC processed at 2016-05-09 22:02:44
    * Questionable time gap: 05070068-audit transmitted 2016-05-09 17:35:04 but MBOC processed at 2016-05-09 18:51:06
    * Questionable time gap: 05070043-audit transmitted 2016-05-09 18:55:34 but MBOC processed at 2016-05-09 18:55:21
    * Questionable time gap: 05070001-audit transmitted 2016-05-09 20:56:18 but MBOC processed at 2016-05-09 20:49:34
    * Questionable time gap: 05070015-audit transmitted 2016-05-09 18:30:26 but MBOC processed at 2016-05-09 18:30:24
    * Questionable time gap: 05070034-audit transmitted 2016-05-09 18:30:45 but MBOC processed at 2016-05-09 17:57:27
    * Questionable time gap: 05070045-audit transmitted 2016-05-09 17:49:07 but MBOC processed at 2016-05-09 18:57:23
    * Questionable time gap: 05070070-audit transmitted 2016-05-09 17:35:02 but MBOC processed at 2016-05-09 19:07:20
    * Questionable time gap: 36290009-audit transmitted 2016-05-10 08:23:33 but MBOC processed at 2016-05-10 07:53:48
    * Questionable time gap: 36290020-audit transmitted 2016-05-10 07:23:19 but MBOC processed at 2016-05-10 07:55:04
    * Questionable time gap: 36290015-audit transmitted 2016-05-10 07:20:41 but MBOC processed at 2016-05-10 07:53:35

Total VCMs with no matching MBOC CCS log file:   27, 14.06%
    * 58010011-audit
    * 58010083-audit
    * 58010008-audit
    * 58010091-audit
    * 58010017-audit
    * 58010055-audit
    * 58010074-audit
    * 58010082-audit
    * 58010073-audit
    * 58010046-audit
    * 58010037-audit
    * 58010048-audit
    * 58010081-audit
    * 58010015-audit
    * 58010034-audit
    * 58010087-audit
    * 58010076-audit
    * 58010018-audit
    * 58010057-audit
    * 58010080-audit
    * 58010012-audit
    * 58010071-audit
    * 58010092-audit
    * 58010035-audit
    * 58010019-audit
    * 58010056-audit
    * 58010042-audit

Total VCMs that did not transmit results during election:  82, 42.71%
    * 05070059-audit
    * 05070025-audit
    * 05070031-audit
    * 05070036-audit
    * 05070003-audit
    * 05070017-audit
    * 05070011-audit
    * 05070063-audit
    * 05070056-audit
    * 05070044-audit
    * 05070033-audit
    * 05070012-audit
    * 05070018-audit
    * 05070062-audit
    * 05070064-audit
    * 05070032-audit
    * 05070007-audit
    * 05070013-audit
    * 58010005-audit
    * 58010030-audit
    * 58010024-audit
    * 58010072-audit
    * 58010066-audit
    * 58010029-audit
    * 58010053-audit
    * 58010036-audit
    * 58010085-audit
    * 58010058-audit
    * 58010022-audit
    * 58010079-audit
    * 58010003-audit
    * 58010088-audit
    * 58010041-audit
    * 58010060-audit
    * 58010010-audit
    * 58010004-audit
    * 58010025-audit
    * 58010031-audit
    * 58010067-audit
    * 58010009-audit
    * 58010028-audit
    * 58010052-audit
    * 58010059-audit
    * 58010090-audit
    * 58010023-audit
    * 58010084-audit
    * 58010016-audit
    * 58010078-audit
    * 58010002-audit
    * 58010054-audit
    * 58010089-audit
    * 58010040-audit
    * 58010061-audit
    * 58010075-audit
    * 58010032-audit
    * 58010026-audit
    * 58010007-audit
    * 58010069-audit
    * 58010045-audit
    * 58010051-audit
    * 58010070-audit
    * 58010064-audit
    * 58010001-audit
    * 58010093-audit
    * 58010020-audit
    * 58010062-audit
    * 58010043-audit
    * 58010027-audit
    * 58010033-audit
    * 58010068-audit
    * 58010006-audit
    * 58010050-audit
    * 58010044-audit
    * 58010065-audit
    * 58010021-audit
    * 58010086-audit
    * 58010063-audit
    * 58010077-audit
    * 58010038-audit
    * 36290017-audit
    * 36290013-audit
    * 36290012-audit

IP Address mismatch between MBOC and processed CCS:      22, 11.46%
    * IP mismatch in 05070054-audit: VCM IP 10.11.129.176 but CCS processed 10.101.1.198 on 2016-05-09 18:48:28
    * IP mismatch in 05070002-audit: VCM IP 10.11.98.105 but CCS processed 10.101.1.198 on 2016-05-09 17:50:23
    * IP mismatch in 05070073-audit: VCM IP 10.11.20.187 but CCS processed 10.101.1.198 on 2016-05-09 18:17:18
    * IP mismatch in 05070046-audit: VCM IP 10.12.111.150 but CCS processed 10.101.1.198 on 2016-05-09 22:38:43
    * IP mismatch in 05070041-audit: VCM IP 10.12.61.116 but CCS processed 10.101.1.198 on 2016-05-09 17:47:34
    * IP mismatch in 05070074-audit: VCM IP 10.12.94.46 but CCS processed 10.101.1.198 on 2016-05-09 19:50:03
    * IP mismatch in 05070058-audit: VCM IP 10.12.24.144 but CCS processed 10.101.1.198 on 2016-05-09 19:09:56
    * IP mismatch in 05070053-audit: VCM IP 10.11.71.142 but CCS processed 10.101.1.198 on 2016-05-09 17:49:51
    * IP mismatch in 05070024-audit: VCM IP 10.19.15.59 but CCS processed 10.101.1.198 on 2016-05-09 17:49:39
    * IP mismatch in 05070019-audit: VCM IP 10.12.60.201 but CCS processed 10.101.1.198 on 2016-05-09 20:00:11
    * IP mismatch in 05070014-audit: VCM IP 10.12.83.240 but CCS processed 10.101.1.198 on 2016-05-09 18:39:51
    * IP mismatch in 05070021-audit: VCM IP 10.12.114.185 but CCS processed 10.101.1.198 on 2016-05-09 23:02:11
    * IP mismatch in 05070027-audit: VCM IP 10.19.51.104 but CCS processed 10.101.1.198 on 2016-05-09 18:53:54
    * IP mismatch in 05070049-audit: VCM IP 10.12.105.34 but CCS processed 10.101.1.198 on 2016-05-09 22:02:44
```

```
     * IP mismatch in 05070068-audit: VCM IP 10.12.14.245 but CCS processed 10.101.1.198 on 2016-05-09 18:51:06
     * IP mismatch in 05070034-audit: VCM IP 10.11.70.177 but CCS processed 10.101.1.198 on 2016-05-09 17:57:27
     * IP mismatch in 05070045-audit: VCM IP 10.11.99.47 but CCS processed 10.101.1.198 on 2016-05-09 18:57:23
     * IP mismatch in 05070070-audit: VCM IP 10.11.97.102 but CCS processed 10.101.1.198 on 2016-05-09 19:07:20
     * IP mismatch in 05070026-audit: VCM IP 10.19.70.77 but CCS processed 10.101.1.198 on 2016-05-09 23:17:38
     * IP mismatch in 36290009-audit: VCM IP 10.12.123.216 but CCS processed 10.101.1.198 on 2016-05-10 07:53:48
     * IP mismatch in 36290020-audit: VCM IP 10.12.123.72 but CCS processed 10.101.1.198 on 2016-05-10 07:55:04
     * IP mismatch in 36290015-audit: VCM IP 10.11.4.194 but CCS processed 10.101.1.198 on 2016-05-10 07:53:35

Total CCS Processed results from VCMs that did not transmit:2, 2.44%
     * 05070012 did not transmit but CCS processed on 2016-05-09 23:34:12 from IP 10.101.1.198
     * 05070013 did not transmit but CCS processed on 2016-05-09 22:14:48 from IP 10.101.1.198

Identical Transmission Received but not processed    59, 30.73%
     *Identical transmission in 05070054-audit from IP 10.101.1.198 on 2016-05-10 06:14:33
     *Identical transmission in 05070040-audit from IP 10.101.1.198 on 2016-05-10 09:44:43
     *Identical transmission in 05070061-audit from IP 10.101.1.198 on 2016-05-10 06:08:15
     *Identical transmission in 05070023-audit from IP 10.101.1.198 on 2016-05-10 06:47:04
     *Identical transmission in 05070037-audit from IP 10.101.1.198 on 2016-05-10 09:18:31
     *Identical transmission in 05070016-audit from IP 10.101.1.198 on 2016-05-10 06:45:47
     *Identical transmission in 05070002-audit from IP 10.101.1.198 on 2016-05-10 06:04:00
     *Identical transmission in 05070067-audit from IP 10.101.1.198 on 2016-05-10 06:01:55
     *Identical transmission in 05070009-audit from IP 10.101.1.198 on 2016-05-10 06:35:37
     *Identical transmission in 05070073-audit from IP 10.101.1.198 on 2016-05-10 07:40:50
     *Identical transmission in 05070073-audit from IP 10.11.81.243 on 2016-05-09 18:28:49
     *Identical transmission in 05070028-audit from IP 10.101.1.198 on 2016-05-10 06:44:50
     *Identical transmission in 05070052-audit from IP 10.101.1.198 on 2016-05-10 06:09:14
     *Identical transmission in 05070046-audit from IP 10.101.1.198 on 2016-05-10 06:23:12
     *Identical transmission in 05070010-audit from IP 10.101.1.198 on 2016-05-10 06:43:36
     *Identical transmission in 05070004-audit from IP 10.101.1.198 on 2016-05-10 10:11:14
     *Identical transmission in 05070041-audit from IP 10.101.1.198 on 2016-05-10 09:46:55
     *Identical transmission in 05070055-audit from IP 10.101.1.198 on 2016-05-10 06:20:35
     *Identical transmission in 05070074-audit from IP 10.101.1.198 on 2016-05-10 06:13:37
     *Identical transmission in 05070060-audit from IP 10.101.1.198 on 2016-05-10 06:16:14
     *Identical transmission in 05070058-audit from IP 10.101.1.198 on 2016-05-10 06:10:06
     *Identical transmission in 05070022-audit from IP 10.101.1.198 on 2016-05-10 06:29:54
     *Identical transmission in 05070008-audit from IP 10.101.1.198 on 2016-05-10 06:34:34
     *Identical transmission in 05070072-audit from IP 10.101.1.198 on 2016-05-10 06:03:14
     *Identical transmission in 05070066-audit from IP 10.101.1.198 on 2016-05-10 06:17:42
     *Identical transmission in 05070047-audit from IP 10.101.1.198 on 2016-05-10 08:15:21
     *Identical transmission in 05070047-audit from IP 10.19.14.223 on 2016-05-09 19:25:12
     *Identical transmission in 05070029-audit from IP 10.101.1.198 on 2016-05-10 06:57:39
     *Identical transmission in 05070053-audit from IP 10.101.1.198 on 2016-05-10 06:02:29
     *Identical transmission in 05070005-audit from IP 10.101.1.198 on 2016-05-10 10:10:38
     *Identical transmission in 05070030-audit from IP 10.101.1.198 on 2016-05-10 06:38:33
     *Identical transmission in 05070024-audit from IP 10.101.1.198 on 2016-05-10 06:48:54
     *Identical transmission in 05070019-audit from IP 10.101.1.198 on 2016-05-10 06:27:06
     *Identical transmission in 05070038-audit from IP 10.101.1.198 on 2016-05-10 09:45:30
     *Identical transmission in 05070042-audit from IP 10.101.1.198 on 2016-05-10 08:43:03
     *Identical transmission in 05070014-audit from IP 10.101.1.198 on 2016-05-10 06:31:49
     *Identical transmission in 05070021-audit from IP 10.101.1.198 on 2016-05-10 06:28:04
     *Identical transmission in 05070050-audit from IP 10.101.1.198 on 2016-05-10 06:12:43
     *Identical transmission in 05070065-audit from IP 10.101.1.198 on 2016-05-10 06:18:50
     *Identical transmission in 05070071-audit from IP 10.101.1.198 on 2016-05-10 06:00:45
     *Identical transmission in 05070027-audit from IP 10.101.1.198 on 2016-05-10 08:22:51
     *Identical transmission in 05070049-audit from IP 10.101.1.198 on 2016-05-10 06:21:18
     *Identical transmission in 05070068-audit from IP 10.101.1.198 on 2016-05-10 06:19:50
     *Identical transmission in 05070006-audit from IP 10.101.1.198 on 2016-05-10 06:37:40
```

```
*Identical transmission in 05070039-audit from IP 10.101.1.198 on 2016-05-10 09:47:52
*Identical transmission in 05070043-audit from IP 10.101.1.198 on 2016-05-10 08:43:42
*Identical transmission in 05070057-audit from IP 10.101.1.198 on 2016-05-10 05:59:23
*Identical transmission in 05070001-audit from IP 10.101.1.198 on 2016-05-10 06:16:59
*Identical transmission in 05070015-audit from IP 10.101.1.198 on 2016-05-10 06:30:55
*Identical transmission in 05070034-audit from IP 10.101.1.198 on 2016-05-10 06:32:36
*Identical transmission in 05070034-audit from IP 10.11.70.177 on 2016-05-09 18:31:05
*Identical transmission in 05070020-audit from IP 10.101.1.198 on 2016-05-10 06:33:33
*Identical transmission in 05070045-audit from IP 10.101.1.198 on 2016-05-10 06:04:54
*Identical transmission in 05070051-audit from IP 10.101.1.198 on 2016-05-10 06:10:57
*Identical transmission in 05070070-audit from IP 10.101.1.198 on 2016-05-10 06:15:21
*Identical transmission in 05070048-audit from IP 10.101.1.198 on 2016-05-10 06:22:25
*Identical transmission in 05070026-audit from IP 10.101.1.198 on 2016-05-10 06:26:14
*Identical transmission in 05070069-audit from IP 10.101.1.198 on 2016-05-10 08:16:13
*Identical transmission in 36290009-audit from IP 10.101.1.198 on 2016-05-10 08:16:00
```