EXHIBIT A

2022 Election Cycle / Voting Equipment*

County	System Type	Manufacturer	Maintenance	Model	Firmware Type	Software Type
Apache	Central Count - Digital Scan	ES&S	ES&S	DS450	3.1.1.0	EVS 6.0.4.0
	Accessible Voting Device			ExpressVote (BMD)	1.5.2.1	EVS 6.0.4.0
	Central Count - Digital Scan			DS850	3.1.1.0	EVS 6.0.4.0
Cochise	Digital Scan	ES&S	ES&S	DS200	2.17.4.0	ElectionWare 6.0.4.0
	Accessible Voting Device			ExpressVote (BMD)	1.5.2.1	ElectionWare 6.0.4.0
	Central Count - Digital Scan			DS450	3.1.1.0	ElectionWare 6.0.4.0
	Central Count - Digital Scan			DS850	3.1.1.0	ElectionWare 6.0.4.0
Coconino	Accessible Voting Device	ES&S	ES&S	ExpressVote (BMD)	2.4.5.1	ElectionWare 6.0.4.0
	Central Count - Digital Scan			DS450	3.1.1.0	ElectionWare 6.0.4.0
Gila	Central Count - Digital Scan	ES&S	ES&S	DS850	3.1.1.0	ElectionWare 5.0.4.0
	Accessible Voting Device			ExpressVote (BMD)	1.5.2.0	ElectionWare 5.0.4.0
Graham	Digital Scan	ES&S	ES&S	DS200	2.17.4.0	ElectionWare 6.0.4.0
	Accessible Voting Device			ExpressVote (BMD)	1.5.2.1	ElectionWare 6.0.4.0
	Central Count - Digital Scan			DS850	3.1.1.0	ElectionWare 6.0.4.0
Greenlee	Digital Scan	ES&S	ES&S	DS200	2.17.4.0	ElectionWare 6.0.4.0
	Accessible Voting Device			ExpressVote (BMD)	2.4.5.0	ElectionWare 6.0.4.0
	Central Count - Digital Scan			DS450	3.1.1.0	ElectionWare 6.0.4.0
La Paz	Digital Scan	ES&S	ES&S	DS200	2.12.2.0	EVS 6.0.4.0
	Accessible Voting Device			ExpressVote (BMD)	1.5.2.1	EVS 6.0.4.0
	Central Count - Digital Scan			DS450	3.1.1.0	EVS 6.0.4.0
Maricopa	Accessible Ballot Marking Device	Dominion	Dominion	ImageCast X BMD	V5.5.13.2	Democracy Suite 5.5b
	Ballot Marking Device Printer	HP		HP LaserJet Pro M402dne	Unmodified COTS	Democracy Suite 5.5b
	Accessible Voting Device	Dominion		ICX ATI Unit	Rev. A	Democracy Suite 5.5b
	Central Count - Digital Scan	Cannon		ICC Cannon DR-G1130	Unmodified COTS	Democracy Suite 5.5b
Mohave	Central Count - Digital Scan	ES&S	ES&S	DS850	3.1.1.0	ElectionWare 6.0.4.0
Navajo	Central Count - Digital Scan	ES&S	ES&S	DS450	3.1.1.0	ElectionWare 5.0.4.0
	Central Count - Digital Scan			DS850	3.1.1.0	ElectionWare 5.0.4.0
	Accessible Voting Device			ExpressVote (BMD)	1.5.2.1	ElectionWare 5.0.4.0
Pima	Central Count - Digital Scan	ES&S	ES&S	DS850 Ver. 1	2.4.0.0	ElectionWare 6.0.4.0
	Accessible Voting Device			ExpressVote (BMD)	2.4.5.1	ElectionWare 6.0.4.0
Pinal	Central Count - Digital Scan	ES&S	ES&S	DS850	3.1.1.0	ElectionWare 6.0.4.0
	Accessible Voting Device			ExpressVote (BMD)	2.4.5.0	ElectionWare 6.0.4.0
Santa Cruz	Accessible Voting Device	ES&S	ES&S	ExpressVote (BMD)	1.5.2.1	ElectionWare 6.0.4.0
	Central Count - Digital Scan			DS450	3.1.1.0	ElectionWare 6.0.4.0
Yavapai	Central Count - Digital Scan	Unisyn	Unisyn	OVCS	2.80	OCS OpenElect 2.1
	Accessible Voting Device			FVT	OpenElect 2.1	OCS OpenElect 2.1
Yuma	Digital Scan	ES&S	ES&S	DS200	2.17.4.0	ElectionWare 6.0.4.0
	Accessible Voting Device			ExpressVote (BMD)	2.4.5.0	ElectionWare 6.0.4.0
	Central Count - Digital Scan			DS450	3.1.1.0	ElectionWare 6.0.4.0

**This list may be updated prior to the next election.*
Revised February 2022

EXHIBIT B

Arizona Secretary of State
Certified Vote Tabulating Equipment
Pursuant to A.R.S. § 16-442

Updated 07/22/2020

Company	Voting System/System Component	Software	Hardware/Firmware	EAC System ID #	Date of Certification
Diebold	GEMS 1-18-24 Voting System *See original certification 11/2005	EMP Model D software (4.6.2)	AccuVote-TSX DRE Model D (4.6.4) ExpressPoll 5000 Electronic Poll Book (CardWriter (1.0) component only) Election Media Processor (EMP) Model D software (4.6.2)	N-1-06-22-22-003 N-1-06-22-22-004 N-1-06-22-22-005	6/15/2007
Dominion	EMS 5.5.32.4	EMS-Election Event Designer (5.5.32.4) EMS- Results Tally & Reporting (5.5.32.4) EMS-Audio Studio (5.5.32.4) EMS-Data Center Manager (5.5.32.4) EMS-Application Server (5.5.32.4) EMS-Network Attached Storage (5.5.32.4) EMS-Database Server (5.5.32.4) EMS-Election Data Translator (5.5.32.4) EMS-Adjudication (5.5.32.1) ImageCast Voter Activation (ICVA) (5.5.32.4) ImageCast Central (ICC) (5.5.32.5) ImageCast Precinct2 (ICP2) (5.5.1.8) ImageCast X (ICX) (5.5.13.2)	ICVA Smart Card Reader (ACR39U) ICC Scanner (Canon Dr-G1130) ICC Scanner (InoTec HiPro 821) ICP2 Hardware (PCOS-330A) Ballot Box-Stacking (ICP2) (PCOS-350A) ICX Classic Hardware (Avalue SID-21V-Z37) ICX BMD Printer (HP M402dne) ICX BMD UPS (APC SMT1500C) Accessible Tactile Interface USB (ATI for ICP2 & ICX) (Rev.A)	DVS-DemSuite5.5-B	11/05//2019
ES&S	EVS 6.0.4.0	ElectionWare (5.0.4.0)	ExpressVote hardware version 1.0 (firmware version 1.5.2.0) ExpressVote hardware version 2.1 (firmware version 2.4.5.0) DS200 hardware versions 1.2 & 1.3, (firmware version 2.17.4.0) DS450 hardware version 1.0 (firmware version 3.1.1.0) DS850 hardware version 1.0 (firmware version 3.1.1.0)	ESSEVS6040	11/05/2019

This is a summary of the vote tabulating equipment certified by the Secretary of State since January 1, 2006.
Copies of the actual certifications can be obtained by contacting the Secretary of State's office.

Arizona Secretary of State
Certified Vote Tabulating Equipment
Pursuant to A.R.S. § 16-442

Updated 07/22/2020

Company	Voting System/System Component	Software	Hardware/Firmware	EAC System ID #	Date of Certification
ES&S	EVS 6.0.2.1.	ElectionWare (5.0.2.0)	ExpressVote hardware version 1.0 (firmware version 1.5.1.0) ExpressVote hardware version 2.1 (firmware version 2.4.3.0) DS200 hardware versions 1.2 & 1.3, (firmware version 2.17.0.0) DS450 hardware version 1.0 (firmware version 3.1.0.0) DS850 hardware version 1.0 (firmware version 3.1.0.0)	ESSEVS6021	05/01/2019
ES&S	EVS 5.2.2.0	ElectionWare (4.7.1.1) Election Reporting Manager (ERM) (8.12.1.1) Event Log Service (1.5.5.0) Removable Media Service (1.4.5.0) ExpressVote Previewer (1.4.1.2) VAT Previewer (1.8.6.1)	* ExpressVote hardware version 1.0 (firmware version 1.4.1.2) ExpressVote Kiosk hardware version 1.0 DS200 hardware versions 1.2.1, 1.2.3, & 1.3 (firmware version 2.12.2.0) AutoMARK A100 hardware version 1.0 (firmware version 1.8.6.1) AutoMARK A200 hardware version 1.1 (firmware version 1.8.6.1) AutoMARK A300 hardware version 1.3 (firmware version 1.8.6.1) DS850 hardware version 1.0 (firmware version 2.10.2.0) DS450 hardware version 1.0 (firmware version 3.0.0.0) Plastic Ballot Box hardware versions 1.2, 1.3 Metal Ballot Box with/without Diverter hardware versions 1.0, 1.1, 1.2	ESSEVS5220	01/08/2018 * ExpressVote hardware version 1.0 (firmware version 1.4.1.2) is conditionally certified through January 2020.

This is a summary of the vote tabulating equipment certified by the Secretary of State since January 1, 2006.
Copies of the actual certifications can be obtained by contacting the Secretary of State's office.

Arizona Secretary of State
Certified Vote Tabulating Equipment
Pursuant to A.R.S. § 16-442

Updated 07/22/2020

Company	Voting System/System Component	Software	Hardware/Firmware	EAC System ID #	Date of Certification
ES&S	EVS 5.2.1.0	ElectionWare (4.7.1.0) Election Reporting Manager (ERM) (8.12.1.0) Event Log Service (1.5.5.0) Removable Media Service (1.4.5.0) VAT Previewer (1.8.6.0)	ExpressVote hardware version 1.0 (firmware version 1.4.1.0) ExpressVote Kiosk hardware version 1.0 DS200 hardware versions 1.2.1, 1.2.3, & 1.3 (firmware version 2.12.1.0) AutoMARK A100 hardware version 1.0 (firmware version 1.8.6.0) AutoMARK A200 hardware versions 1.1, 1.3 (firmware version 1.8.6.0) AutoMARK A300 hardware version 1.3 (firmware version 1.8.6.0) DS850 hardware version 1.0 (firmware version 2.10.1.0) Plastic Ballot Box hardware versions 1.2, 1.3 Metal Ballot Box with/without Diverter hardware versions 1.0, 1.1, 1.2	ESSEVS5210	2/18/2016
ES&S	EVS 5.2.0.0	ElectionWare (4.6.0.0) Election Reporting Manager (ERM) (8.11.0.0) Event Log Service (1.5.5.0) Removable Media Service (1.4.5.0) VAT Previewer (1.8.6.0)	ExpressVote hardware version 1.0 (firmware version 1.4.0.0) DS200 hardware versions 1.2.1, 1.2.3, & 1.3 (firmware version 2.12.0.0) AutoMARK A100 hardware version 1.0 (firmware version 1.8.6.0) AutoMARK A200 hardware versions 1.1, 1.3 (firmware version 1.8.6.0) DS850 hardware version 1.0 (firmware version 2.10.0.0) Plastic Ballot Box hardware versions 1.2, 1.3 Metal Ballot Box with/without Diverter hardware versions 1.0, 1.1, 1.2	ESSEVS5200	12/17/2014

This is a summary of the vote tabulating equipment certified by the Secretary of State since January 1, 2006.
Copies of the actual certifications can be obtained by contacting the Secretary of State's office.

Arizona Secretary of State
Certified Vote Tabulating Equipment
Pursuant to A.R.S. § 16-442

Updated 07/22/2020

Company	Voting System/System Component	Software	Hardware/Firmware	EAC System ID #	Date of Certification
ES&S	EVS 5.0.0.0 Voting System	ElectionWare (4.1.0.0) Event Log Service (1.5.0.0) Election Reporting Manager (ERM) (8.6.0.0) Removable Media Service (1.4.0.0) VAT Previewer (1.8.1.0)	Paper Ballot (3.1.0.0) DS200 hardware version 1.2 (firmware version 2.7.0.0) DS850 hardware version 1.0 (firmware version 2.4.0.0) AutoMARK A100 hardware version 1.0 (firmware version 1.8.1.0) AutoMARK A200 hardware version 1.1 (firmware version 1.8.1.0) AutoMARK A200 hardware version 1.3 (firmware version 1.8.1.0) AutoMARK A300 hardware version 1.3 (firmware version 1.8.1.0) DS200 Plastic Ballot Box (hardware versions 1.2, 1.3) Metal Ballot Box with Diverter (hardware versions 1.0, 1.1, 1.2) Metal Ballot Box without Diverter (hardware versions 1.0, 1.1, 1.2)	ESSEVS5000	6/17/2013
ES&S	Model 650 with the Unity 3.0.1.1 Voting System *See original certification June 15, 2007	Unity 3.0.1.1	Model 650 (2.1.0.0) *Now included as part of the voting system suite previously certified on June 15, 2007	N-2-02-22-22-006 ESSUnity3210	10/19/2011
ES&S	Unity 3.0.1.1 / AutoMark 1.1 Voting System	Audit Manager (7.3.0.0) Election Data Manager (7.4.4.0) ES&S Image Manager (7.4.2.0) Hardware Programming Manager (5.2.4.0) Data Acquisition Manager (6.0.0.0) Election Reporting Manager (7.1.2.1) AutoMARK AIMS (1.2.18)	Model 100 Optical Scan Tabulator (5.2.1.0) AutoMARK Voter Assist Terminal (1.1.2258)	N-2-02-22-22-006 N-2-02-22-22-007 ESSUnity3210	6/15/2007

This is a summary of the vote tabulating equipment certified by the Secretary of State since January 1, 2006.
Copies of the actual certifications can be obtained by contacting the Secretary of State's office.

Arizona Secretary of State
Certified Vote Tabulating Equipment
Pursuant to A.R.S. § 16-442

Updated 07/22/2020

Company	Voting System/System Component	Software	Hardware/Firmware	EAC System ID #	Date of Certification
Sequoia	AVC Edge 5.0/WinEDS 3.1.038	WinEDS (3.1.038) SPR Host (1.09)	AVC Edge II (5.0.24) VeriVote Printer (4.3) HAAT Model 50 Card Activator (1.0.79L) Optech Insight (APX K2.10, HPX K1.42) Optech Insight with Modem (APX K2.10, HPX K1.42, CPX K1.14) 400-C/WinETP (3.00P/1.12.4) Memory Pack Reader (MPR) (2.15)	N-1-07-22-22-003	8/1/2006
Sequoia	WinEDS 3.1.074 / AVC Edge 5.0 Voting System	WinEDS (3.1.074) SPR Host (1.0.10)	AVC Edge II (5.0.31) with VeriVote VVPAT (4.3) & Edge Audio Unit (5.0 Rev. C) HAAT Model 50 Card Activator (2.1.18) Optech Insight (APX K2.12, HPX K1.44) Optech Insight with Modem (APX K2.12, HPX K1.44, CPX K1.14) 400-C/WinETP (3.00P/1.14.3) Memory Pack Reader (MPR) (2.15)	N-1-07-22-22-004	6/15/2007
Sequoia	WinEDS 3.1.074 /AVC Edge 5.0 Voting System	WinEDS (3.1.074)	*WinEDS Election Reporting (1.1.7.0) *WinEDS Election Results (1.1.1.0) *EDGE2plus Model 300 DRE (1.2.33) *Additional components of the previously certified voting system	N-1-07-22-22-004	11/19/2007

This is a summary of the vote tabulating equipment certified by the Secretary of State since January 1, 2006.
Copies of the actual certifications can be obtained by contacting the Secretary of State's office.

Arizona Secretary of State
Certified Vote Tabulating Equipment
Pursuant to A.R.S. § 16-442

Updated 07/22/2020

Company	Voting System/System Component	Software	Hardware/Firmware	EAC System ID #	Date of Certification
Unisyn	OpenElect Voting System 2.1	Ballot Layout Manager (2.1) Election Manager (2.1) Tabulator Client (2.1) Tabulator (2.1) Tabulator Reports (2.1) Validator (2.1) Scripter (2.1) Common (2.1) OCS Installer (2.1) Regkey Builder (2.1) OVCS Application (2.1) Unisyn Secure Library (2.1) Logger (2.1) Auditor (2.1)	OVO hardware (version 2.1) OVI-VC (version 2.1) OVCS (version 2.1) OVCSmini (version 2.1) FVT (version 2.1)	UNS10121966-2.1	05/27/2020
Unisyn	OpenElect Voting System 2.0.A	Ballot Layout Manager (2.0.A) Election Manager (2.0.A) Tabulator Client (2.0.A) Tabulator (2.0.A) Tabulator Reports (2.0.A) Validator (2.0.A) Scripter (2.0.A)	OVO hardware revisions A and E (software version 2.0.A) OVI-VC hardware revisions A and B (software version 2.0.A) OVCS hardware revision ImageFORMULA DR-X10C & M160 (software version 1.3) OVCSmini (software version 2.0.A) FVT hardware revision A (software version 2.0.A)	UNS10121966-2.0.A	9/14/2018

This is a summary of the vote tabulating equipment certified by the Secretary of State since January 1, 2006.
Copies of the actual certifications can be obtained by contacting the Secretary of State's office.

Arizona Secretary of State
Certified Vote Tabulating Equipment
Pursuant to A.R.S. § 16-442

Updated 07/22/2020

Company	Voting System/System Component	Software	Hardware/Firmware	EAC System ID #	Date of Certification
Unisyn	OpenElect 1.3 Voting System	Ballot Layout Manager (1.3) Election Manager (EM) (1.3) Software Server (SS) (1.3) Election Server (ES) (1.3) Tabulator Client (TC) (1.3) Tabulator (Tab) (1.3) Tabulator Reports (TR) (1.3) Scripter (1.3) Validator (1.3)	OVO hardware revisions A and E (software version 1.3) OVI-7 hardware revision F (software verison 1.3) OVI-VC hardware revisions A and B (software version 1.3) OVCS hardware revision v. 1.3 (software version 1.3) Ballot Box - Plastic (hardware version 1.1) Ballot Box - Portable (hardware version 1.2) Ballot Box – Plastic (hardware version 1.3)	04211950-1.3	7/2/2015
Unisyn	OpenElect 1.2 Voting System	Ballot Layout Manager (1.2) Election Manager (EM) (1.2) Software Server (SS) (1.2) Election Server (ES) (1.2) Tabulator Client (TC) (1.2) Tabulator (Tab) (1.2) Tabulator Reports (TR) (1.2) Scripter (1.2) Validator (1.2)	OVO hardware revision E (software version 1.2) OVI-7 hardware revision F (software verison 1.2) OVI-VC hardware revisions A and B (software version 1.2) Ballot Box - Plastic (hardware version 1.1) Ballot Box - Portable (hardware version 1.2)	UNS10121966-OE-1.2	5/23/2014

This is a summary of the vote tabulating equipment certified by the Secretary of State since January 1, 2006.
Copies of the actual certifications can be obtained by contacting the Secretary of State's office.

EXHIBIT C

700 Boulevard South
Suite 102
Huntsville, AL 35802
Phone (256)713-1111
Fax (256)713-1112

Test Report for EAC 2005 VVSG Certification Testing Dominion Voting Systems Democracy Suite (D-Suite) Version 5.5-B Voting System

EAC Project Number: DVS-DemSuite5.5-B

Version: Rev. 02

Date: 08/21/2019

EAC Lab Code 1501

NVLAP LAB CODE 200908-0

SIGNATURES

Approved by: _____ 8/21/19
Michael Walker, VSTL Project Manager — Date

Approved by: _____ 8/21/19
Wendy Owens, VSTL Program Manager — Date

REVISIONS

Revision	Description	Date
00	Initial Release	08/05/2019
01	Updated with EAC Comments	08/20/2019
02	Added ECO information to section 2.2.1.2	08/21/2019

TABLE OF CONTENTS

1.0 INTRODUCTION **1**
1.1 Description and Overview of EAC System Being Modified 1
1.2 References 14
1.3 Terms and Abbreviations 14
2.0 CERTIFICATION TEST BACKGROUND 15
2.1 Revision History 15
2.2 Scope of Testing 16
2.2.1 Modification Overview 17
2.2.1.1 Detailed List of Changes 17
2.2.1.2 ECO Assessment 17
2.2.2 Block Diagram 19
2.2.3 System Limits 20
2.2.4 Supported Languages 21
2.2.5 Supported Functionality 22
2.2.6 VVSG 22
2.2.7 RFIs 22
2.2.8 NOCs 23
3.0 TEST FINDINGS AND RECOMMENDATION 23
3.1 Summary Findings and Recommendation 23
3.1.1 Physical Configuration Audit (PCA) 23
3.1.2 TDP Review 24
3.1.3 QA & CM System Review 30
3.1.4 Source Code Review, Compliance Build, Trusted build, and Build Documentation Review 30
3.1.5 Security Testing 31
3.1.6 System Level Testing 31
3.1.6.1 Functional Configuration Audit (FCA) 32
3.1.6.2 Volume & Stress 33
3.1.6.3 Accuracy 34
3.1.6.4 System Integration 34
3.1.7 Usability and Accessibility Testing 35
3.1.8 Hardware Testing 36

3.2 Anomalies and Resolutions 39
3.3 Deficiencies and Resolutions 39
4.0 RECOMMENDATION FOR CERTIFICATION 39

ATTACHMENT A HARDWARE TEST REPORTS A-1
ATTACHMENT B AS-RUN TEST PLAN B-1

1.0 INTRODUCTION

The purpose of this Test Report is to document the procedures that Pro V&V, Inc. followed to perform certification testing during a system modification campaign for the Dominion Voting Systems Democracy Suite (D-Suite) 5.5-B Voting System to the requirements set forth for voting systems in the U.S. Election Assistance Commission (EAC) 2005 Voluntary Voting System Guidelines (VVSG), Version 1.0. Certification testing of D-Suite 5.5-B was performed to ensure the applicable requirements of the EAC VVSG 1.0 and the EAC Testing and Certification Program Manual, Version 2.0 were met. Additionally, all EAC Request for Interpretations (RFI) and Notices of Clarification (NOC) relevant to the system under test were incorporated in the test campaign.

Prior to submitting the voting system for testing, Dominion Voting Systems submitted an application package to the EAC for certification of the D-Suite 5.5-B Voting System. The application was accepted by the EAC and the project was assigned the unique Project Number of DVS-DemSuite5.5-B.

1.1 Description and Overview of EAC Certified System Being Modified

The EAC Certified System that is the baseline for the submitted modification is described in the following subsections. All information presented was derived from the previous Certification Test Report, the EAC Certificate of Conformance and/or the System Overview.

The D-Suite 5.5-B Voting System is a paper-based optical scan voting system with a hybrid paper/DRE option consisting of the following major components: The Election Management System (EMS), the ImageCast Central (ICC), the ImageCast Precinct (ICP and ICP2), the ImageCast X (ICX) DRE w/ Reports Printer, ImageCast X (ICX) DRE w/VVPAT, the ImageCast Evolution (ICE), and the ImageCast X (ICX) BMD. The D-Suite 5.5-B Voting System configuration is a modification from the EAC approved D-Suite 5.5 system configuration.

The following subsections describe the baselined D-Suite 5.5 Voting System.

Election Management System (EMS)

The D-Suite 5.5 EMS consists of various components running as either a front-end/client application or as a back-end/server application. A listing of the applications and a brief description of each is presented below.

Front-end/Client applications:

- EMS Adjudication: Represents the client component responsible for adjudication, including reporting and generation of adjudicated result files from ImageCast Central tabulators and adjudication of write-in selections from ImageCast Precinct and ImageCast Central tabulators. This client component is installed on both the server and the client machines.

- EMS Audio Studio: A client application that represents an end-user helper application used to record audio files for a given election project. As such, it is utilized during the pre-voting phase of the election cycle.

- EMS Election Data Translator: End-user application used to export election data from election project and import election data into election project.

- EMS Election Event Designer: A client application that integrates election definition functionality together with ballot styling capabilities and represents a main pre-voting phase end-user application

- ImageCast Voter Activation: An application, installed on a workstation or laptop at the polling place, which allows the poll workers to program smart cards for voters. The smart cards are used to activate voting sessions on ImageCast X.

- EMS Results Tally and Reporting: A client application that integrates election results acquisition, validation, tabulation, reporting, and publishing capabilities and represents the main post-voting phase end-user application.

Back-end/Server applications:

- EMS Adjudication Service: Represents a server side application which provides ballot information such as contests, candidates and their coordinates from EMS to the Adjudication application.

- EMS Application Server: Represents a server side application responsible for executing long running processes, such as rendering ballots, generating audio files and election files, etc.

- EMS Database Server: Represents a server side RDBMS repository of the election project database which holds all the election project data, including pre-voting and post-voting data.

- EMS Data Center Manager: A server application that represents a system level configuration application used in EMS back-end data center configuration.

- EMS File System Service: A back-end application that acts as a stand-alone service that runs on client machines, enabling access to low level operating system API for partitioning CF cards, reading raw partition on ICP CF card, etc.

- EMS NAS Server: Represents a server side file repository of the election project file based artifacts, such as ballots, audio files, reports, log files, election files, etc.

- Smart Card Helper Service: A service that is installed on a workstation or laptop at the polling place, and provides required data format for programming smart cards for ImageCast devices, or, for jurisdiction's voting registration system in case of integration.

ImageCast Precinct (ICP)

The ICP device is a hybrid precinct optical scan paper/DRE ballot counter designed to provide six major functionalities: ballot scanning, second chance voting, accessible voting, ballot review, tabulation, and poll worker functions.

For ballot scanning functionality the ICP scans marked paper ballots, interprets voter marks on the paper ballots and stores the ballots for tabulation when the polls are closed.

Second Chance voting refers to scenarios in which an error has been detected on the voter's paper ballot (e.g., blank ballot, undervoted ballot, overvoted ballot, misread ballot, cross-over voted ballot), and the ICP notifies the voter by displaying a message or providing an audio visual cue, that one of these situations has been detected, and offers the voter an opportunity to reject and fix their ballot, or to cast the ballot as-is.

Accessible voting allows voters with disabilities to listen to an audio representation of a ballot and use a hand held controller called an Audio Tactile Interface (ATI) to make vote selections, which are then saved directly to the ICP when the voter casts their Accessible Voting ballot.

The Ballot Review feature allows a voter to review their vote selections using an audio or visual representation, which displays or presents the voter with a complete listing of all contests contained on the ballot and an indication of the results which will be recorded for each contest once the voter's ballot is cast.

The Tabulation of paper ballots and Accessible Voting ballots cast by voters is performed when the polls are closed on the ICP unit and the unit tabulates the results, generates results files for aggregation into RTR, and prints a results report containing the results of the ballots cast.

For poll worker functions the ICP contains a small touch-screen LCD to allow the poll worker to initiate polling place activities, diagnostics and reports.

ImageCast Central (ICC) Count Scanner

The ICC is a high-speed, central ballot scan tabulator based on Commercial off the Shelf (COTS) hardware, coupled with the custom-made ballot processing application software. It is used for high speed scanning and counting of paper ballots.

ImageCast X (ICX)

The Democracy Suite ImageCast X (ICX) consists exclusively of COTS available hardware and operating system, while the applications installed on top customize its behavior to turn it into a Ballot Marking Device (BMD) or a Direct-Recording Electronic (DRE) device with or without a Voter Verifiable Paper Audit Trail (VVPAT). ICX application is the application that verifies voter's session eligibility, using the smart card and then presents the appropriate ballot to the voter. When a voter is satisfied with choices selected, ICX application verifies them and produces an Electronic Mobile Ballot or stores the votes onto a memory device. The ICX is designed to perform the following functions:

- Ballot marking and printing of electronic mobile ballots (in BMD mode)
- Ballot review and second chance voting
- Accessible voting and ballot marking
- Saving voting results (in DRE mode)
- Printing votes on a voter verifiable paper audit trail device (when VVPAT is in use)

1.1.1 Baseline Certified System

The baseline system for this modification is the D-Suite 5.5 Voting System. The tables below describe the certified equipment and firmware versions. Detailed descriptions of the D-Suite 5.5 test campaign are contained in Pro V&V Report No.TR-01-01-DVS-2017-02.01 Rev. B, which is available for viewing on the EAC's website at www.eac.gov.

This subsection lists the proprietary and COTS software to be provided by the manufacturer as part of the test campaign.

Table 1-1. Democracy Suite 5.5 EMS Software Component Descriptions

Software	Version	Filename	Configuration	
			Standard	Express
EMS Election Event Designer (EED)	5.5.12.1	setup.exe: EED_FED_CERT_Setup_x64.msi	X	X
EMS Results Tally and Reporting (RTR)	5.5.12.1	setup.exe: RTR_FED_CERT_Setup_x64.msi	X	X
EMS Application Server	5.5.12.1	setup.exe: APPS_FED_CERT_Setup_x64.msi	X	X
EMS File System Service (FSS)	5.5.12.1	setup.exe: FSSSetup.msi	X	X
EMS Audio Studio (AS)	5.5.12.1	setup.exe: EMSAudioStudioSetup.msi	X	X
EMS Data Center Manager (DCM)	5.5.12.1	DemocracySuiteEMS_DCM.exe	X	X
EMS Election Data Translator (EDT)	5.5.12.1	setup.exe: EDTSetup_x86.msi EDTSetup_x64.msi	X	X
ImageCast Voter Activation (ICVA)	5.5.12.1	setup.exe: ICVASetup.msi	X	X
EMS Adjudication (Adj.)	5.5.8.1	DVS ImageCast Adjudication Client Setup.msi	X	X
EMS Adjudication Service	5.5.8.1	DVS Adjudication Services Setup.msi	X	X
Smart Card Helper Service	5.5.12.1	setup.exe: SmartCardServiceSetup.msi	X	X

Table 1-2. Democracy Suite 5.5 ImageCast Precinct Software Component Descriptions

Firmware/Software	Version	Filename
Election Firmware	5.5.3-0002	cf2xx.sig
Firmware Updater	5.5.3-0002	firmUp.enc
Firmware Extractor	5.5.3-0002	FirmwareExtract.enc
Kernel (uClinux)	5.5.3-0002	image.bin.gz
Boot Loader (COLILO)	20040221	colilo.bin
Asymmetric Key Generator	5.5.3-0002	Keygen.enc
Asymmetric Key Exchange Utility	5.5.3-0002	KeyExchange.enc
Firmware Extractor (Uses Technician Key)	5.5.3-0002	TechExtract.enc

Table 1-3. Democracy Suite 5.5 ImageCast Central Software Component Descriptions

Firmware/Software	Version	Filename
ImageCast Central Application	5.5.3.0002	ICCSetup.exe

Table 1-4. Democracy Suite 5.5 ImageCast X Software Component Descriptions

Firmware/Software	Version	Filename
ICX Application	5.5.10.25	ICX.apk

Table 1-5. Democracy Suite 5.5 EMS Client/Server Software Component Descriptions

Firmware/Software	Version	Filename	Configuration	
			Standard	Express
Microsoft Windows Server	2012 R2 Standard	Physical Media from Microsoft	X	
Microsoft Windows	10 Professional	Physical Media from Microsoft	X	X
.NET Framework	3.5	Physical Media from Microsoft	X	X
Microsoft Visual J#	2.0	vjredist64.exe vjredist.exe	X	X
Microsoft Visual C++ 2013 Redistributable	2013	vcredist_x64.exe vcredist_x86.exe	X	X
Microsoft Visual C++ 2015 Redistributable	2015	vc_redist.x64.exe vc_redist.x86.exe	X	X
Java Runtime Environment	7u80	jre-7u80-windows-x64.exe jre-7u80-windows-i586.exe	X	X
Java Runtime Environment	8u144	jre-8u144-windows-x64.exe jre-8u144-windows-i586.exe	X	X
Microsoft SQL Server 2016 Standard	2016 Standard	Physical Media from Microsoft	X	

Table 1-5. Democracy Suite 5.5 EMS Client/Server Software Component Descriptions *(continued)*

Firmware/Software	Version	Filename	Configuration	
			Standard	Express
Microsoft SQL Server 2016 Service Pack 1	2016 SP1	SQLServer2016SP1-KB3182545-x64-ENU.exe	X	
Microsoft SQL Server 2016 SP1 Express	2016 SP1	SQLEXPRADV_x64_ENU.exe		X
Cepstral Voices	6.2.3.801	Allison (English): Cepstral_Allison_windows_6.2.3.801.exe Alejandra (Spanish): Cepstral_Alejandra_windows_6.2.3.801.exe	X	X
Arial Narrow Fonts	2.37a	ARIALN.TTF ARIALNB.TTF ARIALNBI.TTF ARIALNI.TTF	X	X
Maxim iButton Driver	4.05	install_1_wire_drivers_x86_v405.msi install_1_wire_drivers_x64_v405.msi	X	X
Adobe Reader DC	AcrobatDC	AcroRdrDC1501020060_en_US.exe	X	X
Microsoft Access Database Engine	2010	AccessDatabaseEngine.exe AccessDatabaseEngine_x64.exe	X	X
Open XML SDK 2.0 for Microsoft Office	2.0	OpenXMLSDKv2.msi	X	X

Table 1-6. Democracy Suite 5.5 EMS Software Platform Unmodified COTS Component Descriptions

Firmware/Software	Version	Filename
Infragistics NetAdvanatage Win Forms 2011.1	2011 Vol.1	NetAdvantage_WinForms_20111.msi
Infragistics NetAdvanatage WPF 2012.1	2012 Vol.1	NetAdvantage_WPF_20121.msi
TX Text Control Library for .NET	16.0	TXText Control.NET for Windows Forms 16.0.exe
SOX	14.3.1	sox.exe , libgomp-1.dll, pthreadgc2.dll, zlib1.dll
NLog	1.0.0.505	NLog.dll
iTextSharp	5.0.5	itextsharp.dll
OpenSSL	1.0.2k & 2.0.14 FIPS	openssl.exe, libeay32.dll, ssleay32.dll

Table 1-6. Democracy Suite 5.5 EMS Software Platform Unmodified COTS Component Descriptions *(continued)*

Firmware/Software	Version	Filename
SQLite	1.0.103.0	System.Data.SQLite.DLL (32-bit and 64-bit)
Lame	3.99.4	lame.exe
Speex	1.0.4	speexdec.exe and speexenc.exe
Ghostscript	9.04	gsdll32.dll (32-bit and 64-bit)
One Wire API for .NET	4.0.2.0	OneWireAPI.NET.dll
Avalon-framework-cvs-20020806	20020806	avalon-framework-cvs-20020806.jar
Batik	0.20-5	batik.jar
Fop	0.20-5	fop.jar
Microsoft Visual J# 2.0 Redistributable Package-Second Edition(x64)	2.0	vjc.dll , vjsjbc.dll, vjslibcw.dll, vjsnativ.dll , vjssupuilib.dll , vjsvwaux.dll
Entity framework	6.1.3	EntityFramework.dll
Spreadsheetlight	3.4.3	SpreadsheetLight.dll, SpreadsheetLight.xml
Open XML SDK 2.0 For Microsoft Office	2.0.5022.0	DocumentFormat.OpenXml.dll, DocumentFormat.OpenXml.xml

Table 1-7. Democracy Suite 5.5 ImageCast Precinct Unmodified COTS Component Descriptions

Firmware/Software	Version	Filename
OpenSSL 1.0.2k	1.0.2k	openssl-1.0.2k.tar.gz
OpenSSL FIPS 2.0.10	2.0.10	openssl-fips-2.0.10.tar.gz
Zlib	1.2.3	Zlib-1.2.3.tar.gz

Table 1-8. Democracy Suite 5.5 ImageCast X Unmodified COTS Component Descriptions

Firmware/Software	Version	Filename
Google Text-to-Speech Engine	3.11.12	ARM: com.google.android.tts_3.11.12-210311121_minAPI19(armeabi-v7a)(nodpi).apk x86: com.google.android.tts_3.11.12-210311123_minAPI15(x86)(nodpi).apk
ICX Prime Android 5.1.1 Image	0405	0405_5.1.1-01.12_user_android_x86.iso
ICX Classic Android 4.4.4 Image	0.0.98	byt_t_crv2_64-ota-BCX18-V0.0.98.zip

Table 1-9. Democracy Suite 5.5 ImageCast Central Software Build Library Source Code (Unmodified COTS)

Firmware/Software	Version	Filename
OpenSSL 1.0.2k	1.0.2k	openssl-1.0.2k.tar.gz
OpenSSL FIPS 2.0.10	2.0.10	openssl-fips-2.0.10.tar.gz

Table 1-10. Democracy Suite 5.5 ImageCast Central Runtime Software Components (Unmodified COTS)

Firmware/Software	Version	Filename
1-Wire Driver (x86)	4.05	install_1_wire_drivers_x86_v405.msi
1-Wire Driver (x64)	4.05	install_1_wire_drivers_x64_v405.msi
Canon DR-G1130 TWAIN Driver	1.2 SP6	G1130_DRIT_V12SP6.exe
Canon DR-M160II TWAIN Driver	1.2 SP6	M160II_DRIT_V12SP6.exe
Visual C++ 2013 Redistributable (x86)	12.0.30501	vcredist_x86.exe

Table 1-11. Democracy Suite 5.5 ImageCast Precinct Modified COTS Software Component Descriptions

Firmware/Software	Version	Filename
uClinux	20070130	uClinux-dist-20070130.tar.gz
COLILO Bootloader	20040221	Colilo20040221.tar.gz

Table 1-12. Democracy Suite 5.5 ImageCast X Modified COTS Software Component Descriptions

Firmware/Software	Version	Filename
Zxing Barcode Scanner	4.7.5	BS-4.7.5.zip
SoundTouch	1.9.2	Soundtouch-1.9.2.tar.gz

Table 1-13. Democracy Suite 5.5 EMS Software Build Environment Component Descriptions

Firmware/Software	Version	Filename
Windows 10 Professional	10 Professional	Physical Media from Microsoft
.NET Framework 3.5	3.5	Physical Media from Microsoft
Internet Information Server (IIS)	10.0	Physical Media from Microsoft
7-Zip	9.20 (64 Bit)	7z920-x64.msi
Visual Studio 2015 Professional with Update 3	2015 Update 3	en_visual_studio_professional_2015_with_update_3_x86_x64_web_installer_8922978.exe

Table 1-13. Democracy Suite 5.5 EMS Software Build Environment Component Descriptions
(continued)

Firmware/Software	Version	Filename
.NetDiscUtils	0.10	DiscUtilsBin-0.10.zip
Infragistics NetAdvantage Win Forms 2011.1	2011.1	NetAdvantage_WinForms_20111.msi
Infragistics Net Advantage – WPF 2012.1	2012.1	NetAdvantage_WPF_20121.msi
TX Text Control 16.0.NET	16	TX Text Control.NET for Windows Forms 16.0.exe
Speex	1.0.4	speex_win32_1.0.4_setup.exe
Microsoft Visual J#	2.0	vjredist64.exe
iTextSharp	5.0.5	itextsharp-5.0.5-dll.zip
Ghostscript	9.0.4	gs904w32.exe gs904w64.exe
Nlog	1.0.0.505	NLog-1.0-Refresh-bin.zip
OneWireAPI.NET	4.0	1-wiresdkver400_beta2.zip
Lame	3.99.4	lame3.99.4-20120130.zip
Sox	14.3.1	sox-14.3.1-win32.zip
Avalon Framework	20020806	avalon-framework-cvs-20020806.jar.zip
Fop	0.20-5	fop-0.20.5.jar
Batik	0.20-5	batik-1.5-fop-0.20-5.jar
SQLite	1.0.103.0	sqlite-netFx46-setup-bundle-x64-2015-1.0.103.0.exe
OpenSSL 1.0.2k	1.0.2k	openssl-1.0.2k.tar.gz
OpenSSL FIPS 2.0.10	2.0.10	openssl-fips-2.0.10.tar.gz
Strawberry Perl	5.24.1.1	strawberry-perl-5.24.1.1-64bit.msi
Patch	2.5.9-7	patch-2.5.9-7-bin.zip
ISOnewspaper	30.4	ISOnewspaper30v4_gr.icc.zip
Ogg Vorbis Encoder	2.88	oggenc2.88-1.3.5-generic.zip
Ogg Vorbis Encoder	1.10.1	oggdecV1.10.1.zip
Prism Mvvm	1.1.1	prism.mvvm.1.1.1.nupkg
Bitmiracle.libtiff.net	2.4.560	Bitmiracle.libtiff.net.2.4.560.nupkg
Prism	4.0.0	prism.4.0.0.nupkg
Prism.UnityExtensions	4.0.0	prism.unityextensions.4.0.0.nupkg
PDF Printing	2.9.5.2	PDFPrinting.zip
Entity Framework	6.1.3.net45	entityframework.6.1.3.nupkg

Table 1-13. Democracy Suite 5.5 EMS Software Build Environment Component Descriptions *(continued)*

Firmware/Software	Version	Filename
WiX	3.10	Wix310.exe
Spreadsheet Light	3.4.3	spreadsheetlight.3.4.3.nupkg
Open XML SDK 2.0 for Microsoft Office	2.0	OpenXMLSDKv2.msi
Adobe Reader DC	AcrobatDC	AcroRdrDC1501020060_en_US.exe
Arial Narrow Fonts	2.37a	ArialNarrowFonts.zip
SSH.NET	2016.1.0	SSH.NET-2016.1.0-bin.zip
SSMS	14.0.17119.0	SSMS-Setup-ENU.exe
TwainDSM	2.3.0	Twaindsm-2.3.0.win.bin

Table 1-14. Democracy Suite 5.5 ICC Software Build Environment Component Descriptions

Firmware/Software	Version	Filename
NASM Assembler	2.12.02	nasm-2.12.02-win32.zip
OpenSSL 1.0.2k	1.0.2k	openssl-1.0.2k.tar.gz
OpenSSL FIPS 2.0.10	2.0.10	openssl-fips-2.0.10.tar.gz
CSC3-2010	N/A	CSC3-2010.crl
tss-ca-g2	N/A	tss-ca-g2.crl

Table 1-15. Democracy Suite 5.5 Adjudication Software Build Environment Component Descriptions

Firmware/Software	Version	Filename
Microsoft Enterprise Library	5.0	Enterprise Library 5.0.msi
Microsoft Prism	4.0.0	Prism.4.0.0.nupkg
Microsoft Identity Foundation SDK	4.0	WindowsIdentityFoundation-SDK-4.0.msi
Toggle Switch Control Library	1.1.1	ToggleSwitch 1.1.1.zip
Infragistics NetAdvantage Ultimate 2013.1	2013.1	NetAdvantage_20131_PlatformInstaller.zip
iTextSharp	5.5.1	itextsharp-all-5.5.1.zip
CLR Security	June 2010	clrsecurity_june10.zip
OpenSSL 1.0.2k	1.0.2k	openssl-1.0.2k.tar.gz
OpenSSL FIPS 2.0.10	2.0.10	openssl-fips-2.0.10.tar.gz
Community MSI Extensions	1.4	msiext-1.4.zip
TreeViewEx	3.0.0.0	TreeViewEx.dll

Table 1-16. Democracy Suite 5.5 ImageCast Precinct Election Firmware Compiler Descriptions

Firmware/Software	Version	Filename
g++ (GNU C++ compiler)	gcc3.4.0-20040603	m68k-uclinux-tools-c++-gcc3.4.0-20040603.sh

Table 1-17. Democracy Suite 5.5 ImageCast Precinct Firmware Build Environment Component Descriptions

Firmware/Software	Version	Filename
Ubuntu 16.04.1	16.04.1	ubuntu-16.04.1-desktop-i386.iso
Toolchain Installation Script	N/A	Toolchain.sh
m68k uClinux tools base gcc	3.4.0-20040603	m68k-uclinux-tools-base-gcc3.4.0-20040603.sh
m68k uClinux tools c++ gcc	3.4.0-20040603	m68k-uclinux-tools-c++-gcc3.4.0-20040603.sh
m68k uClinux tools gdb	20040603	m68k-uclinux-tools-gdb-20040603.sh
OpenSSL 1.0.2k	1.0.2k	openssl-1.0.2k.tar.gz
OpenSSL FIPS 2.0.10	2.0.10	openssl-fips-2.0.10.tar.gz

Table 1-18. Democracy Suite 5.5 ImageCast X Firmware Build Environment Component Descriptions

Firmware/Software	Version	Filename
Ubuntu 14.04.4	14.04.4	ubuntu-14.04.4-desktop-amd64.iso

Table 1-19. D-Suite 5.5 Configuration Files

Configuration File	Version	Filename
Machine Configuration File (MCF)	5.5.10.20_20180806	MCF_5.5.10.20_20180806.mcf
Device Configuration File (DCF)	5.4.01_20170521	DCF_5.4.01_20170521.dcf

Table 1-20. D-Suite 5.5 Voting System Equipment

Component	Serial Number
Proprietary Hardware	
ImageCast Precinct Optical Scanner PCOS-320C	AAFAJFM0061, AAFAJFN0030, AAFAJGI6764, AAFAJEL0352
ImageCast Precinct Optical Scanner PCOS-320A	AANAGCP0347, AANAGCP0002
ICP Ballot Box BOX-330A	AAUCCFX0083, AAUCCGI0011
ICX Inline EMI Filter	[DVS-EMIFILTER-001] thru [DVS-EMIFILTER-003]
COTS Hardware	

Table 1-20. D-Suite 5.5 Voting System Equipment *(continued)*

Component	Serial Number
ICX aValue 15” Tablet (SID-15V)	0E14AF00014, B03G005400006, B033G00540008, 9E274118, 1D274118, Baytrail98D750C0, Baytrail12034DCC, Baytrail9A6550C8
ICX aValue 21” Tablet (SID-21V)	0E14AF00027, B03G005500019, 03G005500009, 0039BZ2D, 0039B209, Baytrail0039B22D, BaytrailF1B2587F, BaytrailF1B25983
A Value 21" ICX DRE (Prime)	1707101522, 1707101789, 1707101730, 1707101887, 1707101710, 1707101725, 1707101731, 1708100916, 1708100876, 1708100915, 1707101845, 1707101778, 1717101720, 1707101845, 1707101722, 1707202552, 1711300282, 1707100089, 1707101795, 1707101793, 17101793
SII Thermal Printer	1115271A, 1115273A, 115270A, 1115275A
KFI VRP3 Thermal Printer (VVPAT)	KPR000000715, KPR0000078339, KPR0000078377, KRP000000711, KPR000000712, KPR170900010119, KPR0000078337, KPR0000078364, KPR170900008115, KPR0000078339, KPR170900008116, KPR170900010347, KPR170900009733, KPR170900010120, KPR170900010119, KPR170900010337, KPR170900010338, KPR170900010348
Dell OptiPlex 7440 All In One	HVNRFB2, HVNQFB2, HVNPFB2
Dell PowerEdge R630	4Z07T52
Dell PowerEdge R640	JMP9CM2
Canon imageFormula DR-G1130 Scanner	GF301092, GF304418
Canon DR-M160II Scanner	GX333569, GX333573, GX324846, GX326272, GX319353
Dell Precision T3420 PC	HS0VFB2, HS0TFB2, HS0RFB2, HS0SFB2, 4TB3MN2, F575HH2
HP LaserJet Pro Printer M402dn	PHBQF20342, PHBQF20345, PHBQC12619, PHBQC19613, PHBQC12519, PHBQD18790, PHBQC12616, PHBQG09329
HP LaserJet Pro Printer M402dne	PHB5D00782, PHB5D04714, PHB5F04770, PHB5B18304, PHB5D04713
Dell OptiPlex 9030 All-In-One	CF73S52
Dell Ultrasharp 24” Monitor U2414H	1PVZ152, 62VZ152
Dell OptiPlex 3050 All-In-One	19YWWK2
Smart Card Reader ACR39	RR374-010362

Table 1-21. D-Suite 5.5 Voting System Support Equipment

Component	Serial Number
Dell Monitor KM632	FYNTY12, CKX6Y12, CN-0524N3-72461-59H-6U5U
Dell Monitor P2414Hb	CN-0524N3-74261-5AH-2DNU, CN-0524N3-74261-5AH-2DAU
Dell DVD Multi Recorder GP60NB60	[DVS-Dell-001]
Dell Latitude E7450 Laptop	30GFH72, 369FH72
Dell Latitude e3480 Laptop	1VD3NJ2
Maxim iButton Programmer DS9490R# with DS1402	[DVS-Maxim-001] thru [DVS-Maxim-006]
APC Smart-UPS SMT1500	3S1536X06436, 3S1536X06475, 3S1536X06461, 3S1536X06485, 3S1536X06484, 3S1536X06322, 3S1536X07467, 3S1536X06485, 3S1536X06272, 3S1536X06201, 3S1536X07305, 3S1504X00395, 3S1504X00396, 351716X02289, W51530180004, 3S171X06059
Dell X1008 Network Switch	4R8XX42, 26SXX42
Dell X1018 Network Switch	6TN7Y42, 63SXX42
Enabling Devices Sip and Puff	[DVS-enabling devices-001] - [DVS-enabling devices-002]
Cyber Acoustics Headphones ACM-70	[DVS-cyber acoustics-001] - [DVS-cyber acoustics-005]
4-Way Joystick Controller S26	PME QC 1550 12, [DVS-JOY-001], [DVS-JOY-002]
Enablemart # 88906 Rocker (Paddle) Switch	[DVS-paddle-001]
Dell PowerConnect 2808 Network Switch	3S2P0Z1
IOGEAR SDHC/microSDHC 0U51USC410 Card Reader	8632, 8633
Lexar USB 3.0 Dual-Slot Reader	24020845007435
Hoodman Steel USB 3.0 UDMA Reader 102015	[DVS-hoodman-001]
ATI Handset	98862010101-035, 98862010103-075, 00659010100-046, 98862010100-232, 093015-1-1, 00659010100-035
ATI-USB Handset	02440010100-011, [DVS-ATIUSB-001], [DVS-ATIUSB-002], B104326-1-4-040, B104326-1-4-035
ACS PC-Linked Smart Card Reader ACR39U	RR374-006272, RR374-010356, RR374-010365
Lexar Professional CF Card Reader Workflow CFR1	24050361400108, 24050361401994, 24050361401991, 24050361401990
CORCOM Filter P/N#: 15EMC1	[DVS-CorcomEMIFilter-001]
Delta Filter P/N#: 16PDCG5C	[DVS-DeltaEMIFILTER-001]
Kingston Card Reader FCR-HS4	08738174208132

1.2 References

- Election Assistance Commission 2005 Voluntary Voting System Guidelines (VVSG) Version 1.0, Volume I, "Voting System Performance Guidelines", and Volume II, "National Certification Testing Guidelines"
- Election Assistance Commission Testing and Certification Program Manual, Version 2.0
- Election Assistance Commission Voting System Test Laboratory Program Manual, Version 2.0
- National Voluntary Laboratory Accreditation Program NIST Handbook 150, 2016 Edition, "NVLAP Procedures and General Requirements (NIST HB 150-2016)", dated July 2016
- National Voluntary Laboratory Accreditation Program NIST Handbook 150-22, 2008 Edition, "Voting System Testing (NIST Handbook 150-22)", dated May 2008
- United States 107th Congress Help America Vote Act (HAVA) of 2002 (Public Law 107-252), dated October 2002
- Pro V&V, Inc. Quality Assurance Manual, Revision 1.0
- Election Assistance Commission "Approval of Voting System Testing Application Package" letter dated May 6, 2016
- EAC Requests for Interpretation (RFI) (listed on www.eac.gov)
- EAC Notices of Clarification (NOC) (listed on www.eac.gov)
- Dominion Voting Systems Technical Data Package *(A listing of the D-Suite 5.5-B documents submitted for this test campaign is listed in Section 3.1.2 of this Test Report)*

1.3 Terms and Abbreviations

This subsection lists terms and abbreviations relevant to the hardware, the software, or this Test Plan.

"ADA" – Americans with Disabilities Act 1990

"BMD" – Ballot Marking Device

"CM" – Configuration Management

"COTS" – Commercial Off-The-Shelf

"DRE" – Direct Record Electronic

"EAC" – United States Election Assistance Commission

"EMS" – Election Management System

"FCA" – Functional Configuration Audit

"HAVA" – Help America Vote Act

"ICC" – ImageCast Central

"ICE" – ImageCast Evolution

"ICP" – ImageCast Precinct

"ICX" – ImageCast X

"ISO" – International Organization for Standardization

"NOC" – Notice of Clarification

"PCA" – Physical Configuration Audit

"QA" – Quality Assurance

"RFI" – Request for Interpretation

"TDP" – Technical Data Package

"UPS" – Uninterruptible Power Supply

"VSTL" – Voting System Test Laboratory

"VVPAT" – Voter Verifiable Paper Audit Trail

"VVSG" – Voluntary Voting System Guidelines

2.0 CERTIFICATION TEST BACKGROUND

The D-Suite 5.5-B system is a modification of a previously certified system (D-Suite 5.5). Pro V&V performed an evaluation of results from the previous test campaign to determine the scope of testing required for certification of the D-Suite 5.5-B. Based on this evaluation, Pro V&V determined that testing from the previous test campaign would establish the baseline and that the focus of this test campaign would be on the documented system updates.

2.1 Revision History

The table below details the version history of the D-Suite 5.5-B System:

Table 2-1. D-Suite 5.5-B System Revision History

System Version	Certification Type	Baseline System	Certification Number
D-Suite 5.0	New System	--- (Original System)---	DVS-DemSuite5.0
D-Suite 5.5	Modification	D-Suite 5.0	DVS-DemSuite5.5
D-Suite 5.5-B	Modification	D-Suite 5.5	DVS-DemSuite5.5-B *

*Upon grant of certification by the EAC

2.2 Scope of Testing

The scope of testing focused on the addition of the ImageCast Evolution (ICE) optical ballot hybrid precinct-based scanner and BMD, the ICP2 optical ballot counter, the InoTec HiPro 821 Scanner, and the Canon imageFORMULA DR-M260 Scanner. To evaluate the D-Suite 5.5-B test requirements, the submitted modifications were evaluated against each section of the EAC 2005 VVSG to determine the applicable tests to be performed.

Based on this assessment, it was determined that multiple areas within the EAC 2005 VVSG would be evaluated to encompass the required tests. A breakdown of the areas and associated tests is listed below:

- EAC 2005 VVSG Volume 1, Section 2: Functional Requirements
  - System Integration Testing
  - Functional Configuration Audit (FCA)
  - Physical Configuration Audit (PCA), including System Loads & Hardening
  - Technical Documentation Package (TDP) Review
  - Volume & Stress Testing
  - Accuracy Testing
- EAC 2005 VVSG Volume 1, Section 3: Usability & Accessibility
  - Usability & Accessibility Testing
  - Technical Documentation Package (TDP) Review
- EAC 2005 VVSG Volume 1, Section 4: Hardware Requirements
  - Hardware Testing
  - Technical Documentation Package (TDP) Review
- EAC 2005 VVSG Volume 1, Section 5: Software Requirements
  - Source Code Review, Compliance Build, Trusted Build, and Build Document Review
  - Technical Documentation Package (TDP) Review
  - Functional Configuration Audit (FCA)
- EAC 2005 VVSG Volume 1, Section 7: Security Requirements
  - Security Testing
  - Technical Documentation Package (TDP) Review
  - Functional Configuration Audit (FCA)

2.2.1 Modification Overview

The Dominion Democracy Suite 5.5-B Voting System is a modified voting system configuration that includes upgrades to the components of the D-Suite 5.5 Voting System. Section 2.2.1.1 details changes between this system and the baseline of the Democracy Suite 5.5 Voting System.

To verify the modifications were successfully addressed throughout the test campaign, each modification was tracked and verified to be addressed during the execution of the relevant test area. For example, source code changes were verified during the source code review. Modifications requiring functional test verification were evaluated by executing the standard Accuracy Test, the System Integration Test, or during performance of the FCA. Modifications that were not adequately evaluated during the performance of these tests were subjected to specifically designed test cases. Additionally, Pro V&V functionally verified that any corrected issues from the baseline system were not present in the modified system and that all enhancements implemented did not adversely impact system performance.

2.2.1.1 Detailed List of Changes

General Modifications

- Addition of ImageCast Evolution (ICE), a precinct-level, optical scan, ballot counter (tabulator) designed to perform three major functions: ballot scanning and tabulation, ballot review and second chance voting, and accessible voting and ballot marking. *Submitted version: ICE firmware version 5.5.6.5, model number PCOS 410A.*

- Addition of InoTec HiPro 821 Scanner as a component for the ImageCast Central Count (ICC). *Submitted version: ICC software application version 5.5.32.5, HiPro 821 Scanner driver version 1.2.3.17.*

- Addition of Canon imageFORMULA DR-M260 Scanner as a component for the ImageCast Central Count (ICC). *Submitted version: ICC software application version 5.5.32.5, DR-M260 Scanner driver version 1.1 SP2.*

- Addition of ImageCast Precinct 2 (ICP2) optical ballot counter. The ICP2 is a precinct-based optical scan ballot tabulator that is used in conjunction with ImageCast compatible ballot storage boxes. *Submitted version: ICP2 firmware version 5.5.1.8, model number PCOS-330A.*

EMS Election Event Designer

- Added support for Ballot ID creation by Precinct Portion

ICC

- Corrected drop-out of red colored ovals when using the HiPro scanner

ICP

- Added support for the new election file revision number of 0x0307 which increased the field size of the 'Number of Ballots' field in the VIF ELECTION.DVD file from an unsigned short (2 bytes) to an unsigned integer (4 bytes)

ICX

- Corrected discrepancy between Total Ballots Cast and Report Tape after results are deleted manually from the ICX
- Corrected issue where the application was not providing a warning for low paper amount on VVPAT tape

2.2.1.2 ECO Assessment

Since certification of previous versions of D-Suite systems that are relevant to the D-Suite 5.5-B configuration, DVS has implemented various Engineering Change Orders (ECOs), each of which were evaluated and determined to be De Minimis in nature. Although testing was not warranted during implementation of these ECOs, they are either included as part of the D-Suite 5.5-B system test campaign or may be applied to the D-Suite 5.5-B system. A listing of these ECOs, along with a brief description is provided below:

- ECO 100503 - Adding a COTS collapsible ballot box to AVL for use with the ICP
- ECO 100521 - Added DELL P2419H monitor as a display device for server and client workstations
- ECO 100527 - Added DELL Latitude 3490 computer with updated i3-8130U processor (Dual Core, 4MB Cache, 2.2GHz) to DVS PN 190-000061 (a client workstation)
- ECO 100543 - Update to the DR-G1130 Scanner LCD Panel User Interface
- ECO 100588 - Added new models of VVPAT printer for use with the D-Suite ICX workstation due to previous model becoming commercially unavailable
- ECO 100596 - Added DELL Latitude 3400 computer as a client workstation due to the DELL Latitude 3490 computer becoming commercially unavailable for purchase
- ECO 100597 - Added DELL PowerEdge R640 computer with new processor and RAM as an AVL to the existing R640 server computer configurations
- ECO 100602 - Added DELL Precision 3431 computer in an EMS Express Server and EMS Client Workstation configuration due to the DELL Precision 3430 computer becoming commercially unavailable for purchase
- ECO 100603 - Added DELL P2418HT monitor as a display device for ICC HiPro scanner workstation configuration due to the Lenovo 10QXPAR1US monitor becoming commercially unavailable for purchase

Note: The ECOs listed above were approved by the EAC prior to the issue date of this final test report.

2.2.2 Block Diagram

Figure 1-1. D-Suite 5.5-B System Overview

2.2.3 System Limits

The system limits verified to be supported by the D-Suite 5.5-B System during this test campaign or during testing of the baselined system are provided in the table below.

Table 2-2. D-Suite 5.5-B System Limits by Configuration

Characteristic	Configuration Values		Limiting Component
	Standard	Express	
Portrait Ballot Orientation			
Ballot positions	462	462	*22-inch Portrait Ballot
Precincts in an election	1000	250	Memory
Contests in an election	1000	250	Memory
Candidates/Counters in an election	10000	2500	Memory
Candidates/Counters in a precinct	462	462	*22-inch Portrait Ballot
Candidates/Counters in a tabulator	10000	2500	Memory
Ballot Styles in an election	3000	750	Memory
Contests in a ballot style	156	156	*22-inch Portrait Ballot
Candidates in a contest	231	231	*22-inch Portrait Ballot (Column Span 3)
Ballot styles in a precinct	5	5	Memory
Number of political parties	30	30	No Limitation
“Vote for” in a contest	30	30	No Limitation
Supported languages in an election	5	5	Memory
Number of write-ins	462	462	*22-inch Portrait Ballot
Landscape Ballot Orientation			
Ballot positions	292	292	*22-inch Landscape Ballot (240 candidates + 24 write-ins + 28 Yes/No choices)
Precincts in an election	1000	250	Memory
Contests in an election	1000	250	Memory
Candidates/Counters in an election	10000	2500	Memory
Candidates/Counters in a precinct	240	240	*22-inch Ballot
Candidates/Counters in a tabulator	10000	2500	Memory
Ballot Styles in an election	3000	750	Memory
Contests in a ballot style	38	38	*22-inch Landscape Ballot (24 candidacy contests + 14 Propositions)

Table 2-2. D-Suite 5.5-B System Limits by Configuration *(continued)*

Characteristic	Configuration Values		Limiting Component
	Standard	Express	
Candidates in a contest	240	240	*22-inch Ballot
Ballot styles in a precinct	5	5	Memory
Number of political parties	30	30	Memory
"Vote for" in a contest	24	24	*22-inch Landscape Ballot
Supported languages in an election	5	5	Memory
Number of write-ins	24	24	*22-inch Landscape Ballot

***The 22-inch Ballot length applies only to hand-marked paper ballots.**

2.2.4 Supported Languages

Support for the following languages was verified during this test campaign or during testing of the baselined system:

Table 2-3. D-Suite 5.5-B Supported Languages

Language	ICE	ICP	ICX
Alaska Native	Yes, if using Latin alphabet	Yes	No
Apache	Audio only	Audio only	No
Bengali	Yes	Yes	Yes
Chinese	Yes	Yes	Yes
English	Yes	Yes	Yes
Eskimo	Yes, if using Latin alphabet	Yes	No
Filipino	Yes, if using Latin alphabet	Yes	No
French	Yes	Yes	Yes
Hindi	Yes	Audio only	Yes
Japanese	Yes	Yes	Yes
Jicarilla	Audio only	Audio only	No
Keres	Audio only	Audio only	No
Khmer	Yes	Audio only	No
Korean	Yes	Yes	Yes
Navajo	Audio only	Audio only	No
Seminole	Audio only	Audio only	No
Spanish	Yes	Yes	Yes
Tagalog	No	No	Yes
Thai	Yes	Audio only	Yes
Towa	Audio only	Audio only	No
Ute	Audio only	Audio only	No
Vietnamese	Yes	Yes	Yes
Yuman	Audio only	Audio only	No

Support for all stated languages was verified; however, only English and Spanish language ballots were cast during the performance of functional testing. Additionally, one character based language (Chinese) was tested during System Integration Testing.

For the character based language, the ballot was created by Pro V&V and voted utilizing both paper ballots and ADA voting devices along with all applicable peripherals. The Chinese Language for the ballot was created using a readily available online translation tool. The translated language text was entered into the Election Event Designer Application. A ballot preview was generated in the EED application. The Chinese characters displayed in the ballot preview were compared to the characters generated by the online translation tool, to ensure that the characters matched. The ballots were then generated and printed, and the election loaded onto the tabulators and the BMD units. The Chinese characters displayed on both the printed ballots and displayed on the BMD units were compared to the original Chinese Characters generated by the online translation tool to verify that the characters matched.

2.2.5 Supported Functionality

The D-Suite 5.5-B was verified to support the following voting variations:

- General Election
- Closed Primary
- Open Primary
- Early Voting
- Partisan/Non-Partisan Offices
- Write-In Voting
- Primary Presidential Delegation Nominations
- Split Precincts
- Vote for N of M
- Ballot Rotation
- Provisional or Challenged Ballots

2.2.6 VVSG

The D-Suite 5.5-B Voting System was evaluated against the relevant requirements contained in the EAC 2005 VVSG, Version 1.0.

2.2.7 RFIs

There are no RFIs released by the EAC as of the date of this Test Report that pertain to this test campaign that were not in effect at the time of the baseline system certification.

2.2.8 NOCs

There are no NOCs released by the EAC as of the date of this Test Report that pertain to this test campaign that were not in effect at the time of the baseline system certification.

3.0 TEST FINDINGS AND RECOMMENDATION

The D-Suite 5.5-B Voting System was evaluated against the relevant requirements contained in the EAC 2005 VVSG, Volumes I and II. The focus of this test campaign is the addition of the ImageCast Evolution (ICE) optical ballot hybrid precinct-based scanner and BMD, the ICP2 optical ballot counter, the InoTec HiPro 821 Scanner, and the Canon imageFORMULA DR-M260 Scanner. The summary findings and recommendations for each area of testing are provided in the following sections.

3.1 Summary Findings and Recommendation

Summary findings for the System Level Testing (System Integration Testing, Accuracy Test, Volume & Stress Testing, and FCA), Hardware Testing, Usability & Accessibility Testing, Security Testing, and Source Code Review are detailed in the relevant sections of this report. In addition to these areas of testing, a PCA (including System Loads & Hardening), a TDP Review, and a QA & CM System Review were performed, as described below.

3.1.1 Physical Configuration Audit (PCA)

The Physical Configuration Audit (PCA) compares the voting system components submitted for qualification to the manufacturer's technical documentation, and shall include the following activities:

- Establish a configuration baseline of software and hardware to be tested; confirm whether manufacturer's documentation is sufficient for the user to install, validate, operate, and maintain the voting system
- Verify software conforms to the manufacturer's specifications; inspect all records of manufacturer's release control system; if changes have been made to the baseline version, verify manufacturer's engineering and test data are for the software version submitted for certification
- If the hardware is non-COTS, Pro V&V shall review drawings, specifications, technical data, and test data associated with system hardware to establish system hardware baseline associated with software baseline
- Review manufacturer's documents of user acceptance test procedures and data against system's functional specifications; resolve any discrepancy or inadequacy in manufacturer's plan or data prior to beginning system integration functional and performance tests

- Subsequent changes to baseline software configuration made during testing, as well as system hardware changes that may produce a change in software operation are subject to re-examination

Summary Findings

During execution of the PCA, the components of the D-Suite 5.5-B were documented by component name, model, serial number, major component, and any other relevant information needed to identify the component. For COTS equipment, every effort was made to verify that the COTS equipment had not been modified for use. Additionally, each technical document submitted in the TDP was recorded by document name, description, document number, revision number, and date of release. At the conclusion of the test campaign, test personnel verified that any changes made to the software, hardware, or documentation during the test process were fully and properly documented.

3.1.2 TDP Review

In order to determine compliance of the modified TDP documents with the EAC 2005 VVSG, a limited TDP review was conducted. This review focused on TDP documents that have been modified since the certification of the baseline system. The review consisted of a compliance review to determine if each regulatory, state, or manufacturer-stated requirement had been met based on the context of each requirement. Results of the review of each document were entered on the TDP Review Checklist and reported to the manufacturer for disposition of any anomalies. This process was ongoing until all anomalies were resolved.

Summary Findings

The submitted TDP was determined to be in compliance with the requirements set forth in the EAC 2005 VVSG. A listing of all documents contained in the D-Suite 5.5-B TDP is provided in Table 3-1.

Table 3-1. TDP Documents

Document Number	Description	Version
	Adjudication Documents	
2.05	Democracy Suite Adjudication Software Design and Specification	5.5-B::122
2.08	Democracy Suite Adjudication System Operation Procedures	5.5-B::170
2.09	Democracy Suite Adjudication System Maintenance Manual	5.5-B::102
	D-Suite Documents	
2.02	Democracy Suite System Overview	5.5-B::172
2.06	Democracy Suite System Security Specification	5.5-B::574
2.07	Democracy Suite System Test and Verification	5.5-B::196
2.10	Democracy Suite Personnel Deployment and Training Requirements	5.5-B::132

Table 3-1. TDP Documents *(continued)*

Document Number	Description	Version
2.11	Democracy Suite Configuration Management Process	5.5-B::396
2.12	Democracy Suite Quality Assurance Program	5.5-B::160
2.13	Democracy Suite System Change Notes	5.5-B::173
EMS Documents		
2.03	Democracy Suite EMS Functional Description	5.5-B::383
2.05	Democracy Suite EMS Software Design and Specification	5.5-B::329
2.08	Democracy Suite EMS System Operations Procedures	5.5-B::727
2.09	Democracy Suite EMS System Maintenance Manual	5.5-B::148
---	Democracy Suite EMS System Installation and Configuration Procedure	5.5-B::349
ImageCast Central Documents		
2.03	Democracy Suite ImageCast Central Functionality Description	5.5-B::191
2.05	Democracy Suite ImageCast Central Software Design and Specification	5.5-B::125
2.08	Democracy Suite ImageCast Central System Operation Procedures	5.5-B::213
---	Democracy Suite ImageCast Central Installation and Configuration Procedure	5.5-B::203
ImageCast Evolution Documents		
2.03	Democracy Suite ImageCast Evolution Functionality Description	5.5-B::120
2.04	Democracy Suite ImageCast Evolution System Hardware Specifications	5.5-B::332
2.05	Democracy Suite ImageCast Evolution Software Design and Specifications	5.5-B::175
2.08	Democracy Suite ImageCast Evolution System Operation Procedures	5.5-B::241
2.09	Democracy Suite ImageCast Evolution System Maintenance Manual	5.5-B::165
ImageCast Precinct Documents		
2.03	Democracy Suite ImageCast Precinct Functionality Description	5.5-B::182
2.04	Democracy Suite ImageCast Precinct System Hardware Specification	5.5-B::153
2.04.1	Democracy suite ImageCast Precinct System Hardware Characteristics	5.5-B::98
2.05	Democracy Suite ImageCast Precinct Software Design and Specification	5.5-B::155
2.08	Democracy Suite ImageCast Precinct System Operation Procedures	5.5-B::277
2.09	Democracy Suite ImageCast Precinct System Maintenance Manual	5.5-B::125
ImageCast Precinct2 Documents		

Table 3-1. TDP Documents *(continued)*

Document Number	Description	Version
2.03	Democracy Suite ImageCast Precinct 2 Functionality Description	5.5-B::10
2.04	Democracy Suite ImageCast Precinct 2 System Hardware Specifications	5.5-B-11
2.05	Democracy Suite Imagecast Precinct 2 Software Design and Specifications	5.5-B::16
2.08	Democracy Suite Imagecast Precinct 2 System Operation Procedures	5.5-B::10
2.09	Democracy Suite Imagecast Precinct 2 System Maintenance Manual	5.5-B::9
ImageCast X Documents		
2.03	Democracy Suite ImageCast X Functionality Description	5.5-B::97
2.05	Democracy Suite ImageCast X Software Design and Specification	5.5-B::98
2.08	Democracy Suite ImageCast X System Operation Procedures	5.5-B::85
---	Democracy Suite ImageCast X System Installation and Configuration	5.5-B::87
2.09	Democracy Suite ImageCast X System Maintenance Manual	5.5-B::76
User Guides		
---	Democracy Suite ImageCast Adjudication User Guide	5.5-B::141
---	Democracy Suite Election Data Translator User Guide	5.5-B::89
---	Democracy Suite EMS Audio Studio User Guide	5.5-B::38
---	Democracy Suite EMS Election Event Designer User Guide	5.5-B::339
---	Democracy Suite EMS Results Tally and Reporting User Guide	5.5-B::145
---	Democracy Suite ImageCast Central User Guide	5.5-B::137
---	Democracy Suite ImageCast Precinct User Guide	5.5-B::56
---	Democracy Suite ImageCast Voter Activation User Guide	5.5-B::53
---	Democracy Suite ImageCast X User Guide	5.5-B::252
---	Democracy Suite ImageCast Precinct 2 User Guide	5.5-B::6
---	Canon imageFORMULA DR-M160II User Manual	---
---	Canon imageFORMULA DR-M260 User Manual	---
---	Canon imageFORMULA DR-G1130 DR-G1100 User Manual	---
---	HiPro User Manual	1.2
Supplementary Documents		
----	Common Industry Format for Usability Test Report ImageCast X 5.2 with VVPAT	5.5::1
---	Dell Latitude E7450 Owner's Manual	Rev. A00
---	SID-15V-Z37-A1R User Manual	Rev. 1.0

Table 3-1. TDP Documents *(continued)*

Document Number	Description	Version
---	SID-21V-Z37-A1R User Manual	Rev. 1.0
---	Cyber Acoustics ACM-70B Stereo Headphones Product Sheet	---
---	Democracy Suite ImageCast C++ Coding Standard	5.5-B::59
---	Democracy Suite C# Automated Code Review Process	5.5-B::54
---	Dell Latitude E7450/Latitude 7450 Regulatory Compliance Sheet	Rev. A09
---	Dell OptiPlex 9020 AIO Regulatory Compliance Sheet	Rev. A09
---	Dell OptiPlex 9030 AIO Regulatory Compliance Sheet	Rev. A09
---	Dell Networking X-Series Specification Sheet	Ver. 1.9
---	Dell OptiPlex 9020 All-in-One Technical Specification Sheet	---
---	Dell OptiPlex 9030 All-in-One Technical Specification Sheet	---
---	Dell OptiPlex 3050 All-in-One Technical Specification Sheet	---
---	Google Java Style Dominion XML	---
---	Dominion Voting Systems Java Coding Standards	1.0
---	Dominion Voting Systems JavaScript Coding Standards	1.0
---	ICX Machine Configuration File (MCF) Parameters Settings	5.5-B::31
---	Democracy Suite ImageCast Device Configuration Files	5.5-B::105
---	Democracy Suite ImageCast Printing and Finishing Specification	5.5-B::99
---	Democracy Suite ImageCast Total Results File Format	5.5-B::62
---	Democracy Suite ImageCast Election Definition Files	5.5-B::88
---	Democracy Suite ImageCast Precinct Extracting Firmware Contents	5.5-B::41
---	Democracy Suite ImageCast Precinct Firmware Update Procedure	5.5-B::68
---	Democracy Suite ImageCast Precinct Level One (L1) Maintenance Manual	5.5-B::67
---	Democracy Suite ImageCast Precinct Technical Guide	5.5-B::57
---	Usability Test Report of ImageCast Precinct 5.0 with 36 Participants for VVSG 1.0	5.0::10
---	Usability Test Report of ImageCast X 5.0 with 36 Participants for VVSG 1.0	5.0::13
---	YEDU.E95462 Uninterruptible Power-supply Equipment Sheet	---
---	Dell Latitude E7440 Regulatory Compliance Sheet	Rev. A09
---	Dell Latitude 3480 Regulatory Compliance Sheet	Rev. A11
---	Dell PowerEdge R630 Regulatory Compliance Sheet	Rev. A10
---	Dell Precision T1700 MT Regulatory Compliance Sheet	Rev. A09

Table 3-1. TDP Documents *(continued)*

Document Number	Description	Version
---	Dell PowerConnect 2808 Product Safety. EMC, and Environmental Datasheet	---
---	Dell PowerConnect 2816 Product Safety. EMC, and Environmental Datasheet	---
---	Smart Pro SM1500 Datasheet	---
---	APC Smart-UPS 230V Product Information Sheet	---
---	HP M402dn Datasheet	Rev. 2
---	HP M402dne Datasheet	May 2017
---	Dell Latitude 3480 Owner's Manual	Rev. A00
---	Dell Latitude 3470 Owner's Manual	Rev. A00
---	Dell Precision T3420 Owner's Manual	Rev. A00
---	aValue HID-21V-BTX-A1R User Manual	Rev. 2.0
---	aValue SID-15V-Z37-A1R Data Sheet	---
---	aValue SID-21V-Z37-A1R Data Sheet	---
---	APC Installation and Operation Back-UPS Pro BR1000G	10/2014
---	APC Smart-UPS SMT1500 Operation Manual	03/2013
---	Dell OptiPlex 7440 All-In-One Owner's Manual	Rev. A01
---	Dell OptiPlex 3050 All-In-One Owner's Manual	Rev. A01
---	Dell P2417H Monitor User's Guide	Rev. A01
---	Dell PowerEdge R630 Owner Manual	Rev. A03
---	Lexar Pro USB 3 Dual Slot Reader	---
---	Usability Study of Dominion Voting Systems ImageCast Evolution Versions 4.1.1.1 and 4.6.1.1	1.0.0::37
---	Democracy Suite ImageCast Evolution Firmware Installation Procedure	5.5-B::96
---	Democracy Suite ImageCast Evolution Level One (L1) Maintenance Manual	5.5-B::134
---	Democracy Suite ImageCast Evolution Machine Behavior Settings	5.5-B::96
---	Seiko SII RP-D10 Series User's Guide	Jan. 2018
---	Dell EMC PowerEdge R640 Installation and Service Manual	Rev. A01
---	Dell OptiPlex 7060 Small Form Factor Service Manual	Rev. A00
---	Dell Latitude E7470 Owner's Manual	Rev. A02
---	APC Operation Manual Smart-UPS 750/1000/1500/2200/3000 VA	---
---	ACR38x CCID Smart Card Reader Reference Manual	V6.05

Table 3-1. TDP Documents *(continued)*

Document Number	Description	Version
---	ACR39 Series PC-linked Smart Card Readers Reference Manual	V1.04
---	Ablenet Single Switch Quickstart Guide	---
---	The Programing Group High-Integrity C++Coding Standard Manual	Ver. 2.2
---	HIC++ Standards Model for C++	9.5.4
---	Dominion Voting Voter-verified paper audit trail (VVPAT) Model: VRP3 User Manual Safety Precautions	07 2018
---	Democracy Suite ImageCast Precinct 2 Machine Behavior Settings	5.5-B::10
---	APC Smart-UPS 1500 Specification Sheet	---
---	Democracy Suite ImageCast Precinct 2 Extracting Firmware Contents and Verifying SHA256 Values	5.5-B::13
---	Democracy Suite ImageCast Precinct 2 Level One (L1) Maintenance Manual	5.5-B::9
---	APC Back-UPS BE600M1 User Manual	09/2015
---	APC Back-UPS SMT1500C Operation Manual	01/2017
---	Avalue HID-21V-BTX FactSheet (---)	---
---	Avalue SID 21V Quick Reference	Feb 2018
---	Dell Latitude 3490 Owners Manual	Rev. A01
---	Dell OptiPlex 3050 AIO EMC Emissions Compliance Sheet	Rev. A11
---	Dell Latitude 3400 Setup and Specification Guide	Rev. A00
---	Dell PowerEdge R640 Technical Guide	Rev. A00
---	Dell Networking X Series User Guide	Rev. A06
---	Dell P2419H Monitor User's Guide	Rev. A05
---	DisplayLink DL 125 Product Brief	---
---	HIC Coding Standard	2.2
---	Kingston USB 3.0 High-Speed Media Reader Datasheet	---
---	Lenovo ThinkCentre TIO24Gen3Touch Monitor User Guide	May 2017
---	Dell Optiplex 7050 Tower Owners Manual (A01)	Rev. A01
---	Dell Optiplex 7060 Small Form Factor Service Manual	Rev. A00
---	Dell Optiplex 7060 Tower Setup and Specifications Guide	Rev. A01
---	Dell Precision 3430 Small Form Factor Service Manual	Rev. A00
---	Dell Precision 3430 Small Form Factor Setup and Specifications Guide	Rev. A00
---	Scamax 8x1 Scanner Brochure	---

Table 3-1. TDP Documents *(continued)*

Document Number	Description	Version
---	Tripp Lite SmartPro SM1500RMXL2UTAA Datasheet	---
Build Documents		
---	Democracy Suite ImageCast Evolution Firmware Build, Prerequisite Setup and Installation	5.5-B::95
---	Democracy Suite ImageCast Precinct Firmware Build and Install	5.5-B::90
---	Democracy Suite ImageCast X Build	5.5-B::55
---	Democracy Suite Windows Build Document	5.5-B::40
---	Democracy Suite ImageCast Precinct 2 Build Environment and Prerequisite Setup, Firmware Build and Installation	5.5-B::24

3.1.3 QA & CM System Review

The Dominion Voting Systems Quality and Configuration Management Manuals were reviewed for their fulfillment of Volume I, Sections 8 and 9, and the requirements specified in Volume II, Section 2. The requirements for these sections establish the quality assurance and configuration standards for voting systems to which manufacturers must conform and require voting system manufacturers to implement a quality assurance and configuration management program that is conformant with recognized ISO standards. As part of the review process, the Dominion Voting Systems TDP documents were reviewed to determine if the stated policies were being followed.

Summary Findings

This testing utilized the TDP Review in conjunction with the PCA to determine compliance to the EAC 2005 VVSG requirements and the requirements stated in the Dominion Voting Systems technical documentation. The review of the Quality Assurance and Configuration Management documentation focused on the Dominion Voting Systems' adherence to its stated QA and CM processes. No discrepancies were noted during the reviews.

3.1.4 Source Code Review, Compliance Build, Trusted build, and Build Documentation Review

Pro V&V reviewed the submitted source code to the EAC 2005 VVSG and the manufacturer-submitted coding standards. Prior to initiating the software review, Pro V&V verified that the submitted documentation was sufficient to enable: (1) a review of the source code and (2) Pro V&V to design and conduct tests at every level of the software structure to verify that design specifications and performance guidelines are met.

For the ICE software, a combination of Automated Source Code Review and Manual Source Code Review methods was used to review the source code. For all other components, the submitted source code was compared to the previously certified Democracy Suite 5.5 voting system versions to determine the changes, if any. A combination of Automated Source Code Review and Manual Source Code Review methods was used to review the changes in the source code. In addition, 10% of the source code comments were manually reviewed.

Summary Findings

- Automated Source Code Review: The Automated Source Code Review was performed to review the changes in the source code from the previously certified voting system. No source code issues were found during the Automated Source Code review.
- Manual Source Code Review: The Manual Source Code review was performed on 10% of the comments for compliance to VVSG Volume 1 Section 5.2.7. No source code issues were found during the Manual Source Code review.
- Compliance Build: The compliance build was performed following the compliance review. Once the compliance review was performed and the source was deemed stable enough to proceed with testing, the source code and all additional packages were compiled into a Compliance Build.

Trusted Build: The trusted build consisted of inspecting customer submitted source code, COTS, and Third Party software products and combining them to create the executable code. This inspection followed the documented process from the "United States Election Assistance Commission Voting System Testing and Certification Program Manual" Section 5.5 – 5.7. Performance of the trusted build includes the build documentation review. The Trusted Build was performed following the completion of the Functional Configuration Audit.

3.1.5 Security Testing

A complete security evaluation was performed on a previously certified version of the D-Suite System. The submitted modifications were evaluated during the source code review, security tests, and FCA.

To evaluate the integrity of the system, Pro V&V developed specifically designed test cases in an attempt to defeat the access controls and security measures documented in the system TDP as well verifying compliance to EAC RFI 2012-05. During the security testing, the system was inspected for various controls and measures that were in place to meet the objectives of the security standards which included: protection of the critical elements of the voting system; establishing and maintaining controls to minimize errors; protection from intentional manipulation, fraud and malicious mischief; identifying fraudulent or erroneous changes to the voting system; and protecting the secrecy in the voting process.

Summary Findings

As a result of the Security Testing, it was determined that the D-Suite 5.5-B met the requirements of the security review. Any deficiencies encountered during testing were successfully resolved.

3.1.6 System Level Testing

System Level testing was implemented to evaluate the complete system. This testing included all proprietary components and COTS components (software, hardware, and peripherals) in a configuration of the system's intended use. For software system tests, the tests were designed

according to the stated design objective without consideration of its functional specification. The system level hardware and software test cases were prepared independently to assess the response of the hardware and software to a range of conditions. Pro V&V reviewed the manufacturer's program analysis, documentation, and module test case design and evaluated the test cases for each module with respect to flow control parameters and entry/exit data.

System Level Testing included the evaluations of the following test areas FCA, Accuracy Testing, Volume & Stress, and System Integration Testing. Each of these areas is reported in detail in the subsections that follow.

Component Level Testing was implemented during the FCA for each component and subcomponent. During the source code review, compliance builds, and security testing, Pro V&V utilized limited structural-based techniques (white-box testing). Additionally, specification-based techniques (black-box testing) were utilized for the individual software components.

Pro V&V defined the expected result for each test and the ACCEPT/REJECT criteria for certification. If the system performed as expected, the results were accepted. If the system did not perform as expected, an analysis was performed to determine the cause. If needed, the test was repeated in an attempt to reproduce the results. If the failure could be reproduced and the expected results were not met, the system was determined to have failed the test. If the results could not be reproduced, the test continued. Any errors encountered were documented and tracked through resolution.

To verify the modifications were successfully addressed throughout the test campaign, each modification was tracked and verified to be addressed during the execution of the relevant test area. For example, source code changes were verified during the source code review. Modifications requiring functional test verification were evaluated by executing the standard Accuracy Test, the System Integration Test, or during performance of the FCA. Modifications that were not adequately evaluated during the performance of these tests were subjected to specifically designed test cases.

3.1.6.1 Functional Configuration Audit (FCA)

The functional configuration audit encompasses an examination of manufacturer's tests, and the conduct of additional tests, to verify that the system hardware and software perform all the functions described in the manufacturer's documentation submitted in the TDP.

In addition to functioning according to the manufacturer's documentation, tests will be conducted to insure all applicable EAC 2005 VVSG requirements are met.

For this campaign FCA testing included several exhaustive paths applied in concert:

- FCA-VVSG Testing: Each component of the system was evaluated against a standardized test-case suite centered upon requirements stated in the VVSG and administered through a test-management software tool. All applicable tests-cases were performed while any non-applicable test-cases were logged as "n/a" for substantiation. The system operations and functional

capabilities were categorized in the tool as follows by the phase of election activity in which they are required:

- Pre-voting Capabilities: These functional capabilities are used to prepare the voting system for voting. They include ballot preparation, the preparation of election-specific software (including firmware), the production of ballots, the installation of ballots and ballot counting software (including firmware), and system and equipment tests.
- Voting System Capabilities: These functional capabilities include all operations conducted at the polling place by voters and officials including the generation of status messages.
- Post-voting Capabilities: These functional capabilities apply after all votes have been cast. They include closing the polling place; obtaining reports by voting machine, polling place, and precinct; tabulation of paper ballots at the central location; accumulation of results from all voting methods; obtaining consolidated reports; and obtaining reports of audit trails.

- FCA-Claims Testing: System user instructions and procedures found in the TDP were followed to verify their accuracy and completeness. In addition any functional claims discovered in the TDP that were not specifically examined in other areas or that were items of interest were also tested.
- FCA-Mapping: Any modified functional paths (buttons, dropdowns, etc.) were mapped by qualified VSTL personnel, to help ensure all functional options had been noted and exercised. Any items of interest were examined and/or tested.

All issues (if any) found during these efforts are detailed in Section 3.3. Any issues noted were tracked using an issue tracking software program and issue tracking spreadsheets.

Summary Findings

All functional tests were successfully executed. During execution of the test procedure, it was verified that the D-Suite 5.5-B System successfully completed the system level integration tests with all actual results obtained during test execution matching the expected results.

3.1.6.2 Volume & Stress

The Volume & Stress Tests are utilized to investigate the system's response to conditions that tend to overload the system's capacity to process, store, and report data.

The test parameters focused on the system's stated limits and the ballot logic for areas such as the maximum number of active voting positions, maximum number of ballot styles, maximum candidates, maximum contests, and stated limits within the EMS. This test is utilized to ensure the system could achieve the manufacturer's TDP claims of what the system can support. Testing was performed by exercising multiple election definitions and test cases developed specifically to test for volume and stress conditions of the system being tested.

Summary Findings

During the performance of the Volume and Stress Test, the ICC configured with the Canon DR-M260 scanner encountered a significant number of double feeds when casting 22" ballots. To address the issue Dominion Voting Systems provided a "T" shaped support to provide support for these long ballots, as well as advised that smaller batches of ballots were used when scanning. These changes were documented and addressed the double feed issue and the ICC configured with the Canon DR-M260 scanner successfully completed the Volume and Stress Test. Volume and stress testing was successfully performed on the D-Suite 5.5-B System.

3.1.6.3 Accuracy

The accuracy test ensures that each component of the voting system can each process 1,549,703 consecutive ballot positions correctly within the allowable target error rate. The Accuracy test is designed to test the ability of the system to "capture, record, store, consolidate and report" specific selections and absences of a selection. The required accuracy is defined as an error rate. This rate is the maximum number of errors allowed while processing a specified volume of data. For paper-based voting systems the ballot positions on a paper ballot must be scanned to detect selections for individual candidates and contests and the conversion of those selections detected on the paper ballot converted into digital data.

Summary Findings

The accuracy requirements for the ICE, ICP2, and the ICC units were met by the execution of the standard accuracy test utilizing hand-marked ballots of each ballot length supported by the system, and ballots produced by the ICX BMD, to achieve an accuracy rate greater than 1,549,703 correct ballot positions.

The ICX BMD, ICX DRE, ICP, and the ICX DRE with VVPAT Accuracy test were reused from the Dominion 5.5 EAC Campaign.

3.1.6.4 System Integration

System Integration is a system level test for the integrated operation of both hardware and software. System Integration evaluates the compatibility of the voting system software components or subsystems with one another, and with other components of the voting system environment. This compatibility was determined through functional tests integrating the voting system software with the remainder of the system. During test performance, the system was configured exactly as it would for normal field use. This included connecting all supporting equipment and peripherals including ballot boxes, voting booths (regular and accessible), and any physical security equipment such as locks and ties.

Summary Findings

During System Integration testing, three General Elections and three Primary Elections were successfully exercised on the voting system, as described below:

Three general elections with the following breakdowns:

- General Election GEN-01: A basic election held in 4 precincts, one of which is a split precinct. This election contains 19 contests compiled into 4 ballot styles. 5 of the contests are in all 4 ballot styles. The other 15 contests are split between at least 2 of the precincts with a maximum of 4 different contest spread across the 4 precincts.

- General Election GEN-02: A basic election held in 3 precincts. This election contains 15 contests compiled into 3 ballot styles. 10 of the contests are in all 3 ballot styles with the other five split across the 3 precincts.

- General Election GEN-03: A basic election held in 2 precincts. This election contains 8 contests and compiled into 2 ballot styles. 4 of the contests are in both ballot styles. The other 4 contests are split between the two precincts. This election is designed to functionally test the handling of multiple ballot styles, support for at least three languages including a character-based language, support for common voting variations, and audio support for at least three languages and an ADA binary input device.

Three primary elections with the following breakdowns:

- Primary Election PRIM-01: Open Primary Election in two precincts. This election contained thirty contests compiled into five ballot styles. Each ballot style contains 6 contests.

- Primary Election PRIM-02: Open Primary Election held in two precincts. This election contained thirteen contests compiled into three ballot styles. One contest is in all three ballot styles; all other contests are independent.

- Primary Election PRIM-03: A basic election held in 2 precincts. This election contains 10 contests and is compiled into 2 ballot styles. 2 of the contests are in both ballot styles. The other 8 contests are split between the two parties' ballots. This Primary Election is designed to functionally test the handling of multiple ballot styles, support for at least three languages including a character-based language, support for common voting variations, and audio support for at least three languages and an ADA binary input device.

The D-Suite 5.5-B System successfully passed the System Integration Test. During execution of the test procedure, it was verified that the D-Suite 5.5-B System successfully completed the system level integration tests with all actual results obtained during test execution matching the expected results.

3.1.7 Usability and Accessibility Testing

Usability & Accessibility testing was performed to evaluate the D-Suite 5.5-B System to the applicable requirements. Testing specifically focused on the modifications implemented since the certification of the baseline system.

Usability was defined generally as a measure of the effectiveness, efficiency, and satisfaction achieved by a specified set of users with a given product in the performance of specified tasks.

The Accessibility portion of testing evaluated the requirements for accessibility. These requirements are intended to address HAVA 301 (a) (3) (B).

During test performance, the D-Suite 5.5-B System was configured as per the Dominion Voting Systems TDP. The configured system was tested to the VVSG 1.0 requirements utilizing TestLink which maintains all applicable test cases. Utilization of both negative and positive inputs were entered into the system and documented into TestLink to allow for traceability and reproducibility. All components were evaluated for applicable requirements in which all deficiencies were documented within TestLink and Mantis for tracking purposes. Regression testing was performed on all identified issues to ensure resolution and compliance to the requirements.

Summary Findings

The D-Suite 5.5-B System successfully met the requirements of the Usability and Accessibility evaluation. Any deficiencies encountered during testing were successfully resolved.

3.1.8 Hardware Testing

The ICX BMD, ICX DRE, ICP, the ICX DRE with Report Printer, and the ICX DRE with VVPAT hardware testing were re-used from the previous certification test campaign (D-Suite 5.5). The components added to the modified system required the full suite of hardware and electrical testing as detailed in the 2005 VVSG. These tests are listed below:

Electrical Tests:

- Electrical Power Disturbance
- Electromagnetic Radiation
- Electrostatic Disruption
- Electromagnetic Susceptibility
- Electrical Fast Transient
- Lightning Surge
- Conducted RF Immunity
- Magnetic Fields Immunity
- Electrical Supply

Environmental Tests:

- Bench Handling
- Vibration
- Low Temperature
- High Temperature

- Humidity
- Temperature Power Variation
- Acoustic

Pro V&V utilized third party testing during the performance of hardware testing. All electrical hardware testing was performed at the TUV SUD Canada, Inc. facility located in Ontario, Canada. All testing at the TUV SUD Canada, Inc. facility was witnessed on-site by Pro V&V personnel. All environmental hardware testing was performed at the NTS Longmont facility located in Longmont, Colorado. All testing at the NTS Longmont facility was witnessed on-site by Pro V&V personnel, with the exception of Temperature Power Variation in which Pro V&V qualified staff executed all testing.

Summary Findings

Electrical Testing was performed on the ICE and ICP2 components of the D-Suite 5.5-B System. The procedures and results for this testing are included in the following TUV SUD Canada, Inc. reports:

- Report File #: 7169005633E-00, presented in Attachment A-1
- Report File #: 7169006118E-00, presented in Attachment A-2

The test results from this testing are summarized below:

Table 3-2. Electrical Hardware Test Results

Standard/Method	Description	Criteria	Class/Level	Result
FCC 15.107 ICES-003 VVSG Vol. 1 4.1.2.9	Power Line Conducted Emissions	Normal Operation & No Data Loss	Class B	Pass
FCC 15.109 ICES-003 VVSG Vol. 1 4.1.2.9	Radiated Emissions	Normal Operation & No Data Loss	Class B	Pass
EN61000-4-11 VVSG Vol. 1 4.1.2.5	Electrical Power Disturbance	Normal Operation & No Data Loss	Various	Pass
EN61000-4-4 VVSG Vol. 1 4.1.2.6	Electrical Fast Transient	Normal Operation & No Data Loss	±2kV - Mains	Pass
EN61000-4-5 VVSG Vol. 1 4.1.2.7	Lightning Surge	Normal Operation & No Data Loss	±2kV Line - Line ±2kV Line - Ground	Pass
EN61000-4-2 VVSG Vol. 1 4.1.2.8	Electrostatic Disruption	Normal Operation & No Data Loss	±8kV Contact ±15kV Air	Pass
EN61000-4-3 VVSG Vol. 1 4.1.2.10	Electromagnetic Susceptibility	Normal Operation & No Data Loss	10 V/m, 80 MHz – 1 GHz	Pass

Table 3-2. Electrical Hardware Test Results *(continued)*

Standard/Method	Description	Criteria	Class/Level	Result
EN61000-4-6 VVSG Vol. 1 4.1.2.11	Conducted RF Immunity	Normal Operation & No Data Loss	10 Vrms, 150 kHz – 80 MHz	Pass
EN61000-4-8 VVSG Vol. 1 4.1.2.12	Magnetic Immunity	Normal Operation & No Data Loss	30 A/m	Pass

Environmental Testing was performed on the ICE and ICP2 components of the D-Suite 5.5-B System. The Canon M260 and InoTec HiPro Scanner components were subjected to Temperature-Power Variation Testing only.

The procedures and results for this testing are included in the following NTS Test reports:

- Test Report No. PR094223, presented in Attachment A-3 *(ICE)*
- Test Report No. PR097523, presented in Attachment A-4 *(InoTec HiPro Scanner and ICP2)*

The test results from this testing are summarized below:

Table 3-3. Environmental Hardware Test Results

Standard/Method	Description	Criteria	Result
MIL-STD-810D, 516.3, I-3.8 VVSG Vol. 1 4.1.2.14, VVSG Vol. 2 4.6.2	Shock – Bench Handling	Normal Operation & No Data Loss	Pass
MIL-STD-810D, 514.3, I-3.2.1 VVSG Vol. 1 4.1.2.14, VVSG Vol 2 4.6.3	Vibration - Basic Transportation	Normal Operation & No Data Loss	Pass
MIL-STD-810D, 502.2, II-3 VVSG Vol 1 4.1.2.14, VVSG Vol 2 4.6.4	Low Temperature - Storage	Normal Operation & No Data Loss	Pass
MIL-STD-810D, 501.2, I-3.2 VVSG Vol 1 4.1.2.14, VVSG Vol 2 4.6.5	High Temperature - Storage	Normal Operation & No Data Loss	Pass
MIL-STD-810D, 507.2, I-3.2 VVSG Vol 1 4.1.2.14, VVSG Vol 2 4.6.5	Humidity – Hot/Humid	Normal Operation & No Data Loss	Pass
MIL-STD-810D, 501.2/502.2 VVSG Vol 1 4.1.2.13, 4.3.3, VVSG Vol 2 4.7.1	Reliability, Temp-Power Variation Testing	Normal Operation & No Data Loss	Pass
VVSG Vol. 1 3.1.7.1, 3.2.2.2 (c)	Acoustic Noise Level Test	Normal Operation & No Data Loss	Pass
VVSG Vol. 1 4.1.2.4	Electrical Supply	Normal Operation & No Data Loss	Pass
VVSG Vol. 1 4.7.2	Maintainability	Ease of Maintenance	Pass

3.2 Anomalies and Resolutions

When a result is encountered during test performance that deviates from what is standard or expected, a root cause analysis is performed. Pro V&V considers it an anomaly if no root cause can be determined. In instances in which a root cause is established, the results are then considered deficiencies.

Summary Findings

There were no anomalies encountered during this test campaign.

3.3 Deficiencies and Resolutions

Any violation of the specified requirement or a result encountered during test performance that deviates from what is standard or expected in which a root cause is established was considered to be a deficiency. Any deficiencies encountered were logged throughout the test campaign into the Pro V&V tracking system (Mantis) for disposition and resolution. In each instance, if applicable, the resolutions were verified to be resolved through all required means of testing (regression testing, source code review, and TDP update) as needed. Table 3.4 details the noted deficiencies.

Table 3-4. Noted Deficiencies

ID#	Test Category	Deficiency	Resolution
476	Volume and Stress	A significant number of double feeds occurred while scanning 22" ballots on the ICC configured with the Canon DR-M260 scanner.	Dominion supplied a "T" shaped support to provide support for long ballots, as well as advised that smaller batch sizes be used when scanning.
477	System Loads and Hardening	An error occurred when restarting an ICP2 unit after installing the 5.5.1.1 firmware on the unit. Running the installation process a second time addresses the issue.	This issue was addressed in the 5.5.1.8 release of the ICP2 firmware.
479	General	Using the ATI to vote a ballot caused the ICX application ver. 5.5.13.2 to crash and restart.	This issue was caused by the incorrect build resources being supplied to the VSTL. The correct build resources were supplied and the ICX application version 5.5.13.2 was rebuilt.

4.0 RECOMMENDATION FOR CERTIFICATION

The D-Suite 5.5-B Voting System, as presented for testing, successfully met the requirements set forth for voting systems in the U.S. Election Assistance Commission (EAC) 2005 Voluntary Voting System Guidelines (VVSG), Version 1.0. Additionally, Pro V&V, Inc. has determined

that the D-Suite 5.5-B functioned as a complete system during System Integration Testing. Based on the test findings, Pro V&V recommends the EAC grant the D-Suite 5.5-B System, as identified in Tables 4-1 through 4-29, certification to the EAC 2005 VVSG.

Table 4-1. Democracy Suite 5.5-B EMS Software Component Descriptions

Software	Version	Filename	Configuration	
			Standard	Express
EMS Election Event Designer (EED)	5.5.32.4	setup.exe: EED_FED_CERT_Setup_x64.msi	X	X
EMS Results Tally and Reporting (RTR)	5.5.32.4	setup.exe: RTR_FED_CERT_Setup_x64.msi	X	X
EMS Application Server	5.5.32.4	setup.exe: APPS_FED_CERT_Setup_x64.msi	X	X
EMS File System Service (FSS)	5.5.32.4	setup.exe: FSSSetup.msi	X	X
EMS Audio Studio (AS)	5.5.32.4	setup.exe: EMSAudioStudioSetup.msi	X	X
EMS Data Center Manager (DCM)	5.5.32.4	DemocracySuiteEMS_DCM.exe	X	X
EMS Election Data Translator (EDT)	5.5.32.4	setup.exe: EDTSetup_x86.msi EDTSetup_x64.msi	X	X
ImageCast Voter Activation (ICVA)	5.5.32.4	setup.exe: ICVASetup.msi	X	X
EMS Adjudication (Adj.)	5.5.32.1	DVS ImageCast Adjudication Client Setup.msi	X	X
EMS Adjudication Service	5.5.32.1	DVS Adjudication Services Setup.msi	X	X
Smart Card Helper Service	5.5.32.4	setup.exe: SmartCardServiceSetup.msi	X	X

Table 4-2. Democracy Suite 5.5-B ImageCast Precinct Software Component Descriptions

Firmware/Software	Version	Filename
Election Firmware	5.5.31.1	cf2xx.sig
Firmware Updater	5.5.31.1	firmUp.enc
Firmware Extractor	5.5.31.1	FirmwareExtract.enc
Kernel (uClinux)	5.5.31.1	image.bin.gz
Boot Loader (COLILO)	20040221	colilo.bin
Asymmetric Key Generator	5.5.31.1	Keygen.enc
Asymmetric Key Exchange Utility	5.5.31.1	KeyExchange.enc
Firmware Extractor (Uses Technician Key)	5.5.31.1	TechExtract.enc

Table 4-3. Democracy Suite 5.5-B ImageCast Central Software Component Descriptions

Firmware/Software	Version	Filename
ImageCast Central Application	5.5.32.5	ICCSetup.exe

Table 4-4. Democracy Suite 5.5-B ImageCast X Software Component Descriptions

Firmware/Software	Version	Filename
ICX Application	5.5.13.2	ICX.apk

Table 4-5. Democracy Suite 5.5-B ImageCast Precinct 2 Software Component Descriptions

Firmware/Software	Version	Filename
ICP2 Application	5.5.1.8	dvs-release-image-icp2-5.5.1.8.vhd.7z
ICP2 Application (for units with 2 GB RAM)	5.5.1.8	dvs-release-image-icp2-2G-5.5.1.8.vhd.7z
ICP2 Update Card	5.5.1.8	icp2-update-5.5.1.8

Table 4-6. Democracy Suite 5.5-B ImageCast Evolution Software Component Descriptions

Firmware/Software	Version	Filename
Voting Machine	5.5.6.5	GApplication-5.5.6.5.vhd.7z
Election Application	5.5.6.5	dvs
Linux Kernel	2.6.30.9-dvs-36	uImage
Linux Device File	1.6	mpc8347dvs.dtb
Root File System	5.0.31	rfs
Ram Disk	1.0.2	initrd.img
Motherboard FPGA	1.1.5	ice2_mc_p1.bit
Scanner Board FPGA	1.1.2	ice2_scb_p2.bit
Logger Controller	2.0.2	logger.bin
Power Controller	3.0.5	power.bin
Boot Loader	1.3.4.63	u-boot.bin
Integrated Printer	4.1.6	integratedPrinter.hex, printerFont.hex
Boot Startup Logo	6.0.0	logo_platform.bmp
Linux Startup Logo	6.0.0	logo_os.bmp
Application Startup Logo	6.0.0	application_startup_logo.bmp
Application Verification Logo	6.0.0	logo_application_verification.bmp
Ram Disk Verification Logo	6.0.0	logo_ramdisk_verification.bmp

Table 4-6. Democracy Suite 5.5-B ImageCast Evolution Software Component Descriptions *(continued)*

Firmware/Software	Version	Filename
Linux Tree Verification Logo	6.0.0	logo_os_tree_verification.bmp
Linux Verification Logo	6.0.0	logo_os_verification.bmp
RFS Verification Logo	6.0.0	logo_rfs_verification.bmp

Table 4-7. Democracy Suite 5.5-B EMS Client/Server Software Component Descriptions

Firmware/Software	Version	Filename	Configuration	
			Standard	Express
Microsoft Windows Server	2012 R2 Standard	Physical Media from Microsoft	X	
Microsoft Windows	10 Professional	Physical Media from Microsoft	X	X
.NET Framework	3.5	Physical Media from Microsoft	X	X
Microsoft Visual J#	2.0	vjredist64.exe vjredist.exe	X	X
Microsoft Visual C++ 2013 Redistributable	2013	vcredist_x64.exe vcredist_x86.exe	X	X
Microsoft Visual C++ 2015 Redistributable	2015	vc_redist.x64.exe vc_redist.x86.exe	X	X
Java Runtime Environment	7u80	jre-7u80-windows-x64.exe jre-7u80-windows-i586.exe	X	X
Java Runtime Environment	8u144	jre-8u144-windows-x64.exe jre-8u144-windows-i586.exe	X	X
Microsoft SQL Server 2016 Standard	2016 Standard	Physical Media from Microsoft	X	
Microsoft SQL Server 2016 Service Pack 1	2016 SP1	SQLServer2016SP1-KB3182545-x64-ENU.exe	X	
Microsoft SQL Server 2016 SP1 Express	2016 SP1	SQLEXPRADV_x64_ENU.exe		X
Cepstral Voices	6.2.3.801	Allison (English): Cepstral_Allison_windows_6.2.3.801.exe Alejandra (Spanish): Cepstral_Alejandra_windows_6.2.3.801.exe	X	X
Arial Narrow Fonts	2.37a	ARIALN.TTF ARIALNB.TTF ARIALNBI.TTF ARIALNI.TTF	X	X
Maxim iButton Driver	4.05	install_1_wire_drivers_x86_v405.msi install_1_wire_drivers_x64_v405.msi	X	X

Table 4-7. Democracy Suite 5.5-B EMS Client/Server Software Component Descriptions *(continued)*

Firmware/Software	Version	Filename	Configuration	
			Standard	Express
Adobe Reader DC	AcrobatDC	AcroRdrDC1501020060_en_US.exe	X	X
Microsoft Access Database Engine	2010	AccessDatabaseEngine.exe AccessDatabaseEngine_x64.exe	X	X
Open XML SDK 2.0 for Microsoft Office	2.0	OpenXMLSDKv2.msi	X	X

Table 4-8. Democracy Suite 5.5-B EMS Software Platform Unmodified COTS Component Descriptions

Firmware/Software	Version	Filename
Infragistics NetAdvantage Win Forms 2011.1	2011 Vol.1	NetAdvantage_WinForms_20111.msi
Infragistics NetAdvantage WPF 2012.1	2012 Vol.1	NetAdvantage_WPF_20121.msi
TX Text Control Library for .NET	16.0	TXText Control.NET for Windows Forms 16.0.exe
SOX	14.3.1	sox.exe , libgomp-1.dll, pthreadgc2.dll, zlib1.dll
NLog	1.0.0.505	NLog.dll
iTextSharp	5.0.5	itextsharp.dll
OpenSSL	1.0.2k & 2.0.14 FIPS	openssl.exe, libeay32.dll, ssleay32.dll
SQLite	1.0.103.0	System.Data.SQLite.DLL (32-bit and 64-bit)
Lame	3.99.4	lame.exe
Speex	1.0.4	speexdec.exe and speexenc.exe
Ghostscript	9.04	gsdll32.dll (32-bit and 64-bit)
One Wire API for .NET	4.0.2.0	OneWireAPI.NET.dll
Avalon-framework-cvs-20020806	20020806	avalon-framework-cvs-20020806.jar
Batik	0.20-5	batik.jar
Fop	0.20-5	fop.jar
Microsoft Visual J# 2.0 Redistributable Package-Second Edition(x64)	2.0	vjc.dll , vjsjbc.dll, vjslibcw.dll, vjsnativ.dll , vjssupuilib.dll , vjsvwaux.dll
Entity framework	6.1.3	EntityFramework.dll
Spreadsheetlight	3.4.3	SpreadsheetLight.dll, SpreadsheetLight.xml
Open XML SDK 2.0 For Microsoft Office	2.0.5022.0	DocumentFormat.OpenXml.dll, DocumentFormat.OpenXml.xml

Table 4-9. Democracy Suite 5.5-B ImageCast Precinct Unmodified COTS Component Descriptions

Firmware/Software	Version	Filename
OpenSSL 1.0.2k	1.0.2k	openssl-1.0.2k.tar.gz
OpenSSL FIPS 2.0.10	2.0.10	openssl-fips-2.0.10.tar.gz
Zlib	1.2.3	Zlib-1.2.3.tar.gz

Table 4-10. Democracy Suite 5.5-B ImageCast Precinct 2 Unmodified COTS Component Descriptions

Firmware/Software	Version	Filename
acl	2.2.52	acl-2.2.52.src.tar.gz
attr	2.4.47	attr-2.4.47.src.tar.gz
avahi	0.6.32	avahi-0.6.32.tar.gz
bash	4.3.30	bash-4.3.30.tar.gz
bash-completion	2.4	bash-completion-2.4.tar.xz
busybox	1.24.1	busybox-1.24.1.tar.bz2
bzip2	1.0.6	bzip2-1.0.6.tar.gz
coreutils	8.25	coreutils-8.25.tar.xz
cryptodev-linux	1.8	cryptodev-linux-1.8.tar.gz
cups	2.1.4	cups-2.1.4-source.tar.gz
db	6.0.35	db-6.0.35.tar.gz
dbus	1.10.10	dbus-1.10.10.tar.gz
dbus-glib	0.106	dbus-glib-0.106.tar.gz
dbus-test	1.10.10	dbus-1.10.10.tar.gz
depmodwrapper-cross	1	morty.tar.gz
e2fsprogs	1.43	e2fsprogs.git.tar.gz
expat	2.2.0	expat-2.2.0.tar.bz2
fontconfig	2.12.1	fontconfig-2.12.1.tar.gz
freetype	2.6.5	freetype-2.6.5.tar.bz2
gawk	4.1.3	gawk-4.1.3.tar.gz
gcc-runtime	6.2.0	gcc-6.2.0.tar.bz2
gettext	0.19.8.1	gettext-0.19.8.1.tar.gz
glib-2.0	2.48.2	glib-2.48.2.tar.xz
glibc	2.24	git2_sourceware.org.git.glibc.git.tar.gz

Table 4-10. Democracy Suite 5.5-B ImageCast Precinct 2 Unmodified COTS Component Descriptions *(continued)*

Firmware/Software	Version	Filename
glibc-initial	2.24	git2_sourceware.org.git.glibc.git.tar.gz
glibc-locale	2.24	git2_sourceware.org.git.glibc.git.tar.gz
gmp	6.1.1	gmp-6.1.1.tar.bz2
gnutls	3.5.3	gnutls-3.5.3.tar.xz
gobject-introspection	1.48.0	gobject-introspection-1.48.0.tar.xz
grep	2.25	grep-2.25.tar.xz
icu	57.1	icu4c-57_1-src.tgz
imx-gpu-viv	6.2.2.p0-aarch32	imx-gpu-viv-6.2.2.p0-aarch32.bin
kbd	2.0.3	kbd-2.0.3.tar.xz
kernel-module-imx-gpu-viv	6.2.2.p0	kernel-module-imx-gpu-viv-6.2.2.p0.tar.gz
kmod	23+gitAUTOINC+65a885df5f	git2_git.kernel.org.pub.scm.utils.kernel.kmod.kmod.git.tar.gz
libcap	2.25	libcap-2.25.tar.xz
libcgroup	0.41	libcgroup-0.41.tar.bz2
libcheck	0.10.0	check-0.10.0.tar.gz
libdaemon	0.14	libdaemon-0.14.tar.gz
libdrm	2.4.70	libdrm-2.4.70.tar.bz2
libevdev	1.5.2	libevdev-1.5.2.tar.xz
libffi	3.2.1	libffi-3.2.1.tar.gz
libgcc	6.2.0	libgcc-5-dev_5.4.0-6ubuntu1~16.04.10_amd64.deb
libgcc-initial	6.2.0	libgcc-5-dev_5.4.0-6ubuntu1~16.04.10_amd64.deb
libidn	1.33	libidn-1.33.tar.gz
libinput	1.8.4	libinput-1.8.4.tar.xz
libjpeg-turbo	1.5.0	libjpeg-turbo-1.5.0.tar.gz
libnss-mdns	0.1	nss-mdns-0.10.tar.gz
libpcap	1.7.4	libpcap-1.7.4.tar.gz
libpciaccess	0.13.4	libpciaccess-0.13.4.tar.bz2
libpcre	8.39	libpcre32-3_2%3a8.38-3.1_amd64.deb
libpng	1.6.24	libpng-1.6.24.tar.xz
libpthread-stubs	0.3	libpthread-stubs-0.3.tar.bz2

Table 4-10. Democracy Suite 5.5-B ImageCast Precinct 2 Unmodified COTS Component Descriptions *(continued)*

Firmware/Software	Version	Filename
libtool-cross	2.4.6	libtool-2.4.6.tar.gz
libusb1	1.0.20	libusb-1.0.20.tar.bz2
libusb-compat	0.1.5	libusb-compat-0.1.5.tar.bz2
libxkbcommon	0.6.1	libxkbcommon-0.6.1.tar.xz
libxml2	2.9.4	libxml2-2.9.4.tar.gz
linux-libc-headers	4.9	linux-4.9.tar.xz
log4cplus	1.2.0	log4cplus-1.2.0.tar.gz
m4	1.4.17	m4-1.4.17.tar.gz
mesa	12.0.1	mesa-12.0.1.tar.xz
mtdev	1.1.5	mtdev-1.1.5.tar.bz2
nettle	3.2	nettle-3.2.tar.gz
openssl	1.0.2k	openssl-1.0.2k.tar.gz
openssl-fips	2.0.10	openssl-fips-2.0.10.tar.gz
opkg-utils	0.3.2+gitAUTOINC+3ffe ce9bf1	opkg-0.3.3.tar.gz
pkgconfig	0.29.1+gitAUTOINC+87152c05be	git2_anongit.freedesktop.org.pkg-config.tar.gz
ppp	2.4.7	ppp-2.4.7.tar.gz
procps	3.3.12	procps-ng-3.3.12.tar.xz
qtbase	5.9.6+gitAUTOINC+f4c2fcc052	qtbase-5.9.6+gitAUTOINC+9c50112304.tar.gz
qtdeclarative	5.9.6+gitAUTOINC+dfb e918537	qtdeclarative-5.9.6+gitAUTOINC+283a900c4e.tar.gz
qtgraphicaleffects	5.9.6+gitAUTOINC+3d317f1bb0	qtgraphicaleffects-5.9.6+gitAUTOINC+f61dcaa5e5.tar.gz
qtquickcontrols2	5.9.6+gitAUTOINC+c51 eea8870	qtquickcontrols2-5.9.6+gitAUTOINC+c48b314ede.tar.gz
qtquickcontrols	5.9.6+gitAUTOINC+f050a32a43	qtquickcontrols-5.9.6+gitAUTOINC+ca6bba7163.tar.gz
qtsvg	5.9.6+gitAUTOINC+dec74295e8	qtsvg-5.9.6+gitAUTOINC+7a28db8f5b.tar.gz
qtxmlpatterns	5.9.6+gitAUTOINC+c7c5681004	qtxmlpatterns-5.9.6+gitAUTOINC+4dcae15a5a.tar.gz
quazip	0.7.3	quazip-0.7.3.tar.gz
readline	6.3	readline-6.3.tar.gz
run-postinsts	1	morty.tar.gz

Table 4-10. Democracy Suite 5.5-B ImageCast Precinct 2 Unmodified COTS Component Descriptions *(continued)*

Firmware/Software	Version	Filename
sed	4.2.2	sed-4.2.2.tar.gz
shadow	4.2.1	shadow-4.2.1.tar.xz
shadow-securetty	4.2.1	shadow-4.2.1.tar.xz
sqlite3	3.14.1	sqlite-autoconf-3140100.tar.gz
systemd	230+gitAUTOINC+3a74d4fc90	git2_github.com.systemd.systemd.git.tar.gz
tcl	8.6.6	tcl8.6.6-src.tar.gz
tslib	1.1	tslib-1.1.tar.xz
tzdata	2017a	tzdata2017a.tar.gz
usb-modeswitch	2.2.0	usb-modeswitch-2.2.0.tar.bz2
usb-modeswitch- data	20140529	usb-modeswitch-data-20140529.tar.bz2
util-linux	2.28.1	util-linux-2.28.1.tar.xz
util-macros	1.19.0	util-macros-1.19.0.tar.gz
xz	5.2.2	xz-5.2.2.tar.gz
zbar	0.1	zbar-0.10.tar.bz2
zlib	1.2.8	zlib-1.2.8.tar.xz
flac	1.3.1	flac-1.3.1.tar.xz
gst-plugins-base	1.10.4	gst-plugins-base-1.10.4.tar.xz
gst-plugins-good	1.10.4	gst-plugins-good-1.10.4.tar.xz
gstreamer	1.10.4	gstreamer-1.10.4.tar.xz
harfbuzz	1.3.0	harfbuzz-1.3.0.tar.bz2
libgudev	230	libgudev-230.tar.xz
libical	2.0.0	libical-2.0.0.tar.gz
libogg	1.3.2	libogg-1.3.2.tar.xz
libsamplerate	0.1.8	libsamplerate-0.1.8.tar.gz
libsndfile	1.0.27	libsndfile-1.0.27.tar.gz
libvorbis	1.3.5	libvorbis-1.3.5.tar.xz
speex	1.2rc2	speex-1.2rc2.tar.gz
speexdsp	1.2rc3	speexdsp-1.2rc3.tar.gz
taglib	1.9.1	taglib-1.9.1.tar.gz
qtmultimedia	5.9.6+gitAUTOINC+52f5785cfa	qtmultimedia-5.9.6+gitAUTOINC+52f5785cfa.tar.gz

Table 4-11. Democracy Suite 5.5-B ImageCast Evolution Unmodified COTS Component Descriptions

Firmware/Software	Version	Filename
BusyBox	1.20.2	busybox-1.20.2.tar.bz2
e2fsprogs	1.42.4	e2fsprogs-1.42.4.tar.gz
Expat XML Parser	2.1.0	expat-2.1.0.tar.gz
fontconfig	2.9.0	fontconfig-2.9.0.tar.gz
Freetype	2.4.9	freetype-2.4.9.tar.bz2
I2C Tools for Linux	3.1.0	i2c-tools-3.1.0.tar.bz2
JPEG library	8d	jpegsrc.v8d.tar.gz
libogg	1.3.0	libogg-1.3.0.tar.gz
libPNG	1.5.10	libpng-1.5.10.tar.gz
libusb	1.0.8	libusb-1.0.8.tar.bz2
libusb-compat	0.1.3	libusb-compat-0.1.3.tar.bz2
log4cplus	1.0.4.1	log4cplus-1.0.4.1.tar.bz2
openssl	1.0.2k	openssl-1.0.2k.tar.gz
openssl-fips	2.0.10	openssl-fips-2.0.10.tar.gz
PPP	2.4.5	ppp-2.4.5.tar.gz
quazip	0.5	quazip-0.5.tar.gz
Qt Everywhere Linux	4.7.3	qt-everywhere-opensource-src-4.7.3.tar.gz
skell	1.19	skell-1.19.tar.gz
SoundTouch	1.6.0	soundtouch-1.6.0.tar.gz
speex	1.2rc1	speex-1.2rc1.tar.gz
SQLite	3.7.13	sqlite-autoconf-3071300.tar.gz
Sysfs Utilities	2.1.0	sysfsutils-2.1.0.tar.gz
TIFF library	4.0.1	tiff-4.0.1.tar.gz
timezone	2012b	tzcode2012b.tar.gz
USB ModeSwitch	1.2.4	usb-modeswitch-1.2.4.tar.bz2
zlib	1.2.7	zlib-1.2.7.tar.bz2

Table 4-12. Democracy Suite 5.5-B ImageCast X Unmodified COTS Component Descriptions

Firmware/Software	Version	Filename
Google Text-to-Speech Engine	3.11.12	ARM: com.google.android.tts_3.11.12-210311121_minAPI19(armeabi-v7a)(nodpi).apk x86: com.google.android.tts_3.11.12-210311123_minAPI15(x86)(nodpi).apk
ICX Prime Android 5.1.1 Image	0405	0405_5.1.1-01.12_user_android_x86.iso
ICX Classic Android 4.4.4 Image	0.0.98	byt_t_crv2_64-ota-BCX18-V0.0.98.zip

Table 4-13. Democracy Suite 5.5-B ImageCast Central Software Build Library Source Code (Unmodified COTS)

Firmware/Software	Version	Filename
OpenSSL 1.0.2k	1.0.2k	openssl-1.0.2k.tar.gz
OpenSSL FIPS 2.0.10	2.0.10	openssl-fips-2.0.10.tar.gz

Table 4-14. Democracy Suite 5.5-B ImageCast Central Runtime Software Components (Unmodified COTS)

Firmware/Software	Version	Filename
1-Wire Driver (x86)	4.05	install_1_wire_drivers_x86_v405.msi
1-Wire Driver (x64)	4.05	install_1_wire_drivers_x64_v405.msi
Canon DR-G1130 TWAIN Driver	1.2 SP6	G1130_DRIT_V12SP6.exe
Canon DR-M160II TWAIN Driver	1.2 SP6	M160II_DRIT_V12SP6.exe
Visual C++ 2013 Redistributable (x86)	12.0.30501	vcredist_x86.exe
InoTec HiPro 821 TWAIN Driver	1.2.3.17	TwainSetup-ScamaxUSB3.exe
Canon DR-M260 TWAIN Driver	1.1 SP2	M260_DRIT_V11SP2.zip

Table 4-15. Democracy Suite 5.5-B ImageCast Precinct Modified COTS Software Component Descriptions

Firmware/Software	Version	Filename
uClinux	20070130	uClinux-dist-20070130.tar.gz
COLILO Bootloader	20040221	Colilo20040221.tar.gz

Table 4-16. Democracy Suite 5.5-B ImageCast Precinct 2 Modified COTS Software Component Descriptions

Firmware/Software	Version	Filename
Kernel	4.9.11	zImage
U-BOOT	2017.03	u-boot.bin

Table 4-17. Democracy Suite 5.5-B ImageCast Evolution Modified COTS Software Component Descriptions

Firmware/Software	Version	Filename
Kernel	2.6.30	uImage
U-BOOT	1.3.4	u-boot.bin

Table 4-18. Democracy Suite 5.5-B ImageCast X Modified COTS Software Component Descriptions

Firmware/Software	Version	Filename
Zxing Barcode Scanner	4.7.5	BS-4.7.5.zip
SoundTouch	1.9.2	Soundtouch-1.9.2.tar.gz

Table 4-19. Democracy Suite 5.5-B EMS Software Build Environment Component Descriptions

Firmware/Software	Version	Filename
Windows 10 Professional	10 Professional	Physical Media from Microsoft
.NET Framework 3.5	3.5	Physical Media from Microsoft
Internet Information Server (IIS)	10.0	Physical Media from Microsoft
7-Zip	9.20 (64 Bit)	7z920-x64.msi
Visual Studio 2015 Professional with Update 3	2015 Update 3	en_visual_studio_professional_2015_with_update_3_x86_x64_web_installer_8922978.exe
.NetDiscUtils	0.10	DiscUtilsBin-0.10.zip
Infragistics NetAdvantage Win Forms 2011.1	2011.1	NetAdvantage_WinForms_20111.msi
Infragistics Net Advantage – WPF 2012.1	2012.1	NetAdvantage_WPF_20121.msi
TX Text Control 16.0.NET	16	TX Text Control.NET for Windows Forms 16.0.exe
Speex	1.0.4	speex_win32_1.0.4_setup.exe
Microsoft Visual J#	2.0	vjredist64.exe
iTextSharp	5.0.5	itextsharp-5.0.5-dll.zip

Table 4-19. Democracy Suite 5.5-B EMS Software Build Environment Component Descriptions *(continued)*

Firmware/Software	Version	Filename
Ghostscript	9.0.4	gs904w32.exe, gs904w64.exe
Nlog	1.0.0.505	NLog-1.0-Refresh-bin.zip
OneWireAPI.NET	4.0	1-wiresdkver400_beta2.zip
Lame	3.99.4	lame3.99.4-20120130.zip
Sox	14.3.1	sox-14.3.1-win32.zip
Avalon Framework	20020806	avalon-framework-cvs-20020806.jar.zip
Fop	0.20-5	fop-0.20.5.jar
Batik	0.20-5	batik-1.5-fop-0.20-5.jar
SQLite	1.0.103.0	sqlite-netFx46-setup-bundle-x64-2015-1.0.103.0.exe
OpenSSL 1.0.2k	1.0.2k	openssl-1.0.2k.tar.gz
OpenSSL FIPS 2.0.10	2.0.10	openssl-fips-2.0.10.tar.gz
Strawberry Perl	5.24.1.1	strawberry-perl-5.24.1.1-64bit.msi
Patch	2.5.9-7	patch-2.5.9-7-bin.zip
ISOnewspaper	30.4	ISOnewspaper30v4_gr.icc.zip
Ogg Vorbis Encoder	2.88	oggenc2.88-1.3.5-generic.zip
Ogg Vorbis Encoder	1.10.1	oggdecV1.10.1.zip
Prism Mvvm	1.1.1	prism.mvvm.1.1.1.nupkg
Bitmiracle.libtiff.net	2.4.560	Bitmiracle.libtiff.net.2.4.560.nupkg
Prism	4.0.0	prism.4.0.0.nupkg
Prism.UnityExtensions	4.0.0	prism.unityextensions.4.0.0.nupkg
PDF Printing	2.9.5.2	PDFPrinting.zip
Entity Framework	6.1.3.net45	entityframework.6.1.3.nupkg
WiX	3.10	Wix310.exe
Spreadsheet Light	3.4.3	spreadsheetlight.3.4.3.nupkg
Open XML SDK 2.0 for Microsoft Office	2.0	OpenXMLSDKv2.msi
Adobe Reader DC	AcrobatDC	AcroRdrDC1501020060_en_US.exe
Arial Narrow Fonts	2.37a	ArialNarrowFonts.zip
SSH.NET	2016.1.0	SSH.NET-2016.1.0-bin.zip
SSMS	14.0.17119.0	SSMS-Setup-ENU.exe
TwainDSM	2.3.0	Twaindsm-2.3.0.win.bin

Table 4-20. Democracy Suite 5.5-B ICC Software Build Environment Component Descriptions

Firmware/Software	Version	Filename
NASM Assembler	2.12.02	nasm-2.12.02-win32.zip
OpenSSL 1.0.2k	1.0.2k	openssl-1.0.2k.tar.gz
OpenSSL FIPS 2.0.10	2.0.10	openssl-fips-2.0.10.tar.gz
CSC3-2010	N/A	CSC3-2010.crl
tss-ca-g2	N/A	tss-ca-g2.crl

Table 4-21. Democracy Suite 5.5-B Adjudication Software Build Environment Component Descriptions

Firmware/Software	Version	Filename
Microsoft Enterprise Library	5.0	Enterprise Library 5.0.msi
Microsoft Prism	4.0.0	Prism.4.0.0.nupkg
Microsoft Identity Foundation SDK	4.0	WindowsIdentityFoundation-SDK-4.0.msi
Toggle Switch Control Library	1.1.1	ToggleSwitch 1.1.1.zip
Infragistics NetAdvantage Ultimate 2013.1	2013.1	NetAdvantage_20131_PlatformInstaller.zip
iTextSharp	5.5.1	itextsharp-all-5.5.1.zip
CLR Security	June 2010	clrsecurity_june10.zip
OpenSSL 1.0.2k	1.0.2k	openssl-1.0.2k.tar.gz
OpenSSL FIPS 2.0.10	2.0.10	openssl-fips-2.0.10.tar.gz
Community MSI Extensions	1.4	msiext-1.4.zip
TreeViewEx	3.0.0.0	TreeViewEx.dll

Table 4-22. Democracy Suite 5.5-B ImageCast Precinct Election Firmware Compiler Descriptions

Firmware/Software	Version	Filename
g++ (GNU C++ compiler)	gcc3.4.0-20040603	m68k-uclinux-tools-c++-gcc3.4.0-20040603.sh

Table 4-23. Democracy Suite 5.5-B ImageCast Precinct Firmware Build Environment Component Descriptions

Firmware/Software	Version	Filename
Ubuntu 16.04.1	16.04.1	ubuntu-16.04.1-desktop-i386.iso
Toolchain Installation Script	N/A	Toolchain.sh

Table 4-23. Democracy Suite 5.5-B ImageCast Precinct Firmware Build Environment Component Descriptions *(continued)*

Firmware/Software	Version	Filename
m68k uClinux tools base gcc	3.4.0-20040603	m68k-uclinux-tools-base-gcc3.4.0-20040603.sh
m68k uClinux tools c++ gcc	3.4.0-20040603	m68k-uclinux-tools-c++-gcc3.4.0-20040603.sh
m68k uClinux tools gdb	20040603	m68k-uclinux-tools-gdb-20040603.sh
OpenSSL 1.0.2k	1.0.2k	openssl-1.0.2k.tar.gz
OpenSSL FIPS 2.0.10	2.0.10	openssl-fips-2.0.10.tar.gz

Table 4-24. Democracy Suite 5.5-B ImageCast Precinct 2 Firmware Build Environment Component Descriptions

Firmware/Software	Version	Filename
Ubuntu	16.04 LTS	ubuntu-16.04.5-desktop-amd64.iso
Yocto	imx-morty	morty.tar.gz
asciidoc	8.6.9	asciidoc_8.6.9-3_all.deb
autoconf	2.69	autoconf_2.69-9_all.deb
automake	1.15	automake_1%3a1.15-4ubuntu1_all.deb
p7zip- full	9.20.1~dfsg.1	p7zip-full_9.20.1~dfsg.1-4.2_amd64.deb
avr-libc	1.8.0+Atmel3.5.0	avr-libc_1%3a1.8.0+Atmel3.5.0-1_all.deb
binutils-avr_	2.25+Atmel3.5.0	binutils-avr_2.25+Atmel3.5.0-2_amd64.deb
python-dev	2.7.12	python-dev_2.7.12-1~16.04_amd64.deb
python3	3.5.1	python3-dev_3.5.1-3_amd64.deb
python3-pip	8.1.1	python3-pip_8.1.1-2ubuntu0.4_all.deb
python-pysqlite2	2.7.0	python-pysqlite2_2.7.0-1_amd64.deb
chrpath	0.16	chrpath_0.16-1_amd64.deb
socat	1.7.3.1	socat_1.7.3.1-1_amd64.deb
cvs	1.12.13	cvs_2%3a1.12.13+real-15ubuntu0.1_amd64.deb
desktop-file-utils	0.22	desktop-file-utils_0.22-1ubuntu5.2_amd64.deb
psutils	1.17.dfsg	psutils_1.17.dfsg-2_amd64.deb
docbook-utils	0.6.14	docbook-utils_0.6.14-3ubuntu1_all.deb
libsdl1.2-dev	1.2.15+dfsg1	libsdl1.2-dev_1.2.15+dfsg1-3_amd64.deb
help2man	1.47.3	help2man_1.47.3_amd64.deb

Table 4-24. Democracy Suite 5.5-B ImageCast Precinct 2 Firmware Build Environment Component Descriptions *(continued)*

Firmware/Software	Version	Filename
texi2html	1.82+dfsg1	texi2html_1.82+dfsg1-5_all.deb
libgl1-mesa-dev	18.0.5	libgl1-mesa-dev_18.0.5-0ubuntu0~16.04.1_amd64.deb
libglu1-mesa-dev	9.0.0	libglu1-mesa-dev_9.0.0-2.1_amd64.deb
mercurial	3.7.3	mercurial_3.7.3-1ubuntu1_amd64.deb
g++-5	5.4.0	g++-5_5.4.0-6ubuntu1~16.04.10_amd64.deb
gawk	4.1.3	gawk_1%3a4.1.3+dfsg-0.1_amd64.deb
lzop	1.03	lzop_1.03-3.2_amd64.deb
gcc	5.4.0	gcc-5_5.4.0-6ubuntu1~16.04.10_amd64.deb
gcc-5-multilib	5.4.0	gcc-5-multilib_5.4.0-6ubuntu1~16.04.10_amd64.deb
gcc-avr	4.9.2+Atmel3.5.3	gcc-avr_1%3a4.9.2+Atmel3.5.0-1_amd64.deb
git	2.7.4	git_1%3a2.7.4-0ubuntu1.4_amd64.deb
zlib	1.2.8	zlib1g-dev_1%3a1.2.8.dfsg-2ubuntu4.1_amd64.deb
texinfo	6.1.0.dfsg.1	texinfo_6.1.0.dfsg.1-5_amd64.deb
groff	1.22.3	groff_1.22.3-7_amd64.deb

Table 4-25. Democracy Suite 5.5-B ImageCast Evolution Firmware Build Environment Component Descriptions

Firmware/Software	Version	Filename
Ubuntu	10.04 LTS	ubuntu-10.04.4-desktop-i386.iso
LTIB	10.1.1a	ltib-10-1-1a-sv.tar.gz
g++ (GNU C++ compiler)	gcc-4.5.38-eglibc-2.11.38	freescale-powerpc-linux-gnu-2011.03-38.i686.rpm
autoconf	2.57	autoconf-2.57.tar.bz2
bison	2.3	bison-2.3.tar.bz2
ccache	2.4	ccache-2.4.tar.gz
cksum	19990607	cksum-19990607.tar.gz
cramfs	20081121	cramfs-20081121.tar.gz
distcc	2.18.3	distcc-2.18.3.tar.bz2
dtc	1.2.0	dtc-1.2.0.tar.gz
flex	2.5.33	flex-2.5.33.tar.gz
genext2fs	1.4.1	genext2fs-1.4.1.tar.gz

Table 4-25. Democracy Suite 5.5-B ImageCast Evolution Firmware Build Environment Component Descriptions *(continued)*

Firmware/Software	Version	Filename
gen_init_cpio	2.6.25-rc7	gen_init_cpio-2.6.25-rc7.tar.gz
genromfs	0.5.1	genromfs-0.5.1.tar.gz
git	1.5.6.5	git-1.5.6.5.tar.gz
libtool	1.5	libtool-1.5.tar.gz
lkc	1.4	lkc-1.4.tar.gz
mkspooflinks	3.4	mkspooflinks-3.4.tar.gz
mtd-utils	20060302	mtd-utils-20060302.tar.bz2
mux_server	1.0	mux_server.c
pkg-config	0.21	pkg-config-0.21.tar.gz
sparse	0.4	sparse-0.4.tar.gz
texinfo	4.8	texinfo-4.8.tar.bz2
tunctl	1.5	tunctl-1.5.tar.gz
u-boot-tools	1.1.6	u-boot-tools-1.1.6.tar.bz2
unifdef	1.0	unifdef-1.0.tar.gz
wget	1.9.1	wget-1.9.1.tar.gz
yaffs_utils	20060418	yaffs_utils-20060418.tar.gz
rpm	4.0.4	rpm-4.0.4.tar.gz

Table 4-26. Democracy Suite 5.5-B ImageCast X Firmware Build Environment Component Descriptions

Firmware/Software	Version	Filename
Ubuntu 14.04.4	14.04.4	ubuntu-14.04.4-desktop-amd64.iso

Table 4-27. D-Suite 5.5-B Configuration Files

Configuration File	Version	Filename
ICX Machine Configuration File (MCF)	5.5.12.1_20190510	MCF_5.5.12.1_20190510.mcf
ICP/ICC Device Configuration File (DCF)	5.5.31_20190423	DCF_5.5.31_20190423.dcf
ICE Machine Behavior Settings	5.5.6.3 20190512	behaviorsettings_ICE_5.5.6.3_EAC_20190512.mbs
ICP2 Machine Behavior Settings	5.5.1.4 20190510	behaviorsettings_ICP2_5.5.1.4_EAC_20190510.mbs

Table 4-28. D-Suite 5.5-B Voting System Equipment

Component	Serial Number
Proprietary Hardware	
ImageCast Precinct Optical Scanner PCOS-320C	AAFAJFM0061, AAFAJFN0030, AAFAJGI6764, AAFAJEL0352
ImageCast Precinct Optical Scanner PCOS-320A	AANAGCP0347, AANAGCP0002
ImageCast Precinct 2 Optical Scanner PCOS-330A	FAL18480494, FAL18480510
ImageCast Evolution Optical Scanner PCOS-410A	AAFEBIK1847, AAFEBCN0012
ICP Ballot Box BOX-330A	AAUCCFX0083, AAUCCGI0011
ICE Ballot Box	---
ICX Inline EMI Filter	[DVS-EMIFILTER-001] thru [DVS-EMIFILTER-003]
ICP2 Ballot Box BOX-350A	---
COTS Hardware	
ICX aValue 15” Tablet (SID-15V)	0E14AF00014, B03G005400006, B033G00540008, 9E274118, 1D274118, Baytrail98D750C0, Baytrail12034DCC, Baytrail9A6550C8
ICX aValue 21” Tablet (SID-21V)	0E14AF00027, B03G005500019, 03G005500009, 0039BZ2D, 0039B209, Baytrail0039B22D, BaytrailF1B2587F, BaytrailF1B25983
A Value 21" ICX DRE (Prime)	1707101522, 1707101789, 1707101730, 1707101887, 1707101710, 1707101725, 1707101731, 1708100916, 1708100876, 1708100915, 1707101845, 1707101778, 1717101720, 1707101845, 1707101722, 1707202552, 1711300282, 1707100089, 1707101795, 1707101793, 17101793
SII Thermal Printer	1115271A, 1115273A, 115270A, 1115275A
KFI VRP3 Thermal Printer (VVPAT)	KPR000000715, KPR0000078339, KPR0000078377, KRP000000711, KPR000000712, KPR170900010119, KPR0000078337, KPR0000078364, KPR170900008115, KPR0000078339, KPR170900008116, KPR170900010347, KPR170900009733, KPR170900010120, KPR170900010119, KPR170900010337, KPR170900010338, KPR170900010348
Dell OptiPlex 7440 All In One	HVNRFB2, HVNQFB2, HVNPFB2
Dell PowerEdge R630	4Z07T52
Dell PowerEdge R640	JMP9CM2
Canon imageFormula DR-G1130 Scanner	GF301092, GF304418
Canon DR-M160II Scanner	GX333569, GX333573, GX324846, GX326272, GX319353
Canon DR-M260 Scanner	HG306013, HG306012

Table 4-28. D-Suite 5.5-B Voting System Equipment *(continued)*

Component	Serial Number
InoTec HiPro 821 Scanner	0078K28, 0080K28
Dell Precision T3420 PC	HS0VFB2, HS0TFB2, HS0RFB2, HS0SFB2, 4TB3MN2, F575HH2
HP LaserJet Pro Printer M402dn	PHBQF20342, PHBQF20345, PHBQC12619, PHBQC19613, PHBQC12519, PHBQD18790, PHBQC12616, PHBQG09329
HP LaserJet Pro Printer M402dne	PHB5D00782, PHB5D04714, PHB5F04770, PHB5B18304, PHB5D04713
Dell OptiPlex 9030 All-In-One	CF73S52
Dell Ultrasharp 24" Monitor U2414H	1PVZ152, 62VZ152
Dell OptiPlex 3050 All-In-One	19YWWK2
Smart Card Reader ACR39	RR374-010362

Table 4-29. D-Suite 5.5-B Voting System Support Equipment

Component	Serial Number
Dell Monitor KM632	FYNTY12, CKX6Y12, CN-0524N3-72461-59H-6U5U
Dell Monitor P2414Hb	CN-0524N3-74261-5AH-2DNU, CN-0524N3-74261-5AH-2DAU
Dell DVD Multi Recorder GP60NB60	[DVS-Dell-001]
Dell Latitude E7450 Laptop	30GFH72, 369FH72
Dell Latitude e3480 Laptop	1VD3NJ2
Maxim iButton Programmer DS9490R# with DS1402	[DVS-Maxim-001] thru [DVS-Maxim-006]
APC Smart-UPS SMT1500	3S1536X06436, 3S1536X06475, 3S1536X06461, 3S1536X06485, 3S1536X06484, 3S1536X06322, 3S1536X07467, 3S1536X06485, 3S1536X06272, 3S1536X06201, 3S1536X07305, 3S1504X00395, 3S1504X00396, 351716X02289, W51530180004, 3S171X06059
Dell X1008 Network Switch	4R8XX42, 26SXX42
Dell X1018 Network Switch	6TN7Y42, 63SXX42
Enabling Devices Sip and Puff	[DVS-enabling devices-001] - [DVS-enabling devices-002]
Cyber Acoustics Headphones ACM-70	[DVS-cyber acoustics-001] - [DVS-cyber acoustics-005]
4-Way Joystick Controller S26	PME QC 1550 12, [DVS-JOY-001], [DVS-JOY-002]
Enablemart # 88906 Rocker (Paddle) Switch	[DVS-paddle-001]
Dell PowerConnect 2808 Network Switch	3S2P0Z1

Table 4-29. D-Suite 5.5-B Voting System Support Equipment *(continued)*

Component	Serial Number
IOGEAR SDHC/microSDHC 0U51USC410 Card Reader	8632, 8633
Lexar USB 3.0 Dual-Slot Reader	24020845007435
Hoodman Steel USB 3.0 UDMA Reader 102015	[DVS-hoodman-001]
ATI Handset	98862010101-035, 98862010103-075, 00659010100-046, 98862010100-232, 093015-1-1, 00659010100-035
ATI-USB Handset	02440010100-011, [DVS-ATIUSB-001], [DVS-ATIUSB-002], B104326-1-4-040, B104326-1-4-035
ACS PC-Linked Smart Card Reader ACR39U	RR374-006272, RR374-010356, RR374-010365
Lexar Professional CF Card Reader Workflow CFR1	24050361400108, 24050361401994, 24050361401991, 24050361401990
CORCOM Filter P/N#: 15EMC1	[DVS-CorcomEMIFilter-001]
Delta Filter P/N#: 16PDCG5C	[DVS-DeltaEMIFILTER-001]
Kingston Card Reader FCR-HS4	08738174208132

ATTACHMENT A

Attachment A-1: Report File # 7169005633E-00

Attachment A-2: Report File # 7169006118E-00

Attachment A-3: Test Report No. PR094223

Attachment A-4: Test Report No. PR097523

EXHIBIT D

Certification Test Report - Modification

Report Number: ESY-18004-CTR-01

Prepared for:

Vendor Name	*Election Systems and Software (ES&S)*
Vendor System	*EVS 6.0.4.0*
EAC Application No.	*ESSEVS6040*
Vendor Address	*11208 John Galt Boulevard* *Omaha, Nebraska 68137*

Prepared by:

4720 Independence St.
Wheat Ridge, CO 80033
303-422-1566
www.SLICompliance.com

NVLAP TESTING
NVLAP Lab Code 200733-0

Accredited by the Election Assistance Commission (EAC) for Selected Voting System Test Methods or Services

Revision History

Date	Version	Author	Revision Summary
March 18th, 2019	1.0	J. Panek	Initial Draft
April 23rd, 2019	1.1	J. Panek	Updates to address EAC comments

Disclaimer

The Certification Test results reported herein must not be used by the client to claim product certification, approval, or endorsement by NVLAP, NIST, or any agency of the Federal Government. Results herein relate only to the items tested.

Copyright © 2019 by SLI Compliance^{SM}, a Division of Gaming Laboratories International, LLC

Trademarks

- SLI is a registered trademark of SLI Compliance.
- All products and company names are used for identification purposes only and may be trademarks of their respective owners.

The tests referenced in this document were performed in a controlled environment using specific systems and data sets, and results are related to the specific items tested. Actual results in other environments may vary.

Opinions and Interpretations

There are no opinions or interpretations included in this report, except as noted under Recommendations.

TABLE OF CONTENTS

CERTIFICATION TEST REPORT - MODIFICATION 1

1 INTRODUCTION 4

1.1 CERTIFICATION TEST REPORT ATTACHMENTS 4
1.2 REFERENCES 4
1.3 TERMS AND ABBREVIATIONS 5
1.4 SYSTEM IDENTIFICATION 7
1.4.1 Electionware® 7
1.4.2 ExpressVote XL™ 7
1.4.3 ExpressTouch® 7
1.4.4 ExpressVote® Hardware 1.0 7
1.4.5 ExpressVote® Hardware 2.1 7
1.4.6 DS200® 7
1.4.7 DS450® 8
1.4.8 DS850® 8
1.4.9 Event Log Service (ELS) 8
1.4.10 Removable Media Service (RMS) 8
1.4.11 Block Diagrams 9
1.5 SOFTWARE AND FIRMWARE 12
1.5.1 Manufacturer Software/Firmware 12
1.5.2 COTS Software/Firmware 13
1.6 EQUIPMENT 14
1.6.1 ES&S EVS 6.0.4.0 Equipment 14
1.6.2 COTS Equipment 15
1.7 TEST MATERIALS 17
1.8 ES&S EVS 6.0.4.0 DOCUMENTATION 18
1.9 MODIFICATIONS 18

2 CERTIFICATION TEST BACKGROUND 20

2.1 SYSTEM REVISION HISTORY 20
2.2 IMPLEMENTATION STATEMENT 20
2.3 PCA - DOCUMENT AND SOURCE CODE REVIEWS 21
2.4 FCA - FUNCTIONAL & SYSTEM TESTING 21
2.4.1 Test Methods 21
2.4.2 3rd Party Hardware Testing 22

3 CERTIFICATION TEST RESULTS SUMMARY 23

3.1 SOURCE CODE REVIEW SUMMARY 23
3.2 TECHNICAL DATA PACKAGE REVIEW SUMMARY 23
3.3 HARDWARE TESTING SUMMARY 23
3.4 FUNCTIONAL TESTING SUMMARY 25
3.4.1 Accuracy Test Suite 25
3.4.2 Integration Test Suite 25
3.4.3 Modification Test Suite 25
3.4.4 Pennsylvania Straight Party Method Test Suite 26
3.4.5 Security Test Suite 26

4 RECOMMENDATIONS 26

5 APPROVAL SIGNATURES 27

6 APPENDIX A – ANCILLARY PRODUCTS 27

1 INTRODUCTION

SLI Compliance is submitting this Certification Test Report as a summary of the modification and regression testing performed on the **ES&S EVS 6.0.4.0** system against the Voluntary Voting System Guidelines 1.0 (VVSG 1.0). **ES&S EVS 6.0.4.0** is a modification of the **ES&S EVS 6.0.2.0** voting system, certified by the EAC on October 4th, 2018. The system was tested based on the "modified system" requirements, as set forth in section 4.6.2.3 of the "EAC Voting System Testing and Certification Program Manual, version 2.0". The purpose of this document is to provide an overview of the testing and findings for the **ES&S EVS 6.0.4.0** voting system.

This effort included review of updates made to the Technical Data Package, source code changes, modification, and regression testing of the **ES&S EVS 6.0.4.0** voting system. The process consisted of the development of a test plan, managing system configurations, component and system level tests prepared by SLI, and analysis of results. The review and testing were performed at SLI's Wheat Ridge, Colorado facility.

1.1 Certification Test Report Attachments

The following attachments apply to this Certification Test Report:

- Attachment A – ES&S EVS6040 Attestation Letter
- Attachment B – ES&S EVS6040 Implementation Statement
- Attachment C – ES&S EVS6040 TDP Document List
- Attachment D – ES&S EVS6040 Trusted Build Record
- Attachment E – Accredited Hardware Test Lab Certifications
- Attachment F – ES&S EVS6040 Hardware Test Plans
- Attachment G – ES&S EVS6040 As Run Test Plan
- Attachment H – ES&S EVS6040 Discrepancy Report
- Attachment I – ES&S EVS6040 Hardware Test Reports
- Attachment J – ES&S EVS6040 Safety Report

1.2 References

The following key documents were used in preparing this test plan.

1. Election Assistance Commission Voluntary Voting System Guidelines (EAC VVSG), 2005 Version 1.0 Volumes I and II.
2. NIST Handbook 150: 2016.
3. NIST Handbook 150-22: 2017.
4. EAC Voting System Testing and Certification Program Manual, United States Election Assistance Commission, v 2.0, May 2015
5. SLI VSTL Quality System Manual, v 3.0, February 12, 2019.

1.3 Terms and Abbreviations

The following terms and abbreviations will be used throughout this document:

Table 5 – Terms and Abbreviations

Term	Abbreviation	Description
American Association for Laboratory Accreditation	A2LA	A nonprofit, non-governmental, public service, membership society whose mission is to provide comprehensive services in laboratory accreditation and laboratory-related training.
Cast Vote Record	CVR	Permanent record of all votes produced by a single voter whether in electronic, paper or other form. Also referred to as ballot image when used to refer to electronic ballots.
Compact Flash card	CF	This is a type of flash memory card in a standardized enclosure often used in voting systems to store ballot and/or vote results data.
Commercial Off the Shelf	COTS	Term used to designate computer software, hardware or accessories that are ready-made and available for sale, lease, or license to the general public.
Direct Recording Electronic	DRE	Voting systems that, using touch screen or other user interfaces, directly record the voter's selections in each race or contest on the ballot in electronic form.
Election Assistance Commission	EAC	An independent, bipartisan commission created by the Help America Vote Act (HAVA) of 2002 that operates the federal government's voting system certification program.
Election Management System	EMS	Typically, a database management system used to enter jurisdiction information (district, precincts, languages, etc.) as well as election specific information (races, candidates, voter groups (parties), etc.). In addition, the EMS is also used to layout the ballots, download the election data to the voting devices, upload the results and produce the final results reports.
Electromagnetic Compatibility	EMC	The goal of EMC is to validate the correct functioning of different equipment in the same environment and the avoidance of any interference effects between them.
Functional Configuration Audit	FCA	The testing activities associated with the functional testing of the system.
Hybrid Device	No Abbreviation	A device that combines features of two or more functionalities that traditionally have been implemented separately. For example, ExpressVote HW2.1 can function as a vote capture device and as a precinct scanner, two functions that traditionally have not been implemented in the same hardware device.

Term	Abbreviation	Description
National Institute of Standards and Technology	NIST	A non-regulatory federal agency within the U.S. Dept. of Commerce. Its mission is to promote U.S. innovation and industrial competitiveness by advancing measurement science, standards, and technology in ways that enhance economic security and improve our quality of life.
National Voluntary Laboratory Accreditation Program	NVLAP	A division of NIST that provides third-party accreditation to testing and calibration laboratories.
Physical Configuration Audit	PCA	Confirms that the documentation submitted meets the national certification requirements. Includes Trusted Build activities.
Technical Data Package	TDP	The data package supplied by the vendor, which includes Functional Requirements, Specifications, End-user documentation, Procedures, System Overview, Configuration Management Plan, Quality Assurance Program, and manuals for each of the required hardware, software, firmware components of a voting system.
Test Method	No Abbreviation	SLI proprietary documents which are designed to group sets of EAC VVSG requirements in a logical manner that can be utilized to efficiently validate where and how requirements, or portions of a requirement, are met.
Test Module	No Abbreviation	An actionable component of a Test Method, that functionally verifies that a requirement is met within a voting system. Test Modules are at a generic level within the Test Method, and are customized for a particular voting system, within a Test Suite.
Test Suite	No Abbreviation	An actionable grouping of test modules designed to test a set of functions of a voting system or component in a specific way.
Trusted Platform Module	TPM	A dedicated microcontroller designed to secure hardware through integrated cryptographic keys.
Universal Voting Console	UVC	The UVC features large, color-coded keys labeled with both visible text and Braille characters. The UVC keys enable the voter to adjust the audio volume and tempo, navigate the ballot, make contest selections, open the help screen, and use the blank privacy screen feature.
Voluntary Voting System Guidelines	VVSG	A set of specifications and requirements against which voting systems can be tested to determine if the systems provide all the basic functionality, accessibility and security capabilities required for EAC certification.
Voting System Test Lab	VSTL	An independent testing organization accredited by NVLAP and the EAC to conduct voting system testing for EAC certification.

1.4 System Identification

This section provides a description of the scope of **ES&S EVS 6.0.4.0** voting system and components.

The **ES&S EVS 6.0.4.0** voting system is composed of software applications, central count location devices and polling place devices with accompanying firmware, and COTS hardware and software.

1.4.1 Electionware®

Electionware election management software is an end-to-end election management software application that provides election definition creation, ballot formation, equipment configuration, result consolidation, adjudication and report creation. **Electionware** is composed of five software groups: Define, Design, Deliver, Results and Manage.

1.4.2 ExpressVote XL™

ExpressVote XL Full-Faced Universal Voting System (**ExpressVote XL**) is a hybrid paper-based polling place voting device that provides touch screen vote capture that incorporates the printing of the voter's selections as a cast vote record, and tabulation scanning into a single unit. **ExpressVote XL** is capable of operating in either marker or tabulator mode, depending on the configurable mode that is selected in **Electionware**.

1.4.3 ExpressTouch®

ExpressTouch Electronic Universal Voting System (**ExpressTouch**) is a DRE voting system which supports electronic vote capture for all individuals at the polling place.

1.4.4 ExpressVote® Hardware 1.0

ExpressVote Universal Voting System Hardware 1.0 (**ExpressVote HW1.0**) is a hybrid paper-based polling place voting device that provides touch screen vote capture that incorporates the printing of the voter's selections as a cast vote record, to be scanned for tabulation in any one of the ES&S precinct or central scanners.

1.4.5 ExpressVote® Hardware 2.1

ExpressVote Universal Voting System Hardware 2.1 (**ExpressVote HW2.1**) is a hybrid paper-based polling place voting device that provides touch screen vote capture that incorporates the printing of the voter's selections as a cast vote record, and tabulation scanning into a single unit. **ExpressVote HW2.1** is capable of operating in either marker or tabulator mode, depending on the configurable mode that is selected in Electionware.

1.4.6 DS200®

DS200 is a polling place paper-based voting system, specifically a digital scanner and tabulator that simultaneously scans the front and back of a paper ballot and/or vote summary card in any of four orientations for conversion of voter selection marks to electronic Cast Vote Records (CVR).

1.4.7 DS450®

DS450 is a central scanner and tabulator that simultaneously scans the front and back of a paper ballot and/or vote summary card in any of four orientations for conversion of voter selection marks to electronic Cast Vote Records (CVR).

1.4.8 DS850®

DS850 is a central scanner and tabulator that simultaneously scans the front and back of a paper ballot and/or vote summary card in any of four orientations for conversion of voter selection marks to electronic Cast Vote Records (CVR).

1.4.9 Event Log Service (ELS)

Event Log Service (**ELS**) monitors and logs users' interactions with the Election Management System. Events that happen when a connection to the database is not available are logged to the Windows Operating System log through the **ELS**.

1.4.10 Removable Media Service (RMS)

Removable Media Service (**RMS**) is a utility that runs in the background of the Windows operating system. **RMS** reads specific information from any attached USB devices so that **ES&S** applications such as **Electionware** can use that information for media validation purposes.

1.4.11 Block Diagrams

Figure 1: Voting System Overview – Configuration A

Figure 2: Voting System Overview – Configuration B

Figure 3: Voting System Overview – Configuration C

1.5 Software and Firmware

All software/firmware to be used by the declared voting system whether directly or indirectly, in a production environment, must be validated during the certification process.

The software and firmware employed by **ES&S EVS 6.0.4.0** consists of two types, custom and commercial off the shelf (COTS). COTS applications were verified to be pristine or were subjected to source code review for analysis of any modifications and verification of meeting the pertinent standards.

1.5.1 Manufacturer Software/Firmware

The **ES&S EVS 6.0.4.0** voting system consists of the following software and firmware components:

- **Electionware** Election database creation, media programming and tally/reporting software
- **DS450** Central Count scanner and tabulator, Central Tabulator firmware
- **DS850** Central Count scanner and tabulator, Central Tabulator firmware
- **DS200** Precinct scanner and tabulator, Precinct Tabulator firmware
- **ExpressVote HW1.0** Precinct ballot marker, Universal Voting System firmware
- **ExpressVote HW2.1** Precinct ballot marker and/or Precinct scanner and tabulator, Universal Voting System firmware
- **ExpressVote HW1.0 Previewer** ballot preview software
- **ExpressVote HW2.1 Previewer** ballot preview software
- **ExpressVote XL** Precinct ballot marker and/or Precinct scanner and tabulator, using a full-face touchscreen and Universal Voting System firmware
- **ExpressTouch** DRE, Electronic Universal Voting System firmware
- **Event Log Service (ELS)** software service monitoring user's interactions with the Election Management System
- **Removable Media Service (RMS)** software service supporting election media programming

Table 1 – ES&S EVS 6.0.4.0 Software/Firmware

Application	Version
Electionware – Client/Server	5.0.4.0
Event Log Service	1.6.0.0
Removable Media Service	1.5.1.0
ExpressVote HW1.0	1.5.2.0
ExpressVote HW1.0 Previewer	1.5.2.0
ExpressVote HW2.1	2.4.5.0
ExpressVote HW2.1 Previewer	2.4.5.0
DS200	2.17.4.0

Application	Version
DS200 Ancillary (used to build the DS200, but no resultant output)	6.0.0.0
DS850	3.1.1.0
DS450	3.1.1.0
ExpressVote XL	1.0.3.0
ExpressTouch	1.0.3.0

Note: The Previewer and DS200 Ancillary file(s) are built during the same build as the respective **ExpressVote** and **DS200** application.

1.5.2 COTS Software/Firmware

This section details the COTS software and firmware utilized within the **ES&S EVS 6.0.4.0** voting system.

Table 2 – COTS Software/Firmware

Manufacturer	Application	Version
Microsoft Corporation	Windows 7 Professional	SP-1 (64-bit)
Microsoft Corporation	Windows Server 2008	R2, SP-1 (64-bit)
Microsoft Corporation	Windows 7 Enterprise	SP-1 (64-bit)
Microsoft Corporation	WSUS Microsoft Windows Offline Update Utility	11.5
Symantec	Symantec Endpoint Protection	14.2.0_MP1 (64-bit)
Symantec	Symantec Endpoint Protection Intelligent Updater (File-Based Protection)	20190122-001-core15sdsv5i64.exe
Symantec	Symantec Endpoint Protection Intelligent Updater (Network-Based Protection)	20190121-062-IPS_IU_SEP_14RU1.exe
Symantec	Symantec Endpoint Protection Intelligent Updater (Behavior-Based Protection)	20190115-001-SONAR_IU_SEP.exe
Gigabyte	WindowsImageTool	B17.1116.01
Cerberus	Cerberus FTP Server - Enterprise	10.0.5 (64-bit)
Adobe	Adobe Acrobat Standard	XI
Microsoft Corporation	Visual C++ Redistributable	en_visual_cpp_2015_redistributable_x86_8487157.exe (32-bit)
RSA Security	RSA BSAFE Crypto-C ME for Windows 32-bit	4.1
OpenSSL	OpenSSL	2.0.12
OpenSSL	OpenSSL	2.0.16
OpenSSL	OpenSSL	1.02d
OpenSSL	OpenSSL	1.02h

Manufacturer	Application	Version
OpenSSL	OpenSSL	1.02k

1.6 Equipment

The following equipment is required for the execution of the hardware, software, telecommunications, and security tests. This includes system hardware, general purpose data processing and communications equipment, and any test instrumentation required.

1.6.1 ES&S EVS 6.0.4.0 Equipment

The following manufacturer equipment was used in testing:

Table 3 – ES&S EVS 6.0.4.0 Equipment

Hardware	HW Revision	Model
ExpressVote Universal Voting System	1.0	N/A
ExpressVote Universal Voting System	2.1, 2.1.2.0	2.1.2.0 includes display versions 6.4 and 6.8
DS200 Precinct-based Scanner and Tabulator	1.2, 1.3,1.3.11	N/A
DS450 Central Count Scanner and Tabulator	1.0	N/A
DS850 Central Count Scanner and Tabulator	1.0	N/A
ExpressVote XL Full-Faced Universal Voting System	1.0	N/A
ExpressTouch Electronic Universal Voting System	1.0	N/A
ExpressVote Rolling Kiosk	1.0	98-00049
Quad Express Cart	N/A	41404
ExpressVote Voting Booth	N/A	98-00051
MXB ExpressVote Voting Booth	N/A	95000
ExpressVote Single Table	N/A	87033
ExpressVote Double Table	N/A	87032
ExpressVote ADA Table	N/A	87031
DS200 Collapsible Ballot Box	1.0, 1.1	98-00009
DS200 Plastic Ballot Box	1.2, 1.3, 1.4, 1.5	57521
DS200 Tote Bin	1.0	00074
DS450 Cart	N/A	3002
DS850 Cart	N/A	6823
Universal Voting Console	1.0	98-00077
Tabletop Easel	N/A	14040
ExpressTouch Voting Booth	N/A	98-00081

Hardware	HW Revision	Model
Delkin USB Flash Drive	N/A	Bitlocker 32.2 MB

1.6.2 COTS Equipment

The following COTS equipment was used in testing:

Table 4 – COTS Equipment

Manufacturer	Hardware	Model	Operating System
Innodisk	USB EDC H 2SE (1GB)	DEUH1-01GI72AC1SB (for ExpressVote HW1.0)	N/A
Innodisk	USB EDC H 2SE (16GB)	DEUH1-16GI72AC1SB (for ExpressVote HW2.1)	N/A
Delkin Devices	USB Embedded 2.0 Module (16GB)	MY16TNK7A-RA042-D	N/A
Symbol	Scanner (External)	DS9208	N/A
Zebra Technologies	Scanner (Integrated)	DS457-SR20009 DS457-SR20004ZZWW	N/A
OKI	Audit Printer	Microline 420	N/A
Dell	Report Printer	S2810dn	N/A
OKI	Report Printer	B431DN B431D B432DN	N/A
Tripp Lite	Spike Cube	SPIKECUBE	N/A
APC	Backup power supply (Uninterruptible Power Supply)	Back-UPS Pro 1500 Smart-UPS 1500 Back-UPS RS 1500	N/A
Dell (EMS Standalone configuration)	• Processor: Dual Core • RAM: 4 GB, 8 GB recommended • Hard Disk: 150 GB • Keyboard • Mouse • Monitor: 1280x800 resolution • Monitor – ExpressVote XL	Latitude 5580 Latitude E6430	Windows 7 Professional, SP-1 (64-bit) Windows 7 Enterprise, SP-1 (64-bit)

Manufacturer	Hardware	Model	Operating System
	(Monitor needed for programming election for ExpressVote XL) 1920x1080p resolution • CD/DVD reader: 16x min (internal or external) • 2 USB ports: 2.0 min • Report Printer: w/printer control language driver • Trusted Platform Module (TPM)		
Dell (EMS Networked Server and Client configuration)	• Processor: Dual Core or Quad Core • RAM: 4 GB, 8 GB recommended • Hard Disk: 150 GB or 320 GB • Keyboard • Mouse • Monitor: 1280x800 resolution • Monitor – ExpressVote XL Program Your Own: 1920x1080p resolution • CD/DVD reader: 16x min (internal or external) • 2 USB ports: 2.0 min • Report Printer: Network printer w/printer control language driver • Ethernet Port • Backup power supply: 865 Watts /	OptiPlex 5040 OptiPlex 5050 OptiPlex 7020 PowerEdge T420 PowerEdge T630	Windows Server 2008 R2, SP-1 (64-bit) Windows 7 Professional, SP-1 (64-bit) Windows 7 Enterprise, SP-1 (64-bit)

Manufacturer	Hardware	Model	Operating System
	1500 VA output capacity • Network Switch: 1 GB throughput • Trusted Platform Module (TPM)		
Delkin	USB Flash Drive: 512 MB, 1 GB, 2 GB, 4 GB, 8 GB	N/A	N/A
Delkin	USB Flash Drive: 16 GB (Validation only)	N/A	N/A
AVID	Headphones	86002	N/A
Seiko Instruments	Thermal Printer	LTPD-347B	N/A
NCR / Nashua	Paper Roll	2320	N/A
Delkin	Compact Flash Memory Card: 1 GB max	CE0GTFHHK-FD038-D	N/A
Delkin	Compact Flash Memory Card Reader/Writer	6381	N/A
Delkin	CFAST Card, 2 GB, 4 GB	N/A	N/A
Lexar	CFAST Card Reader/Writer	LRWCR1TBNA	N/A
CardLogix	Smart Card, 16 KB	CLXSU128KC7 / AED C7	N/A
SCM Microsystems	Smart Card Writer	SCR3310	N/A
Fujitsu	Thermal Printer	FTP-62GDSL001 FTP-63GMCL153	N/A
TDS	Ink Cartridge	2278	N/A
HP Inkjet	Ink Cartridge	87002	N/A
Dell	Trusted Platform Module (TPM) chip, version 1.2	R9X21	N/A

1.7 Test Materials

The following test materials are required for the performance of testing including, as applicable, test ballot layout and generation materials, test ballot sheets, test ballot cards and control cards, standard and optional output data report formats, and any other materials used in testing.

- Ballots and blank ballot grade paper

- Activation cards
- Smart cards
- Ballot pens
- Printer paper rolls

1.8 ES&S EVS 6.0.4.0 Documentation

The documents that are a part of the **ES&S EVS 6.0.4.0** voting system are detailed in "Attachment C – ES&S EVS6040 TDP Document List".

1.9 Modifications

The following modifications are part of the **ES&S EVS 6.0.4.0** voting system:

New Hardware Configurations

- The MXB ExpressVote Voting Booth accommodates seated voters on one side and standing voters on the other.
- The Quad Express Cart holds up to four ExpressVote units securely in place. Three of the units are positioned for standing voters while the fourth accommodates a seated voter.

Hardware Modifications

- **DS200** (Revision 1.3.11) updated the following components that were certified with previous versions of the system due to end-of-life:
  - Motherboard
  - Display
  - Touch screen controller and drivers
  - Scanner board motor driver
- **DS200 Collapsible Ballot Box.** (Revision 1.1) introduces better ballot box sidewalls and auxiliary slot for product improvement.
- **DS450** added new uninterruptible power supply (UPS) and report printer as alternatives to accommodate end-of-life component replacement.
- Added a new kiosk barcode scanner due to the old version going end-of-life for the **ExpressVote HW1.0** and **ExpressVote HW2.1**.
- The Delkin USB Embedded 2.0 Module and Compact Flash Memory Card COTS hardware components have been updated.
- A Trusted Platform Module (TPM) chip has been introduced to the system configuration. This chip is required to be added to any EMS workstation that doesn't have an integrated TPM when running Windows Enterprise OS with the Bitlocker feature.
- A Delkin 32.2 MB USB Flash Drive has been added to manufacturer equipment for use with the Bitlocker feature.

Software/Firmware Modifications

- Modification for Windows® 7 Enterprise and Windows® Server 2008 Bitlocker configuration. Bitlocker is a full-volume encryption feature included in select Microsoft operating systems.
- Provided support for the Windows® 7 Enterprise operating system to be used for the **EMS**. This operating system includes BitLocker, which is Microsoft's proprietary disk encryption utility. This operating system also includes the optional dual-factor authentication ability, which is a security enhancement that allows you to present two pieces of evidence when logging in to an account. This modification impacts **Electionware**.
- Introduced Marker mode with front eject only for the **ExpressVote XL**. This modification also affects **Electionware**.
- Removed configuration options that allow a voter to cast a vote without the option of first reviewing the printed card. The voter must be able to choose to Review Card or Cast Vote, rather than only providing the option for Cast Vote. This modification is applicable to **Electionware**, **ExpressVote HW1.0, ExpressVote HW2.1**, and **ExpressVote XL**.
- Updated voter-facing screens to increase focus on the Review Card option before the Cast Vote option. This modification is applicable to **Electionware**, **ExpressVote HW1.0, ExpressVote HW2.1**, and **ExpressVote XL**.
- Added an **Electionware** configuration setting to show or hide the Write-Ins icon, which is used to access the onscreen write-in review feature, on the **DS200** Polls Closed screen.
- Updated the copyright date in the startup splash screen. This modification is applicable to the **DS200**, **DS450**, and the **DS850**.
- Provided support for multi-language audio playback of the write-in keyboard on the **ExpressVote HW1.0**, and **ExpressVote HW2.1**.
- Modified the user interface to properly handle manual candidate selection(s) in a contest after the selections made by the straight party selection are automatically deselected in that contest. This modification applies to the **ExpressVote HW1.0, ExpressVote HW2.1**, and **ExpressVote XL**.
- Modified the **ExpressVote HW1.0** and **ExpressVote HW2.1** firmware/software to accommodate a new version of the kiosk barcode scanner due to the old version going end-of-life.
- Enhanced support for kiosk barcode scanner in "low light" mode for the **ExpressVote HW1.0** and **ExpressVote HW2.1**.
- Modified the **DS200** firmware to accommodate end-of-life component replacement.
- Modified the **DS450** firmware to support an alternative UPS and report printer.
- Modified the **DS850** firmware to support an alternative UPS and report printer. The alternative UPS and report printer are not included in the hardware configuration of the **DS850** for this system and is intended to be used in a future release.
- Removed support in **Electionware** for Adjudication Status Controls (i.e. "Approve Ballot" and "Put On-Hold") for vote summary cards generated from the **ExpressVote HW1.0**, **ExpressVote HW2.1**, and **ExpressVote XL**.

- Enhanced the Reporting Admin Settings in **Electionware** for the Precinct Summary Report to suppress results on a contest-by-contest basis rather than by ballots cast in the precinct.
- Added the ability in **Electionware** to recognize and load media burned from an election restored on any instance of **Electionware** when loading results.
- Updated Users.xml to ensure the most up-to-date version is utilized in **Electionware**.
- Incremented the **ExpressTouch** firmware version to remain synchronized with common code changes in the **ExpressVote XL**.
- Removed **DS200** Status from **ExpressVote HW1.0** System Readiness Report since "tethered mode" will not be supported.
- Removed Ballot Online (BOL) scanner setup functionality from the Maintenance Menu for the **ExpressVote HW1.0**.
- Modified the user interface for the **ExpressVote XL** to properly handle write-in entry for a multiple Vote-For contest.
- Corrected straight party audio inconsistency for the **ExpressVote XL** when changing selection(s).
- Resolved a scenario where the cast button was displayed after the vote session timed out for the **ExpressVote XL**.
- Enhanced the **ExpressVote XL** user interface to display a warning corresponding to the media door being opened.
- Added the ability to automatically print a test deck from tabulation mode for the **ExpressVote XL**. Test deck cards include the word "Test" when printed.
- Added a configuration setting to the **ExpressVote XL** to control on screen selection presentation. Options include show the borders around the selection checkmark, show only the selection checkmark or show nothing.
- Improved latest version of **RMS** to ensure media packaging remains consistent.
- Changes have been made to all COTS software components as listed in Table 2.
- Updates made to SecureSetup.exe hardening script.

2 Certification Test Background

This section provides a brief overview of the EAC Certification Program and the activities involved for a voting system to be considered for certification against the EAC VVSG and the EAC program manual.

2.1 System Revision History

ES&S EVS 6.0.4.0 is a modification of the **ES&S EVS 6.0.2.0** voting system, certified by the EAC on October 4th, 2018. Specific engineering changes are listed in section 1.8 of this report.

2.2 Implementation Statement

The **ES&S EVS 6.0.4.0** voting system incorporates all software and hardware, as well as supporting documentation, as declared in the **ES&S**'s implementation statement, as

provided to the EAC. Please refer to "Attachment B – ES&S EVS6040 Implementation Statement".

2.3 PCA - Document and Source Code Reviews

The Physical Configuration Audit (PCA) review of the **ES&S EVS 6.0.4.0** voting system documentation, submitted in the requisite Technical Data Package (TDP), was performed in order to verify conformance with the Election Assistance Commission Voluntary Voting System Guidelines 1.0 (EAC VVSG 1.0). Source code was reviewed for each software and firmware application modified from **ES&S EVS 6.0.2.0** for the **ES&S EVS 6.0.4.0** voting system. Source code was not reviewed for firmware or software applications that were not modified for **ES&S EVS 6.0.4.0.**

All PCA reviews were conducted in accordance with *Volume 2 Section 2* of the EAC VVSG 1.0, to demonstrate that the system meets the requirements. Results of the PCA documentation review can be found in section 3.2 of this Certification Test Report. Inconsistencies or errors in documentation were identified to **ES&S** for resolution or comment.

All PCA source code reviews were conducted in accordance with *Volume 1 Section 5.2 and Volume 2 Section 5* of the EAC VVSG 1.0, to demonstrate that the system meets the requirements. Results of the PCA source code reviews can be found in section 3.1 of this Certification Test Report.

2.4 FCA - Functional & System Testing

The Functional Configuration Audit (FCA) review of the test documentation submitted by **ES&S** in the TDP was executed to verify testing of the voting system requirements defined in *Volume 1 Sections 2, 6, 7,* and *9* of the EAC VVSG 1.0. Changes made to the TDP from **ES&S EVS 6.0.2.0** to the TDP submitted with **ES&S EVS 6.0.4.0** were reviewed in detail. In addition, the **ES&S** System Development program was reviewed to ensure processes and procedures are properly documented and in agreement with observations by the VSTL through the duration of this test campaign.

SLI's standard Test Suites were customized for the **ES&S EVS 6.0.4.0** voting system and conducted in accordance with *Volume 2 Section 6,* in conjunction with the source code review, TDP review, integration, accuracy, security, modification, and regression tests. Simulations of elections were conducted to demonstrate a beginning-to-end business use case process for the **ES&S EVS 6.0.4.0** voting system.

2.4.1 Test Methods

All test methods employed are within the scope of SLI's VSTL accreditation. The following validated test methods were employed during this test campaign:

Table 6 – Test Methods

SLI VSTL Test Method Name
TM_Accuracy v1.1

SLI VSTL Test Method Name
TM_Basic_Election_Components v1.0
TM_Tally_and_Reporting v1.0
TM_Ballot_Counter v1.1
TM_Accumulating_and_Transmitting_Results v1.1
TM_Pre-Voting_Capabilities v1.2
TM_Voting_Capabilities v1.3
TM_Closing_the_Polls v 1.1
TM_Security_Access_Control v1.1
TM_Security_Software_Security v1.1
TM_Security_Physical_Security_Measures v1.1
TM_Voting_Vote_for_N_of_M v1.1
TM_Voting_Straight_Party v1.2

The above listed test methods are implemented in a complementary fashion: modules are employed from various methods to form suites. Suites include a logical sequence of functionality that is used to validate the requirement addressed by each module within the suite. Please see the Terms and Abbreviations table for additional information about Test Modules and Test Suites.

2.4.2 3rd Party Hardware Testing

Hardware testing was conducted by 3rd Party certified hardware test laboratories to verify the voting system devices that have been modified in the **ES&S EVS 6.0.4.0** voting system are in compliance with the EAC VVSG 1.0 hardware requirements.

SLI Compliance is responsible for all core voting system tests as identified in the NIST NVLAP Handbook 150-22 (2017). Regarding non-core hardware testing for this certification test campaign, this report contains data that were produced under subcontract by the following lab(s):

Table 3 – Labs Performing Hardware Testing

Laboratory	Address	Test(s)	Date(s)
NTS – EMI / EMC	1736 Vista View Drive Longmont, CO 80504	**EMC / EMI Tests:** Radiated Emissions, Conducted Emissions, ESD, Electromagnetic Susceptibility, Electrical Fast Transient, Lightning Surge, Conducted RF Immunity, Magnetic Fields Immunity, Electrical Power Disturbance	10/17/2018 – 10/23/2018 11/27/2018 1/7/19 – 1/14/19

Laboratory	Address	Test(s)	Date(s)
NTS – Environmental / Dynamic	1601 Dry Creek Drive Suite 200 Longmont, CO 80503	**MIL-STD-810D Tests:** Bench Handling, Vibration, Low Temperature, High Temperature, Humidity, Temperature/Power Variation	10/16/2018 – 10/31/2018 12/11/18 – 12/14/18 1/8/19 – 1/11/19

3 Certification Test Results Summary

3.1 Source Code Review Summary

SLI has reviewed the modified software source code for each application in the **ES&S EVS 6.0.4.0** voting system to determine the code's compliance with the EAC VVSG 1.0, *Volume 1 Sections 5, 9* and *Volume 2 Section 5.4* and for compliance with **ES&S'**s internally developed coding standards. **ES&S EVS 6.0.4.0** is implemented with the C, C++, C#, Java, VB and VB.net languages.

Evaluation of Source Code

No discrepancies were observed during the review of delivered source code. All code has successfully passed all reviews.

3.2 Technical Data Package Review Summary

SLI reviewed the **ES&S EVS 6.0.4.0** TDP for compliance with the EAC VVSG 1.0 according to *Volume 2 Section 2*. The documents that are a part of the **ES&S EVS 6.0.4.0** voting system are detailed in "Attachment C – ES&S EVS6040 TDP Document List".

Evaluation of TDP

Twenty four documentation issues were written during the PCA and FCA documentation review phases. The issues were related to either incorrect or missing information. Details can be found in "Attachment H – ES&S EVS6040 Discrepancy Report".

In all instances, the issues were addressed and resolved with updated documentation prior to the writing of this report. Once all identified discrepancies were resolved, the Technical Data Package for the **ES&S EVS 6.0.4.0** voting system was found to comply with all applicable standards.

3.3 Hardware Testing Summary

SLI and their third-party certified hardware test laboratory, National Technical Systems (NTS), performed an analysis and review of the modified **ES&S EVS 6.0.4.0** voting system hardware components. During execution of testing performed at NTS, an SLI representative was present to oversee the testing.

The test methodologies for all tests are identified in the hardware test plans and hardware test reports, as listed in section 1.1 "Certification Test Report Attachments".

SLI also conducted a review on the **ES&S EVS 6.0.4.0** Safety Report, no issues were found, all components were found to be compliant according to EAC VVSG 1.0 *Volume 1 Section 4.3.8.*

Hardware testing conducted specifically for this test campaign involved the **DS200** and **DS450**. The testing consisted of:

- Electromagnetic Emissions / Immunity Tests:
    - Radiated Emissions – FCC, Part 15 Class B ANSI C63.4.
    - Conducted Emissions – FCC, Part 15 Class B ANSI C63.4.
    - ESD – IEC 61000-4-2 (2008) Ed. 2.0.
    - Electromagnetic Susceptibility – IEC 61000-4-3 (1996).
    - Electrical Fast Transient – IEC 61000-4-4 (2004-07) Ed. 2.0.
    - Lightning Surge – IEC 61000-4-5 (1995-02).
    - Conducted RF Immunity – IEC 61000-4-6 (1996-04).
    - Magnetic Fields Immunity – IEC 61000-4-8 (1993-06).
    - Electrical Power Disturbance – IEC 61000-4-11 (1996-06).
- Non-Operating Environmental Tests:
    - Bench Handling - MIL-STD-810D, Method 516.3, Procedure VI
    - Vibration - MIL-STD-810D, Method 514.3, Category 1- Basic Transportation, Common Carrier.
    - Low Temperature - MIL-STD-810D, Methods 502.2, Procedure I-Storage.
    - High Temperature - MIL-STD-810D, Methods 501.2, Procedure I-Storage.
    - Humidity (85%) Soak - MIL-STD-810D, Method 507.2, Procedure I-Natural Hot-Humid.
- Operating Environmental Tests:
    - Temperature/Power Variation - similar to the low temperature and high temperature tests of MIL-STD-810-D, Method 502.2 and Method 501.2.
    - Reliability – Vol. 1, Section 4 for the acceptable Mean Time Between Failure (MBTF).

Evaluation of Hardware Testing

As this test campaign was a modification of an EAC certified voting system, only modified hardware components of the **ES&S EVS 6.0.4.0** voting system were evaluated against applicable hardware requirements.

Seven hardware discrepancies were written during this test campaign for issues encountered during hardware testing. Details can be found in "Attachment H – ES&S EVS6040 Discrepancy Report". **ES&S** appropriately resolved each issue and subsequently passed all hardware tests.

3.4 Functional Testing Summary

SLI performed tests designed to functionally verify the modifications listed in section 1.8 of this report. The testing incorporated end-to-end election scenarios testing the functionality supported by **ES&S**. The following sections detail the test suites that were executed.

3.4.1 Accuracy Test Suite

Accuracy testing was performed to verify the ability of the system to capture, record, store, consolidate and report the specific selections and absence of selections, made by the voter for each ballot position, without error. Additionally, the system was exercised to validate that the accumulation, tallying and reporting mechanisms at the system level accurately perform their functions.

Accuracy testing was conducted at both the device level and the system level. The **ExpressVote HW1.0, ExpressVote HW2.1,** and **ExpressVote XL** were utilized to mark and print vote summary cards, which were then scanned into the **DS200, DS450,** and **DS850** devices. Vote counts were tabulated from **DS450, DS850, ExpressVote HW2.1, ExpressTouch, ExpressVote XL,** and **DS200**. Results were processed through **Electionware** and examined for completeness and correctness.

3.4.2 Integration Test Suite

An Integration Test Suite designed to verify proper integration of system components was conducted using General and Open Primary elections. These elections were utilized multiple times to ensure each EMS operating system, standalone, and server-client configuration was covered.

- The General election definition focused on N of M voting, Partisan offices, Non-Partisan Offices, precincts and districts, write-ins, tally, results loading, and reporting functionality.
- The Open Primary election definition utilized three partisan parties.

Integration testing was conducted at both the device level and the system level. The **ExpressVote HW1.0**, **ExpressVote HW2.1,** and **ExpressVote XL** were utilized to mark and print vote summary cards, which were then scanned into the **DS200, DS450,** and **DS850** devices. Vote counts were tabulated from **DS450, DS850, ExpressVote HW2.1, ExpressTouch, ExpressVote XL,** and **DS200**. Results were processed through **Electionware** and examined for completeness and correctness.

3.4.3 Modification Test Suite

Modification test cases were executed to focus on the specific changes incorporated into the system at the device level, and in conjunction with the **EMS** as applicable. Various elections were used to exercise the **EMS** and devices such that each specific modification was functionally verified, with an appropriate quantity of regression testing performed as determined by analysis of the modifications.

3.4.4 Pennsylvania Straight Party Method Test Suite

A General Election designed to test all variations of the Pennsylvania Straight Party Method was performed to ensure all modifications and performance related enhancements are working correctly as documented, and in accordance with the VVSG 1.0 requirements.

3.4.5 Security Test Suite

A Security Test Suite was designed and executed to examine all the various security related modifications to the system configuration. Included in the assessment was a comprehensive examination of the in-place physical security mitigation measures of the new **DS200** collapsible ballot box. The physical examination comprised of verification of security seal placement and the ability to quickly detect tampering. Attempts to easily bypass ballot box design to introduce untallied ballots or access internal contents of the ballot box were included in the physical examination.

The Security examination also included review of a new version of the endpoint protection software. Utilizing industry defined Anti-virus and Malware test definitions, alongside vulnerability assessment to determine if the machines were susceptible to malicious software and network attacks.

A review of a new EMS configuration Windows® 7 Enterprise operating system, including BitLocker drive encryption, was performed. This configuration was examined to determine if BitLocker drive encryption was successfully implemented utilizing a TPM 1.2 chip and required security keys. In addition, testing confirmed that the BitLocker drive encryption was properly configured and that all storage drives were successfully encrypted utilizing AES-256-bit encryption.

Evaluation of Functional Testing

In this test campaign, **ES&S EVS 6.0.4.0** was subjected to examination for changes, updates, and modifications made from the previously certified system, **ES&S EVS 6.0.2.0**, against applicable requirements within the EAC VVSG 1.0.

Through the duration of testing, six functional issues were written. Details can be found in "Attachment H – ES&S EVS6040 Discrepancy Report". Issues found were reported, resolved, and re-tested as applicable. Once all discrepancies have been addressed, no violation of conformance to EAC VVSG 1.0 requirements was observed. All components of the **ES&S EVS 6.0.4.0** voting system have successfully passed all tests.

4 Recommendations

SLI has successfully completed the testing of the **ES&S EVS 6.0.4.0** voting system. It has been determined that the **ES&S EVS 6.0.4.0** voting system meets the required acceptance criteria of the Election Assistance Commission Voluntary Voting System Guidelines, version 1.0.

This recommendation reflects the opinion of SLI Compliance based on testing scope and results. It is SLI's recommendation based on this testing effort that the EAC grant certification of the **ES&S EVS 6.0.4.0** voting system.

5 APPROVAL SIGNATURES

SLI:

Traci Mapps
VSTL Director
April 26th, 2019

6 Appendix A – Ancillary Products

Ancillary systems represent products and utilities that are not part of the EAC certified configuration, however, they may be used to facilitate testing.

Ancillary systems include:

- Ballot Production
  - **Balotar** is a product that receives ballot artwork PDFs and ballot on demand files from **Electionware**. **Balotar** is specifically designed to automatically generate and print ad hoc ballots.
- Ranked Choice Voting
  - **ExpressRunoff** is a software utility for automating ranked choice voting for single-seat contests. Ranked Choice Voting (RCV) is an electoral system used to elect a single winner from a field of more than two candidates, where voters rank the candidates in order of preference. After results have been loaded into **Electionware**, **ExpressRunoff** uses the Cast Vote Record (CVR) data exported from **Electionware** to create RCV rounds, and produces customizable reports showing the outcome of each round.
- Electronic Pollbook
  - **ExpressPoll** electronic pollbook stores registered voter information for precincts, districts, or entire jurisdictions. The voter registration data can be shared with the **ExpressLink** application to print a voter's activation card for use in an **ExpressVote** or **ExpressVote XL**.
- **ExpressLink** System
  - **ExpressLink** is a Windows PC application that can run in either a standalone mode, or in a monitor mode, where the application monitors requests from a voter registration (VR) system over a shared network folder. The application imports an election definition from **Electionware**, accepts requests to print a voter's activation card for use in an **ExpressVote** or **ExpressVote XL**, determines the voter's ballot style and then prints the activation card on the **ExpressVote Activation Card Printer**. Separately, this application is used to program vote session activator cards for use with **ExpressTouch**.

- **ExpressVote Activation Card Printer**, a thermal, on demand printer, is used to print the ballot activation code on the activation card for use with **ExpressVote** or **ExpressVote XL**.
- **ExpressTouch Smart Card Writer** is a device used to program the ballot activation code on the **ExpressTouch** vote session activator card.

- **Electionware Toolbox** is a set of utilities that can be integrated into the **Electionware** EMS to enhance the software usability experience and streamline various processes. These add-on utilities include **Test Deck**, **Text to Speech** and **Media Restore**.
  - **Test Deck** provides a means for the election official to test the election on each machine that will be used for voting. Vote patterns can be created with automatic ballot marking, and then the ballots can be printed and scanned through the **ES&S** ballot tabulators to test logic and accuracy of the counting. Additionally, a test pattern file can be created for the **ExpressTouch**, **ExpressVote** or **ExpressVote XL** that allows automated logic and accuracy testing on the universal voting machine.
  - **Text to Speech** provides a simplified method for creating the audio wave files that make up the audible ballot.
  - **Media Restore** is used to prepare ES&S-certified USB media flash drives for use with **Electionware** by securely clearing all data and then restoring to the FAT32 format.

End of Certification Test Report

EXHIBIT E

6705 Odyssey Drive
Suite C
Huntsville, AL 35806
Phone (256)713-1111
Fax (256)713-1112

Test Report for EAC 2005 VVSG Certification Testing
Unisyn Voting Solutions OpenElect 2.1 Voting System

EAC Project Number: UNS10121966-2.1

Version: 02

Date: 10/17/2019

EAC Lab Code 1501

NVLAP LAB CODE 200908-0

SIGNATURES

Approved by: _____ 10/17/19
Michael Walker, VSTL Project Manager — Date

Approved by: _____ 10/17/19
Wendy Owens, VSTL Program Manager — Date

REVISIONS

Revision	Description	Date
00	Initial Release	10/01/2019
01	Update version number in Table 3-1	10/08/2019
02	Added model numbers to Table 4-4	10/17/2019

TABLE OF CONTENTS

1.0 INTRODUCTION 1
1.1 Description and Overview of EAC Certified System Being Modified 1
1.1.1 Details of System Tested 8
1.2 References 10
1.3 Terms and Abbreviations 11
2.0 CERTIFICATION TEST BACKGROUND 12
2.1 Revision History 12
2.2 Scope of Testing 13
2.2.1 Modification Overview 13
2.2.1.1 Detailed List of Changes 13
2.2.2 Block Diagram 15
2.2.3 System Limits 16
2.2.4 Supported Languages 18
2.2.5 Supported Functionality 18
2.2.6 VVSG 19
2.2.7 RFIs 19
2.2.8 NOCs 19
3.0 TEST FINDINGS AND RECOMMENDATION 19
3.1 Summary of Findings and Recommendation 20
3.1.1 Source Code Review, Compliance Build, Trusted Build, and Document Review 22
3.1.2 System Level Testing 22
3.1.3 Security Testing 24
3.1.4 Usability and Accessibility Testing 24
3.1.5 Security Testing 24
3.2 Anomalies and Resolutions 25
3.3 Correction of Deficiencies 25
4.0 RECOMMENDATION FOR CERTIFICATION 25

ATTACHMENT A - AS-RUN TEST PLAN 26

1.0 INTRODUCTION

The purpose of this Test Report is to document the procedures that Pro V&V, Inc. followed to perform certification testing during a system modification campaign for the Unisyn Voting Solutions OpenElect Voting System (OVS) Version 2.1 to the requirements set forth for voting systems in the U.S. Election Assistance Commission (EAC) 2005 Voluntary Voting System Guidelines (VVSG), Version 1.0. Certification testing of the OVS Version 2.1 Voting System submitted for evaluation was performed to ensure the applicable requirements of the EAC 2005 VVSG and the EAC Testing and Certification Program Manual, Version 2.0, were met. Additionally, all EAC Request for Interpretations (RFI) and Notices of Clarification (NOC) relevant to the system under test were incorporated in the test campaign.

Prior to submitting the voting system for testing, Unisyn submitted an application package to the EAC for certification of the OpenElect 2.1 Voting System. The application was accepted by the EAC and the project was assigned the unique Project Number of UNS10121966-2.1.

1.1 Description and Overview of EAC Certified System Being Modified

The EAC Certified System that is the baseline for the submitted modification is described in the following subsections. All information presented was derived from the previous Certification Test Report, the EAC Certificate of Conformance and/or the System Overview.

The OVS 2.0.A is a paper ballot voting system using touch screen and scan technology to scan and validate ballots, provide voter assisted ballots to accommodate voters with special needs, and tabulate results. The OCS, FVT, OVO, OVI-VC, and OVCS components of the OVS 2.0.A have previously been tested as part of the OVS 2.0 test campaign. The OVS 2.0.A consists of the following major components:

- **OpenElect Central Suite (OCS)**
- **OpenElect Voting Optical (OVO)**
- **OpenElect Voting Interface – Vote Center (OVI-VC)**
- **FreedomVote Tablet (FVT)**
- **OpenElect Voting Central Scan (OVCS)**

The OVO, FVT, and OVI-VC are the OVS components designed to accept voter input. The FVT and OVI-VC are the OVS ballot marking devices (BMDs). The OVO can collect and tally precinct votes, generate reports, and store election data internally. The OVCS is the bulk scanner used for mail-in ballots, provisional ballots, and recounts.

<u>Open Elect Central Suite (OCS)</u>

The OCS System supports elections on the OVO, OVI-VC, FVT, and OVCS systems. The Election Management System (EMS) consists of the following components running as either a front-end/client application or as a back-end/server application:

Ballot Layout Manager (BLM) – uses a database to create and store precinct and district information and an interface to create, check, translate, and produce the ballot styles needed by a jurisdiction for an election. The BLM output is printer ready artwork of all ballots in all languages and the Unisyn election definition file.

Election Manager (EM) – converts the Unisyn election definition file to a Unisyn-specific XML format and prepares compressed, encrypted election files for output to CD or USB. The EM allows the jurisdiction to add voting device specific options for elections, i.e. whether to check the contests for undervotes, and whether to allow or disallow certain features such as sounds, party icons, reports, etc. The EM also creates and manages Supervisor and Maintenance technician logins and passwords.

Tabulator Client (TC) – retrieves vote files and ballot images from a Transport Media (TM) device (USB), stores them on its disk, and transfers the files to the Tabulator and notifies the Tabulator that a new file is present.

Tabulator (Tab) – receives and validates uploaded voting data and provides a status of uploaded files as well as handling Rank Choice Voting (RCV) functionality. It also updates the database with adjudicated ballots from the Auditor application. The Tabulator maintains the Tabulator database, which stores the results from all precincts.

Auditor – accesses ballot images and data from the OVCS and TC PCs to allow jurisdiction personnel to evaluate ballots with questionable or erroneous marks and change votes in accordance to the voter's perceived intent. The Auditor can also be used to process write-in votes. All changes are uploaded to the Tabulator database and actions are password controlled.

Tabulator Reports (TR) – accesses data from the Tabulator database to generate the necessary reports.

OpenElect Voting Optical (OVO)

The OVO is a full-page dual-sided optical scan precinct scanner that scans and validates voter ballot pages and provides a summary of all ballot pages cast. The election is loaded from an Election TM. On Election Day, an OVO at each poll location scans and validates voters' ballots, and provides precinct tabulation and reporting. The OVO runs Logic Tests and Training Elections in addition to General and Primary Elections. The OVO unit can also paired with OVI-VC and FVT units for early voting to scan and tabulate early voting ballots and election support at voting centers.

OVO units can be used at election headquarters to read absentee, provisional or recount ballots in smaller jurisdictions.

The OVO consists of the following components:

- **Personal Computer (PC) -** Computer component (with a touch panel display) has the OVO application installed that manages data and provides a user interface for voting and maintenance.

A new election loaded via a Transport Media (TM) sets passwords, parameters, and ballot styles for that election. (Valid ballots for a poll location are reinitialized or set on Election Day startup by scanning a ballot header card).

- **Transport Media (TM)** – 1 GB or larger USB thumb drive that provides the means of transporting audit, optional ballot page images and results files from the precinct on Election Night to Election Headquarters where the central count system resides.
- **Ballot Reader -** Dual-sided scanner connected to the PC to scan data from marked ballot pages. The Ballot Reader ejects accepted ballot pages into an attached ballot box or rejects unaccepted ballot pages back out to the voter.
- **Printer -** 58 mm thermal receipt printer connected to the PC to print voter receipts and reports at the OVO.
- **UPS** - Uninterruptible power supply is provided as part of the system.

Figure 1-1. OVO

<u>OpenElect Voting Interface – Vote Center (OVI-VC)</u>

The OVI-VC is a ballot marking device (BMD) that supports both ADA and Regional Early Voting requirements. The OVI-VC has a 15-inch display and is equipped to assist voters, with varying abilities, to prepare their ballots independently and privately. It presents each contest on the correct ballot to the voter in visual and, if needed, audio formats. The OVI-VC gives voters the option to use an audio ballot, ADA keypad, sip and puff binary device, and touchscreen with varying font sizes to make their selections.

Once the voter has made their selections, they are prompted to review their ballot selections. After the validation process is complete, the OVI-VC then prints the ballot and the voter inserts it into the OVO to cast their vote. When authorized, the OVI-VC provides for write-in candidates.

Each OVI-VC can support multiple languages for both visual and audio ballots, allowing the voter to choose their preferred language.

The OVI-VC consists of the following components:

- **Personal Computer (PC)** - Computer component (with a touch panel display) has the OVO application installed that manages data and provides user interfaces for voting and maintenance. A new election loaded via a Transport Media (TM) sets passwords, parameters, audio, and ballot styles for that election.
- **Transport Media (TM)** - USB device with 1 GB or larger storage provides the means of transporting audit files to the OCS system.
- **Printer** – 82.5 mm thermal receipt printer is connected to the PC to print BMD Ballots and reports at the OVI-VC.
- **UPS** - Uninterruptible power supply is provided as part of the system.

Figure 1-2. OVI-VC

Freedom-Vote Tablet (FVT)

The FVT is a tablet ballot marking device that enables voters make their vote selections and to print their voted ballot. It can be used on Election Day or during an early voting period. Like the OVI-VC, the FVT is ADA compliant. It assists voters, with varying levels of ability, through the voting process, ballot review, and printing functions. The FVT presents each contest on the ballot style to the voter in visual and/or audio formats. It facilitates special needs voters through a variety of methods including wheelchair access, sip and puff, zoom-in ballot function and audio assistance for the visually impaired. The voter with limited vision can navigate through the ballot using an audio ballot and the ADA keypad or touchscreen to input their selections. Once the ballot is printed, it is taken to the OVO to be cast. Each FVT can support multiple languages for both visual and audio ballots, allowing the voter to choose their preferred language.

The FVT consists of the following components:

- **Tablet** – The Android tablet has a 13.3 in. touchscreen and comes with pre-installed software that provides user interfaces for voting and maintenance. Election files generated by the EM are loaded manually via a USB. The election files will allow the jurisdiction to determine the FVT's mode such as early voting or training, sets passwords, parameters, audio, and ballot styles for that election.
- **Barcode Reader -** 2D USB Barcode reader will read the 2D barcodes produced by the EM such as the initialize barcode and administrative/maintenance barcodes. It will also read the 'populate' barcode produced by other qualified systems.
- **USB Hub –** A four port USB hub is installed in the FVT case to connect the printer, barcode scanner, and keypad to the tablet.
- **Printer –** 82.5 mm thermal receipt printer is connected to the Tablet to print BMD ballots and reports.
- **Optional ADA Devices** – 10-key keypad with Sip and Puff Interface, Headphones, Sip and Puff Device.

Figure 1-3. FVT

<u>OpenElect Voting Central Scan (OVCS)</u>

The OVCS units reside at election headquarters designated to read absentee, provisional or recount ballots in large jurisdictions or read the entire election's ballots at a central count location in smaller jurisdictions. The OVCS also captures Write-in data images and produces a Write-in image report for manual processing upon request. The OVCS system consists of the following components:

- **PC Desktop** – A desktop PC configuration with the following minimum characteristics:
  - PC: 1.8 GHz Processor, 2 GB RAM, 250GB (or larger) Hard Drive, USB Ports, Network Interface Port (Ethernet), CDRW/DVD, Video Port
  - 16:9 LCD
  - Keyboard and Mouse
- **Bulk Scanner** – A dual-sided scanner (either Canon model DR-M160II or model DR-X10C) that is connected to the PC to scan data from marked ballots.

Figure 1-4. OVCS

1.1.1 Baseline Certified System

The baseline system for this modification is the OVS 2.0.A Voting System. The tables below describe the certified equipment and firmware versions. Detailed descriptions of the OVS 2.0.A test campaign are contained in Pro V&V Report No.TR-01-01-UNI-003-01.01 Rev. A, which is available for viewing on the EAC's website at www.eac.gov.

This subsection lists the proprietary and COTS software to be provided by the manufacturer as part of the test campaign.

Table 1-1. OVS 2.0.A COTS Software Components

FVT, OVO and OVI-VC Device Software	Version
CentOS Linux (OVO and OVI-VC)	6.3
Java JRE + Unlimited Cryptographic Extension	1.6.0_02
Android OS (FVT)	4.4.4

Table 1-2. OCS and OVCS COTS Software Components

OCS and OVCS Device Software	Version
CentOS Linux	6.5 and 6.8
Java JRE + Unlimited Cryptographic Extension	1.6.0_02
Apache-Tomcat Application Server	6.0.13
MySQL Database (BLM, EM, A, and Tab only)	5.0.45-7

Table 1-2. OCS and OVCS COTS Software Components *(continued)*

OCS and OVCS Device Software	Version
JasperReports	2.0.5
OpenVPN	2.4.4
OpenSSL	1.0.1f
OpenSSL FIPS Object Module	2.0.10 (cert #1747)

Table 1-3. Voting System COTS Hardware

Hardware	Make	Model
OVO		
Duplex Ballot Scanner		
Duplex Ballot Scanner	PDI Scan	Pagescan III
Scanner Power Adapter	eUrasia Power	uA36-1024
58 mm Thermal Printer		
58 mm Thermal Printer	Citizen Printer	CT-5281
Printer Power Adapter	Citizen Printer	28AD4
Computer		
Chassis	Morex	Morex 2699
DC/DC converter	Morex	MX-0608F
Chassis Fans	Young Lin Tech	DFB404012M
Motherboard	Jetway	JNF9D-2550
Memory	SuperTalent - Onboard RAM	W1333SA2GV
Hard Drive	Western Digital	WD5000AZLX
AC Adapter	EDAC	EA 10951c-120
1 Gb USB TM		
Innodisk	1 Gb USB	DEUA1-01G172AC1SB-B088
Delkin	1 Gb USB	UY0GTFLSY-XN000-D
7” LCD Touchscreen Display		
7” LCD Touchscreen Display	Xenarc Technologies	700TSV
AC Power In Module		
AC Power In Module	Delta	Emi 10BEEG3G
OVI-VC		
Sip and Puff (Optional)		
Sip and Puff (Optional)	Origin Instruments	AirVoter
Headphone (Optional)		
Headphone (Optional)	Koss On-Ear Headphones	KPH5
15 in LCD Touchscreen Display		
15 in LCD Touchscreen Display	GVision	P15BX-OB-4690
82.5 mm Thermal Printer		
82.5 mm Thermal Printer	Star	TSP743IID-24, serial interface
Printer Adapter	Star	PS60A-24B 1
Computer		
Power Adapter Kit	DC-DC Converter	MX-0608F

Table 1-3. Voting System COTS Hardware *(continued)*

Hardware	Make	Model
Motherboard	Jetway	JNF9D-2550
Hard Drive	Western Digital	WD5000AZLX
Adapter	EDAC	EA 10951c-120
1 Gb USB TM		
Innodisk	1 Gb USB	DEUA1-01G172AC1SB-B088
Delkin	1 Gb USB	UY0GTFLSY-XN000-D
AC Power In Module		
AC Power In Module	Delta	Emi 10BEEG3G
OVCS		
Large Volume Scanner		
Large Volume Scanner	Canon	DR-X10C
Desktop Scanner		
Desktop Scanner	Canon	DR-M160II
FVT		
13.3 in Touchscreen Tablet		
13.3 in Touchscreen Tablet	Android Tablet	GVision - T13
Tablets Battery Charger	Sager Power System	GC30B-4P1J
82.5 mm Thermal Printer		
82.5 mm Thermal Printer	Star	TSP743IIU-24
Printer Adapter	Star	PS60A-24B 1
Barcode Reader 1D,2D series		
Barcode Reader 1D,2D series	Newland	FM420
USB Hub		
USB Hub	D-Link	DUB-H4
Hub Adapter	Meanwell	PSD-15A-05
1 Gb USB TM		
Innodisk	1 Gb USB	DEUA1-01G172AC1SB-B088
Delkin	1 Gb USB	UY0GTFLSY-XN000-D
Micro SD		
Micro SD	San Disk	4 GB Edge
Battery		
Battery Pack 12 V 6.0 AH	Powers Sonic	PSH-1255-FR
Adapter	Mean Well	GC30B-4PIL
AC Power In Module		
AC Power In Module	Delta	Emi 10BEEG3G
Sip and Puff (Optional)		
Sip and Puff	Origin Instruments	AirVoter
Headphone (Optional)		
Headphone	Koss On-Ear Headphones	KPH5
Headphone	Sony	MDR-210LP
USB to Ethernet RJ45 Adapter (Optional)		
USB to Ethernet RJ45 Adapter	D-Link	DUB-E100

Table 1-3. Voting System COTS Hardware *(continued)*

Hardware	Make	Model
UPS		
Minuteman Power Technologies	Para Systems, Inc.	Entrepid Series

Table 1-4. OVCS System COTS Software Components

OVS Hardware	Version
Desktop for non-redundant solutions	Dell OptiPlex
Desktop for redundant solutions	Dell Precision
Canon Scanner (OVCS)	Canon DR-X10C or DR-M160II
Laptop	Dell Latitude

1.2 References

- Election Assistance Commission 2005 Voluntary Voting System Guidelines (VVSG) Version 1.0, Volume I, "Voting System Performance Guidelines", and Volume II, "National Certification Testing Guidelines"
- Election Assistance Commission Testing and Certification Program Manual, Version 2.0
- Election Assistance Commission Voting System Test Laboratory Program Manual, Version 2.0
- National Voluntary Laboratory Accreditation Program NIST Handbook 150, 2016 Edition, "NVLAP Procedures and General Requirements (NIST HB 150-2016)", dated July 2016
- National Voluntary Laboratory Accreditation Program NIST Handbook 150-22, 2008 Edition, "Voting System Testing (NIST Handbook 150-22)", dated May 2008
- United States 107th Congress Help America Vote Act (HAVA) of 2002 (Public Law 107-252), dated October 2002
- Pro V&V, Inc. Quality Assurance Manual, Revision 7.0
- Election Assistance Commission "Approval of Voting System Testing Application Package" letter dated May 15, 2019
- EAC Requests for Interpretation (RFI) (listed on www.eac.gov)
- EAC Notices of Clarification (NOC) (listed on www.eac.gov)
- Pro V&V Test Report No. TR-01-01-UNI-003-01.01 Final, "Test Report for EAC 2005 VVSG Certification Testing Unisyn Voting Solutions OpenElect 2.0.A Voting System"
- OpenElect Voting System Release Notes, System 2.0.A to 2.1, Version 1.2

- Unisyn Voting Solutions Technical Data Package *(A listing of the OpenElect 2.1 documents submitted for this test campaign is listed in Section 4.6 of this Test Plan)*

1.3 Terms and Abbreviations

This subsection lists terms and abbreviations relevant to the hardware, the software, or this Test Plan.

"ADA" – Americans with Disabilities Act 1990

"BLM" – Ballot Layout Manager

"BMD" – Ballot Marking Device

"CM" – Configuration Management

"COTS" – Commercial Off-The-Shelf

"DRE" – Direct Record Electronic

"EAC" – United States Election Assistance Commission

"EM" – Election Manager

"EMS" – Election Management System

"EOS" - Election Operating System

"ES" – Election Server

"FCA" – Functional Configuration Audit

"FVT" – FreedomVote Tablet

"LAT" – Logic and Accuracy Test

"NOC" – Notice of Clarification

"OCS" – OpenElect Central Suite

"OVCS" – OpenElect Voting Central Scan

"OVI-VC" – OpenElect Voting Interface – Vote Center

"OVO" – OpenElect Voting Optical

"OVS" – OpenElect Voting System

"PC" – Personal Computer

"PCA" – Physical Configuration Audit

"QA" – Quality Assurance

"RFI" – Request for Interpretation

"RCV" – Rank Choice Voting

"SS" – Software Server

"TC" – Tabulator Client

"TDP" – Technical Data Package

"TM" – Transport Media (USB Thumb Drive)

"TR" – Tabulator Reports

"UPS" – Uninterruptible Power Supply

"VSTL" – Voting System Test Laboratory

"VVSG" – Voluntary Voting System Guidelines

2.0 CERTIFICATION TEST BACKGROUND

OVS 2.1 is a modification of a previously certified system (OVS 2.0.A). Pro V&V performed an evaluation of results from the previous test campaign to determine the scope of testing required for certification of the OVS 2.1. Based on this evaluation, Pro V&V determined that testing from the previous test campaign would establish the baseline and that the focus of this test campaign would be on the documented system updates.

2.1 Revision History

The table below details the version history of the OVS 2.1 System:

Table 2-1. OVS 2.1 System Revision History

System Version	Certification Type	Baseline System	Certification Number
OVS 2.0	New System	--- (Original System)---	UNS10121966-2.0
OVS 2.0.A	Modification	OVS 2.0	UNS10121966-2.0.A
OVS 2.1	Modification	OVS 2.0.A	* UNS10121966-2.1

*Upon grant of certification by the EAC

2.2 Scope of Testing

Testing from the previous test campaign was used to establish the baseline. The focus of this test campaign is on the introduction of end user enhancements and enhancements implemented based on the results of Vulnerability and Penetration testing conducted by DHS.

Based on the submitted modifications, it was determined the following tasks would be required to verify compliance of the modifications:

- Source Code Review, Compliance Build, Trusted Build, and Build Document Review
- System Level Testing (Functional Configuration Audit (FCA), Regression Testing, System Integration, Security, and Accuracy)
- Technical Documentation Package (TDP) Review
- Physical Configuration Audit, including System Loads and Hardening
- Usability Testing
- Security Testing

2.2.1 Modification Overview

The OVS 2.1 Voting System is a modified voting system configuration that contains updates to various components of the OpenElect Voting System. This release incorporates specific enhancement requests primarily centered on improving the end user experience. It also includes enhancements based on results of Vulnerability and Penetration testing conducted by DHS. Additionally, two laptop models were added for hosting the EMS applications, resulting in the addition of EOS 4.0.03.

2.2.1.1 Detailed List of Changes

The list below includes the submitted software changes between the OVS 2.1 system and the baseline of the OVS 2.0.A Voting System as derived from the *OpenElect Voting System Release Notes, System 2.0.A to 2.1, Version 1.2*:

Auditor (A)

- Add ability for end user to write-in candidate's name in the Auditor application

Ballot Layout Manager (BLM)

- Allow color for the non-partisan party to be changed
- Added support for a superuser user to load a new registration key at any time
- Implemented rotation based on voter registration data, generate a precinct/contest report for all elections, regardless of rotation
- Implemented Decline to State (DTS) function to allow non-partisan voters to selection from a list of DTS parties
- Update ballot design to follow the EAC best practices

- Move sound scripts to BLM
- Increase button text character limits for the OVO and FVT in order to accommodate translated text
- Translate all voter facing messages for the FVT
- Add the target color, Red or Black, to the Ballot's PDF name
- Change target color in drop down from 'Grey' to 'Black'
- Target example in Header Graphics made to match new ballot design
- Increase accuracy of text measurement when comparing against allocated space on ballot for Measure and Instruction Block text
- Change Election Proof Report to correctly represent full page width instruction blocks and measures
- On the Translation tab, create separate translations for the OVO and FVT for the final voting screen
- Update Screen Reader introduction hint text to correctly describe method to show and hide the screen in Screen Reader mode
- Option to combined PDF of all printable ballots
- Ballot Text supports inline size tags for varied font sizes
- Allow dynamic flow across columns for measure text
- Add the ability to have a variable data field in the header that describes the ballot, for example General Election Ballot or Measures Only Ballot
- Adding option for split merging, to allow for splits to merge to highest split instead of automatically to parent
- Merge Duplicate Splits offers two ways to roll-up splits by duplicate ballot style in the build process
- Enable OVO checking of one-time use barcodes
- Allow Contest Headers to use up to 1000 characters and support all inline style tags
- Add additional translations for OVO screens such as Ballot Page Not Cast or Ballot Jam
- In the BLM Translation table added the ability to create a blank field or to add the text "No Translation"

Election Manager (EM)

- Add readme file to election export to allow identification of election without having to decrypt files
- Add an option to the Election Manager's Common Options screen to print the candidate with the most votes in a contest in bold on the OVO and OVCS tally report
- Increment the EM portion of the election parameter version number when EM party icon is changed
- Move sound script management interface to the BLM to avoid moving between two applications to correct/test sound files
- Include ballot style listed on QR barcode when election is exported by ballot style
- Adjust election version number when machines list in election are changed to allow better detection
- Allow user to select font size layout for the text on the Precinct Barcode Report
- Enable sound script editing
- Enable Audio Script report button regardless of "Require Sound Files" option
- Allow custom image on Thank You screen
- Implement new TM USB handling with secure delete of files and signed unisyn_tm.vol file and clear previous error messages.
- Create election_info.txt file to contain election settings for reference without decryption
- Remove ambiguous letters and numbers, such as I and O, from auto-generated passwords (however, the Election Password entered by a user is not required to follow this rule)
- Remove 'Enter Alert Report Header Title' from OVO options screen, this text can be customized in the BLM Translations screen

FreedomVote Tablet (FVT)

- Handle formatting tags in the Instruction blocks
- Add the ability to perform a diagnostic test on the Text-to-Speech function
- Add system application API to allow the date and time to be changed from within the FVT application
- Add Election Type, Straight Ticket type, Retraction ID (if Set), Stub ID (if Set), and Android Build Number to the Machine Info Report

- Allow Custom image on Thank You screen
- Added support of the use of Mixed case (first letter capitalized, rest lower case) in accordance with best practices.
- Change Write-in entry screen's BACK button action to a cancel function
- Decline to State (DTS) functionality
- Skip invalid machine key files during setup
- Translate All Voter Facing Messages
- Update FVT Star printer driver
- Update "Machine Type" from BMT to FVT on Machine Info screen
- Update sound play handling to be more extensible (if a secondary language sound file does not exist it will default to the English sound file)
- Change 'Done' and 'Print' buttons on the screen to green color
- Remove picture of phone with barcode on Main and Start screens
- In Diagnostics Auto Test, when Cancel button pressed, show a confirmation message (yes/no) to exit
- Update the keypad layout to swap the right arrow and the Enter keys
- Display Android build version on the Machine Info Screen and Machine Info Report
- Add FVT application validation/verification handling
- Adjust color contrast on FVT as suggested by accessibility group
- On the FVT ballot screen the Done and Print buttons are not available until the voter is required to use them
- Added process to delete the previous election in the event that the FVT was stored with an election still loaded and the password has been misplaced
- Fixed defect where Header border not drawing correctly in screen group
- Improve spacing on Initialization barcode ticket and add formatting options that have been set
- Use PNG images for buttons on the bottom of the voting screen
- Change alert indicator color for undervotes and no selection on summary screen from RED to YELLOW

- FVT locks out all desktop access. The only user with access is a Supervisor who has access to the special file in place on a Unisyn provided TM, in addition to requiring password
- Update FVT to enable system power off feature
- Machine keystore password should be updated on install to be devices specific
- Update TM file cleaning process to a secure delete
- Text associated with BLANK candidate entry is displaying twice with Formatted Description only layout
- Remove ability to change Machine Name on setup screen
- Ballot Text needs to support inline size tags for varied sizes in a single block
- Allow Provisional Ballots to be printed on the FVT
- Enable OVO checking of one-time use barcodes for better security
- Allow the 'Done' button text on the FVT final voting screen to be changed in the BLM Translations, for example the text could be 'Review'
- Headers are increased (reference BLM)
- Redesigned the layout of the Settings screen to improve its usability (the more often used font size options were moved to the top of the screen)

OpenElect Voting Interface (OVI-VC)

- Implement 'Decline to State' function - allow Non-partisan voter to select from valid list of DTS parties
- Sound played after the ballot is printed is in the language selected
- Handle formatting tags in Instruction Blocks
- Allow translation for Multiple Screen Header Text, such as Ballot Title, Summary Title, and Instruction Title

OpenElect Voting Central Scan (OVCS)

- Add ability to enter custom names for OVCS sessions
- Allow export of OVCS session data to a TM for upload with the TC
- Adjust the definition of a mark and implement changes to support

- Add option to print in bold the candidate with the most votes in a contest on the tally report
- Process two column ballots with and without RCV with separate definition to improve performance
- Rotate the BMD full ballot image display in the write-in report to make it more readable
- Update TM file cleaning process to a secure delete
- Will recognize FVT Provisional ballots as provisional “P” ballots and not as BMD “D” ballots and handle accordingly
- Upload button is disabled after sorting any column in descending order in session list

OpenElect Voting Optical Scan (OVO)

- Add option to print in bold the candidate with the most votes in a contest on the tally report
- Create report which shows precinct count totals while the voting session is open
- Adjust the definition of a mark and implement changes to support
- Upgraded the OVO Printer Drivers to fix the “Printer is not connected” message after the printer paper has been changed
- Improved file cleaning process for TMs to ensure that all files are securely deleted
- Will recognize FVT Provisional ballots as provisional “P” ballots and not as BMD “D” ballots and handle accordingly
- Will recognize the FVT’s Ballot Verification Number (BVN) in the second barcode on an FVT ballot to prevent ballots from being cast a second time

Tabulator

- Implement Single Transferable Vote handling
- Allow automatic blanking of “0” in Voter Registration field
- Add "Elimination Only" option for RCV multiple seats method
- Append number of votes required to win (vote threshold) value to end of contest title in RCV tally report
- Add clarity to how RCV results are calculated for the exhausted ballots

- When "Multiple Deletions" is selected for tie breaking, if multiple deletion during tally process leads to continuing candidates less than seat number, automatically switch to manual elimination (eliminate one at a time)
- Voter Registration totals entered by party
- Voter Registration Totals Entered by Parent or Child
- Increase maximum Manual Entry from 5,000 to 10,000

Tabulator Reports (TR)

- Add option to print in bold the candidate with the most votes in a contest on the tally report
- Ensure that long contest titles are displayed on report
- Handle Voter registration totals entered by party
- When saving a report, the selected file format's extension (PDF or HTML) is automatically added to the end of the file name

Tabulator Client

- Handle OVCS session data to be uploaded from TM
- Improved file cleaning process for TMs to ensure that all files are securely deleted

All OCS Applications

- Change 'X' icon on dialog box to perform Cancel function instead of 'OK' function
- Handle formatting tags in the Instruction blocks
- Update Window Favicon to display the correct application logo instead of the default Java logo
- Display more specific error message when an expired registration key is selected
- When the Print button is selected for reports, it will first save the report to a file and then print the report
- Allow special characters for all OCS application passwords

OCS Installer

- Prevent the 2.1 OCS application release from being installed on EOS 1.1 (CentOS 5.7) systems

2.2.2 Block Diagram

Figure 1-5. OVS 2.1 System Components Overview

2.2.3 System Limits

Unisyn defined the following system performance characteristic limits for the OVS 2.1:

- The OCS and OVCS Software are intended for use by a single jurisdiction with one set of districts, voting centers, and precincts at any given time.
- The OCS and OVCS designed for handling up to 20 consecutive elections.
- Ten elections may be stored in the OCS database simultaneously.
- The system allows 400 ballot styles for an election.
- One OVO can be initialized for use with ballots up to 50 precincts, or All Precincts.
- An OVI-VC and FVT can present all precincts to the voter.
- The system allows voting by precinct and split precinct.
- The OVO provides several voting sessions to accommodate different ballot counting purposes. Only one session can be used at a time:
  - **Election Day** voting at the polls and voting centers
  - **Training Election** voting for training or sales purposes
  - **Logic Test** voting to test data and tabulation
  - **Absentee** session
  - **Recount** session
  - **Provisional** session
  - **Regional Early Voting** session
- The OVO provides the following System Performance:
  - Expected speed (per ballot page) 5 seconds to ballot page confirm
  - Throughput capacity (per ballot page) 6 ballot pages per minute
  - Maximum Volume 5,000 ballot pages
  - Ballot Pages
    - 11” ballot page:
      - Maximum number of voting positions per ballot page (11 inch ballot pages, without Rank Choice Voting): 228
      - Maximum number of voting positions per ballot page (11 inch ballot pages, with Rank Choice Voting): 456
    - 14” ballot page:
      - Maximum number of voting positions per ballot page (14 inch ballot pages, without Rank Choice Voting): 300

        - Maximum number of voting positions per ballot page (14 inch ballot pages, with Rank Choice Voting): 600

    - 17" ballot page:
        - Maximum number of voting positions per ballot page (17 inch ballot pages, without Rank Choice Voting): 372
        - Maximum number of voting positions per ballot page (17 inch ballot pages, with Rank Choice Voting): 744
    - 19" ballot page:
        - Maximum number of voting positions per ballot page (19 inch ballot pages, without Rank Choice Voting): 420
        - Maximum number of voting positions per ballot page (19 inch ballot pages, with Rank Choice Voting): 840

- Maximum number of ballot styles: 50 ballot styles per OVO session if individually entered with a maximum of 400 ballot style choices. For All Precinct sessions, all ballot styles are accepted by the OVO.
- The OVCS provides several voting sessions to accommodate different ballot counting purposes. Only one session can be used at a time:
    - Normal - Election Day Tabulation
        - Election Day Tabulation
        - Recount
        - Training
    - Logic and Accuracy Test (LAT) voting to test data and tabulation
    - Absentee session
    - Provisional session
    - Write-In only session
- The OVCS large scanner provides the following System Performance:
    - Max Ballot pages per batch 500
    - Max Ballot pages per session 5,000
    - Expected speed (ballot pages per hour) 2,100 ballot pages per hour
    - Maximum number of ballot styles: 400 ballot style choices

In the end-to-end OVS, a single election is limited to:

- Up to 12 political parties (including non-partisan) voting their own ballot in a Primary Election. Up to 12 political parties (including non-partisan) voting in a General Straight Ticket Election. Twenty-four parties may appear on the ballot for candidates.
- Up to 2,000 precincts.

- Up to 120 candidates per contest, with a limit of 3,000 combined count of candidates and contests.
- Up to 15 language translations (applies to ballot pages).
- Up to 3 ballot pages per ballot.
- Up to 5,000 ballot pages processed (cast votes) at an OVO during a single voting period.

2.2.4 Supported Languages

The OVS 2.1 Voting System supports the following languages types:

- Hindi
- Chinese
- English
- Japanese
- Korean
- Navajo
- Spanish
- Thai
- Bilingual (Spanish and English)

Support for all stated languages was verified; however, only English and Spanish language ballots were cast during the performance of functional testing. Additionally, one character based language (Chinese) was tested during System Integration Testing.

2.2.5 Supported Functionality

The OVS 2.1 is designed to support the following voting variations:

- General Election
- Closed Primary
- Open Primary
- Early Voting
- Partisan/Non-Partisan Offices
- Write-In Voting
- Primary Presidential Delegation Nominations
- Straight Party Voting (*including Pennsylvania and Indiana Rules*)
- Presidential-only Voting

- Split Precincts
- Multiple Selection Contests: Vote for N of M
- Ballot Rotation
- Cross Party Endorsement
- Multi-Page Ballots
- Multi-Party Candidate Endorsements
- Provisional or Challenged Ballots
- Single Transferrable Vote
- Proportional Rotation
- Absentee Ballots
- Recount Tabulation
- Ranked Choice Voting (RCV)

2.2.6 VVSG

The OVS 2.1 was evaluated against the relevant requirements contained in the EAC 2005 VVSG, Volumes I and II.

2.2.7 RFIs

There are no RFIs released by the EAC as of the date of this Test Report that pertain to this test campaign that were not already implemented into the certified baseline system.

2.2.8 NOCs

There are no NOCs released by the EAC as of the date of this Test Report that pertain to this test campaign that were not in effect at the time of the baseline system certification.

3.0 TEST FINDINGS AND RECOMMENDATIONS

The OVS 2.1 was evaluated against the relevant requirements contained in the EAC 2005 VVSG, Volumes I and II. The focus of this test campaign was on the modifications to the voting system configuration that included upgrades to the components of the baselined system. The summary findings and recommendations for each area of testing are provided in the following sections.

3.1 Summary Findings and Recommendations

Summary findings for the System Level Testing (System Integration Testing, Accuracy, and Limited FCA), PCA, and Source Code Review are detailed in the relevant sections of this report. In addition to these areas of testing, a TDP Review was performed, as described below.

<u>Technical Documentation Package (TDP) Review</u>

In order to determine compliance of the modified TDP documents with the EAC VVSG 1.0, a limited TDP review was conducted. This review focused on TDP documents that have been

modified since the certification of the baseline system. The review consisted of a compliance review to verify that each regulatory, state, or manufacturer-stated requirement had been met based on the context of each requirement.

Results of the review of each document were entered on the TDP Review Checklist and reported to the manufacturer for disposition of any anomalies. This process was ongoing until all anomalies were resolved. Any revised documents during the TDP review process were compared with the previous document revision to determine changes made, and the document was re-reviewed to determine whether subject requirements had been met. A listing of all documents contained in the OVS 2.1 TDP is provided in Table 3-1.

Table 3-1. OVS 2.1 TDP Documents

Document Number	Description	Version	Release
04-00512	Technical Data Package-Document List and Version Control	1.0	2.1
04-00446	OVS System Overview	1.3	2.1
04-00444	System Functionality Description	1.0	2.1
04-00458	System Hardware Specification	1.1	2.1
04-00464	Software Design and Specification	1.1	2.1
04-00447	System Security Specification	1.0	2.1
04-00453	System Test and Verification Plan	1.0	2.1
04-00460	Systems Operations Procedure: Warehouse Technician's Guide	1.0	2.1
04-00459	System Maintenance Procedures	1.0	2.1
04-00445	Personnel Training and Deployment Requirements	1.0	2.1
04-00448	Configuration Management Plan	1.0	2.1
04-00454	Quality Assurance Plan	1.0	2.1
04-00469	Final QA Report	1.0	2.1
04-00427	Election Manager User Guide	1.2	2.1
04-00428	Ballot Layout Manager User Guide	1.3	2.1
04-00431	Tabulator Client User Guide	1.1	2.1
04-00432	Tabulator User Guide	1.0	2.1
04-00433	Tabulator Reports User Guide	1.0	2.1
04-00495	OVCS User Guide	1.3	2.1
04-00530	Auditor Users Guide	1.0	2.1
04-00549	EOS Linux and OCS Installation Guide	1.6	2.1
04-00449	System Coding Standards	1.0	2.1
04-00462	Election Day Troubleshooter’s Guide	1.0	2.1
04-00463	Election Day Poll Worker’s Guide	1.1	2.1
04-00494	OVS Acronyms	1.0	2.1
04-00503	OVS Paper Specification	1.0	2.1
04-00553	Trusted Build – Applications 2.1	2.8	2.1
04-00594	OpenElect Voting System Release Notes, 2.0.A to 2.1	1.2	2.1
04-00513	System Hardware Verification	1.0	2.1

3.1.1 Source Code Review, Compliance Build, Trusted Build, and Build Document Review

Pro V&V reviewed the submitted source code to the EAC 2005 VVSG and the manufacturer-submitted coding standards. Prior to initiating the software review, Pro V&V verified that the submitted documentation was sufficient to enable: (1) a review of the modified source code and (2) Pro V&V to design and conduct tests at every level of the software structure to verify that design specifications and performance guidelines are met.

Summary Findings

- Automated Source Code Review: The Automated Source Code Review was performed during the OCS applications, OVO, OVI-VC, FVT, and OVCS Compliance and Trusted Builds. No source code issues were found during the Automated Source Code review.

- Manual Source Code Review: The Manual Source Code review was performed prior to the Compliance and Trusted Builds. The Manual Source Code was a comparison between the previously certified source code and the source code submitted for this test campaign.

- Compliance Build: The compliance build was performed following the compliance review. Once the compliance review was performed and the source was deemed stable enough to proceed with testing, the source code and all additional packages were compiled into a Compliance Build.

- Trusted Build: The trusted build consisted of inspecting customer submitted source code, COTS, and Third Party software products and combining them to create the executable code. This inspection followed the documented process from the "United States Election Assistance Commission Voting System Test Laboratory Program Manual" Section 5.5 – 5.7. *Performance of the trusted build includes the build documentation review*.

3.1.2 Physical Configuration Audit (PCA)

The Physical Configuration Audit (PCA) compares the voting system components submitted for qualification to the manufacturer's technical documentation, and included the following activities:

- Establish a configuration baseline of software and hardware to be tested; confirm whether manufacturer's documentation is sufficient for the user to install, validate, operate, and maintain the voting system

- Verify software conforms to the manufacturer's specifications; inspect all records of manufacturer's release control system; if changes have been made to the baseline version, verify manufacturer's engineering and test data are for the software version submitted for certification

- If the hardware is non-COTS, Pro V&V reviewed drawings, specifications, technical data, and test data associated with system hardware to establish a system hardware baseline associated with the software baseline

- Review manufacturer's documents of user acceptance test procedures and data against system's functional specifications; resolve any discrepancy or inadequacy in manufacturer's plan or data prior to beginning system integration functional and performance tests

- Subsequent changes to baseline software configuration made during testing, as well as system hardware changes that may produce a change in software operation are subject to re-examination

Summary Findings

During execution of the test procedure, the components of the OVS 2.1 system were documented by component name, model, serial number, major component, and any other relevant information needed to identify the component. For COTS equipment, every effort was made to verify that the COTS equipment had not been modified for use. Additionally, each technical document submitted in the TDP was recorded by document name, description, document number, revision number, and date of release. At the conclusion of the test campaign, test personnel verified that any changes made to the software, hardware, or documentation during the test process were fully and properly documented

3.1.3 System Level Testing

System Level testing included the limited Functional Configuration Audit (FCA), Regression Testing, Accuracy Testing, and the System Integration Tests. System Level testing was implemented to evaluate the complete system. The system was configured exactly as it would for normal field use. This testing included all proprietary components and COTS components (software, hardware, and peripherals) in both the EMS Standard and EMS Express system configurations. For software system tests, the tests were designed according to the stated objective without consideration of its functional specification. The system level hardware and software test cases were prepared independently to assess the response of the hardware and software to a range of conditions.

System Integration testing evaluated the integrated operation of both hardware and software. Compatibility of the voting system software components or subsystems with one another, and with other components of the voting system environment, was determined through functional tests integrating the voting system software with the remainder of the system.

The FCA for this test campaign included an assessment of the submitted modifications and tests conducted to verify that the system hardware and software performed as described in the manufacturer's documentation.

This evaluation utilized baseline test cases as well as specifically designed test cases and included predefined election definitions for the input data. The System Integration Tests were performed to verify the OVS 2.1 functioned as a complete system.

The accuracy test ensured that each component of the voting system could each process 1,549,703 consecutive ballot positions correctly within the allowable target error rate. The Accuracy test was designed to test the ability of the system to "capture, record, store, consolidate

and report" specific selections and absences of a selection. The required accuracy was defined as an error rate. This rate is the maximum number of errors allowed while processing a specified volume of data. For paper-based voting systems the ballot positions on a paper ballot must be scanned to detect selections for individual candidates and contests and the conversion of those selections detected on the paper ballot converted into digital data.

Summary Findings

Accuracy Testing: During Accuracy Testing, the OVO and OVCS were tested by utilizing a combination of hand marked (70%) and pre-marked (30%) ballots to achieve accuracy rate greater than 1,549,703 correct ballot positions. The OVI-VC and FVT were utilized to create ballots that were implemented as part of the pre-marked test deck.

The OVS 2.1 System successfully passed the Accuracy Test. During execution of the test procedure, it was verified that the OVS 2.1 System successfully completed the test with all actual results obtained during test execution matching the expected results.

System Integration Testing: To evaluate the integration of the system components and to perform the necessary regression testing, 3 General Elections and 3 Primary Elections were successfully exercised on the voting system, as described below:

Three general elections with the following breakdowns:

- General Election GEN-01: A basic election held in 4 precincts, one of which is a split precinct. This election contains 19 contests compiled into 4 ballot styles. 5 of the contests are in all 4 ballot styles. The other 15 contests are split between at least 2 of the precincts with a maximum of 4 different contest spread across the 4 precincts.
- General Election GEN-02: A basic election held in 3 precincts. This election contains 15 contests compiled into 3 ballot styles. 10 of the contests are in all 3 ballot styles with the other five split across the 3 precincts.
- General Election GEN-03: A basic election held in 2 precincts. This election contains 8 contests and compiled into 2 ballot styles. 4 of the contests are in both ballot styles. The other 4 contests are split between the two precincts. This election is designed to functionally test the handling of multiple ballot styles, support for at least three languages including a character-based language, support for common voting variations, and audio support for at least three languages and an ADA binary input device.

Three primary elections with the following breakdowns:

- Primary Election PRIM-01: Open Primary Election in two precincts. This election contained thirty contests compiled into five ballot styles. Each ballot style contains 6 contests.
- Primary Election PRIM-02: Open Primary Election held in two precincts. This election contained thirteen contests compiled into three ballot styles. One contest is in all three ballot styles; all other contests are independent.
- Primary Election PRIM-03: A basic election held in 2 precincts. This election contains 10 contests and is compiled into 2 ballot styles. 2 of the contests are in both ballot styles. The other 8 contests are split between the two parties' ballots. This Primary Election is designed to functionally test the handling of multiple ballot styles, support for at least three languages

including a character-based language, support for common voting variations, and audio support for at least three languages and an ADA binary input device.

The OVS 2.1 System successfully passed the System Integration Test. During execution of the test procedure, it was verified that the OVS 2.1 System successfully completed the system level integration tests with all actual results obtained during test execution matching the expected results.

Functional Configuration Audit (FCA): A Functional Configuration Audit was performed on the OpenElect 2.1 system to test the functional changes made to the system. Specially designed test cases were executed to assess that the changes perform as described in the system documentation.

Regression Testing: Using new software builds and updated documentation, all of the deficiencies listed in Table 3-2 were addressed.

3.1.4 Usability and Accessibility Testing

Usability & Accessibility testing was performed to evaluate the OVS 2.1 System to the applicable requirements. The usability testing focused on the usability of the OVS 2.1 system. Usability was defined generally as a measure of the effectiveness, efficiency, and satisfaction achieved by a specified set of users with a given product in the performance of specified tasks. The Accessibility portion of testing evaluated the requirements for accessibility. These requirements are intended to address HAVA 301 (a) (3) (B).

Summary Findings

The OVS 2.1 System successfully met the requirements of the Usability & Accessibility evaluation.

3.1.5 Security Testing

The objective of the security testing was to evaluate the effectiveness of the voting system in detecting, preventing, recording, reporting, and recovering from security threats and to determine the overall security posture of each system component. For this test campaign logical security testing was performed to verify that the EOS 4.0.03 OS introduced in this test campaign is properly configured per the recommendations provided in the Security Content Automation Protocol (SCAP) specifications.

Summary Findings:

Configuration Compliance Checking: An attempt was made to utilize the OpenSCAP Workbench utility to test the evaluation targets against the SCAP Checklists developed for Red Hat Enterprise Linux (RHEL). The RHEL content was used since there are no SCAP content specifications available for CentOS. It is common industry practice to substitute the RHEL SCAP content in evaluating CentOS.

During the testing, it was discovered that the security features that had been implemented on the Unisyn systems prohibited the use of the OpenSCAP Workbench tool. An attempt was made to

manually obtain the packages needed for offline installation. However, the systems also does not have a compiler resident in order to make the package. The automated scans could not be conducted without significant modification to the systems which revert the security mechanisms in place; therefore, the SCAP checklists where evaluated manually. The assessor obtained the vendor's compliance checklist matrix (CIS_CentOS_Linux_7_Benchmark_V2.2.0), evaluated its content completeness to the published checklists, and verified that the systems implemented all items as stated. The results of the Security Test were reported to the manufacturer for disposition of any anomalies. This process was repeated until all anomalies were resolved.

The OVS 2.1 System successfully met the requirements of the Security evaluation.

3.2 Anomalies and Resolutions

When a result is encountered during test performance that deviates from what is standard or expected, a root cause analysis is performed. Pro V&V considers it an anomaly if no root cause can be determined. In instances in which a root cause is established, the results are then considered deficiencies. No anomalies occurred during the testing of the OVS 2.1 System.

3.3 Deficiencies and Resolutions

Any violation of the specified requirement or a result is encountered during test performance that deviates from what is standard or expected in which a root cause is established is considered to be a deficiency. Upon occurrence, deficiencies are logged throughout the test campaign for disposition and resolution. All deficiencies encountered during the Unisyn OVS 2.1 test campaign were successfully resolved. In each instance, the resolutions were verified to be resolved through all required means of testing (regression testing, source code review, and TDP update) as needed.

The noted deficiencies are listed in Table 3-2.

Table 3-2. Noted Deficiencies

Deficiency #	Description of Deficiency	Resolution
BLM-1	Creating a contest with a large quantity of text in the Contest Title field but no text in the Report Title field causes the Election XML file to fail to be parsed by the Election Manager application when importing, and the Report Title field to be un-editable in BLM.	Addressed in the Ballot Layout Manager application Version 2.1
BLM-2	Entering Chinese text into translation fields in the BLM application, causes backslashes to be inserted in front of each character. Additionally, editing then exiting the translation field will add additional backslashes in front of each character.	Addressed in the Ballot Layout Manager application Version 2.1
EM-1	The Election Manager application does not properly support Contest Titles with 1000 characters. Upon attempting to import such an election, the application displays an error stating that the import could not be completed.	Addressed in the Election Manager application Version 2.1_R4

Table 3-2. Noted Deficiencies *(continued)*

Deficiency #	Description of Deficiency	Resolution
TR-1	Three of the reports generated by the Tabulator Report application do no properly append the file format extension to the filename when saving the reports.	Addressed in the Tabulator Reports application Version 2.1_R4
TR-2	The "Bold Candidate with Most Votes" option does not properly highlight candidates who receive a thousand votes or more.	Addressed in the Tabulator Reports application Version 2.1
FVT-1	Resolving a Printer Error while at a Ballot Review screen, causes the Review Screen Instruction audio to play while printing, instead of audio stating that the voter's ballot is being printed.	Addressed in the FreedomVote application Version 2.1_R3
OVI-VC-1	Alternate language ballots printed from the OVI-VC, do not print the Election Title on the BMD ballot even though the Translation is properly set.	Addressed in the OVI-VC application Version 2.1_R4
OVCS-1	The Accuracy Test Attempt #1 for the OVCS Mini, Canon DR-M160II scanner), failed due to a significant number of Double Feed Errors being encountered during testing (The system halted scanning and properly handled the errors, thus maintaining the accuracy of the test).	Unisyn Voting Solutions performed a Root Cause Analysis to determine the cause of the issue. The issue was addressed by adding a Document Feed Extension Support, and a roller bar to the scanner unit, as well as stating that no more than 20 ballots should be fed into the scanner at one time.

4.0 RECOMMENDATION FOR CERTIFICATION

The OVS 2.1 Voting System, as presented for testing, successfully met the requirements set forth for voting systems in the U.S. Election Assistance Commission (EAC) 2005 Voluntary Voting System Guidelines (VVSG), Version 1.0. Additionally, Pro V&V, Inc. has determined that the OVS 2.1 functioned as a complete system during System Integration Testing. Based on the test findings, Pro V&V recommends the EAC grant the OVS 2.1, identified in the following tables, certification to the EAC VVSG 1.0.

Table 4-1. OVS 2.1 COTS Software Components

FVT, OVO and OVI-VC Device Software	Version
CentOS Linux (OVO and OVI-VC)	6.3
Java JRE + Unlimited Cryptographic Extension	1.6.0_02
Android OS (FVT)	4.4.4

Table 4-2. OCS and OVCS COTS Software Components

OCS and OVCS Device Software	Version
CentOS Linux	6.5 and 6.8 and 7.6
Java JRE + Unlimited Cryptographic Extension	1.6.0_02
Apache-Tomcat Application Server	6.0.13
MySQL Database (BLM, EM, A, and Tab only)	5.0.45-7
JasperReports	2.0.5
OpenVPN	2.4.4
OpenSSL	1.0.1f
OpenSSL FIPS Object Module	2.0.10 (cert #1747)

Table 4-3. Voting System COTS Hardware

Hardware	Make	Model
OVO		
Duplex Ballot Scanner		
Duplex Ballot Scanner	PDI Scan	Pagescan III
Scanner Power Adapter	eUrasia Power	uA36-1024
58 mm Thermal Printer		
58 mm Thermal Printer	Citizen Printer	CT-5281
Printer Power Adapter	Citizen Printer	28AD4
Computer		
Chassis	Morex	Morex 2699
DC/DC converter	Morex	MX-0608F
Chassis Fans	Young Lin Tech	DFB404012M
Motherboard	Jetway	JNF9D-2550
Memory	SuperTalent - Onboard RAM	W1333SA2GV
Hard Drive	Western Digital	WD5000AZLX
AC Adapter	EDAC	EA 10951c-120
1 Gb USB TM		
Innodisk	1 Gb USB	DEUA1-01G172AC1SB-B088
Delkin	1 Gb USB	UY0GTFLSY-XN000-D
7” LCD Touchscreen Display		
7” LCD Touchscreen Display	Xenarc Technologies	700TSV
AC Power In Module		
AC Power In Module	Delta	Emi 10BEEG3G
OVI-VC		
Sip and Puff (Optional)		
Sip and Puff (Optional)	Origin Instruments	AirVoter
Headphone (Optional)		
Headphone (Optional)	Koss On-Ear Headphones	KPH5
15 in LCD Touchscreen Display		
15 in LCD Touchscreen Display	GVision	P15BX-OB-4690

Table 4-3. Voting System COTS Hardware *(continued)*

Hardware	Make	Model
82.5 mm Thermal Printer		
82.5 mm Thermal Printer	Star	TSP743IID-24, serial interface
Printer Adapter	Star	PS60A-24B 1
Computer		
Power Adapter Kit	DC-DC Converter	MX-0608F
Motherboard	Jetway	JNF9D-2550
Hard Drive	Western Digital	WD5000AZLX
Adapter	EDAC	EA 10951c-120
1 Gb USB TM		
Innodisk	1 Gb USB	DEUA1-01G172AC1SB-B88
Delkin	1 Gb USB	UY0GTFLSY-XN000-D
AC Power In Module		
AC Power In Module	Delta	Emi 10BEEG3G
OVCS		
Large Volume Scanner		
Large Volume Scanner	Canon	DR-X10C
Desktop Scanner		
Desktop Scanner	Canon	DR-M160II
FVT		
13.3 in Touchscreen Tablet		
13.3 in Touchscreen Tablet	Android Tablet	GVision - T13
Tablets Battery Charger	Sager Power System	GC30B-4P1J
82.5 mm Thermal Printer		
82.5 mm Thermal Printer	Star	TSP743IIU-24
Printer Adapter	Star	PS60A-24B 1
Barcode Reader 1D,2D series		
Barcode Reader 1D,2D series	Newland	FM420
USB Hub		
USB Hub	D-Link	DUB-H4
Hub Adapter	Meanwell	PSD-15A-05
1 Gb USB TM		
Innodisk	1 Gb USB	DEUA1-01G172AC1SB-B088
Delkin	1 Gb USB	UY0GTFLSY-XN000-D
Micro SD		
Micro SD	San Disk	4 GB Edge
Battery		
Battery Pack 12 V 6.0 AH	Powers Sonic	PSH-1255-FR
Adapter	Mean Well	GC30B-4PIL
AC Power In Module		
AC Power In Module	Delta	Emi 10BEEG3G

Table 4-3. Voting System COTS Hardware *(continued)*

Hardware	Make	Model
Sip and Puff (Optional)		
Sip and Puff	Origin Instruments	AirVoter
Headphone (Optional)		
Headphone	Koss On-Ear Headphones	KPH7
USB to Ethernet RJ45 Adapter (Optional)		
USB to Ethernet RJ45 Adapter	D-Link	DUB-E100
UPS		
Minuteman Power Technologies	Para Systems, Inc.	Entrepid Series

Table 4-4. OVCS System COTS Software Components

OVS Hardware	Version
Desktop for non-redundant solutions	Dell OptiPlex 360, 755, 7010, D075/XE2
Desktop for redundant solutions	Dell PrecisionT3500, T3600, T5810, T5820, 3420
Canon Scanner (OVCS)	Canon DR-X10C or DR-M160II
Laptop	Dell Latitude E5500, E5540, E5570, E5590, E5500 v2, Dell XPS m1530,HP 2000

ATTACHMENT A

AS-RUN TEST PLAN

Pro V&V Document No. TP-01-01-UNI-005-01.02, "Test Plan for EAC 2005 VVSG Certification Testing Unisyn Voting Solutions OpenElect 2.1 Voting System"

EXHIBIT F

IN THE UNITED STATES DISTRICT COURT
FOR THE NORTHERN DISTRICT OF GEORGIA
ATLANTA DIVISION

DONNA CURLING, *et al.* *Plaintiffs*, v. BRAD RAFFENSPERGER, *et al.*, *Defendants*.	CIVIL ACTION FILE NO. 1:17-cv-2989-AT

DECLARATION OF DR ERIC D. COOMER

Pursuant to 28 U.S.C. § 1746, I, Dr. Eric D. Coomer, make the following declaration:

1.

My name is Dr. Eric D. Coomer. I am over the age of 21 years, and I am under no legal disability which would prevent me from giving this declaration. If called to testify, I would testify under oath to these facts.

2.

I am the Director of Product Strategy and Security for Dominion Voting Systems ("Dominion"). I was previously the Vice President of U.S. Engineering for Dominion, and prior to that, I was the Vice President of

Research and Development for Sequoia Voting Systems. I have been in product development for election systems since 2005.

3.

Georgia is implementing a voting solution developed by Dominion for its elections. Specifically, Georgia has purchased Dominion Democracy Suite for use in its elections. Democracy Suite includes ImageCast X Ballot-Marking Devices (BMD), which operate with a printer to produce a paper ballot, and ImageCast Precinct Polling Place Scanners, which are optical scanners which allow ballots to be scanned and tabulated. Dominion is also providing KNOWInk Poll Pads for voter check-in.

4.

The BMDs produce printed ballots with barcodes to accurately and efficiently count each vote along with human-readable plain-text ballot summaries that voters can use to verify their selections. The barcodes do not include any identifying information that could be used to violate the anonymity of the voter. The BMD touch-screen tablets run the Dominion Voting Systems software in Kiosk mode. This prevents any access to software or features outside of the certified installed program. The certified software on the BMD does not allow any communication from the device over the internet. In addition, the BMD has no physical component that would allow

for wireless transmissions. The format and accuracy of the data stored in the barcode for a BMD-marked ballot was subject to testing and verification during the certification process.

5.

Several Election Technology Vendors, including Dominion Voting Systems, Election Systems and Software, Unisyn, and Smartmatic supply BMDs that use barcodes to create summary ballots for voters. Dominion's ImageCast X BMD system is currently used by Cook County and the City of Chicago, Illinois, several jurisdictions within the States of Michigan and Pennsylvania, and will be used by several California counties including San Francisco, Alameda, Riverside, Contra Costa, and San Diego in the upcoming 2020 election cycle. In addition, 6 of the 10 largest counties in the country use BMD devices with barcodes (1. Los Angeles, CA, 2. Cook County/City of Chicago, 4. Maricopa, AZ, 5. San Diego, CA, 8. Dallas, TX, and 10. Riverside, CA). Of the 6 counties, 5 are using the Dominion BMD.

6.

Texas' decision not to certify the Dominion system was primarily based on perceived issues of complexity of configuring the equipment especially for smaller counties. However, Dominion's Democracy Suite solution with the ImageCastX BMD has been successfully used in multiple small counties

throughout the country since 2016. Other perceived issues raised in the Texas certification were assertions that the secrecy of the ballot was not preserved. This is incorrect. All data stored on the ImageCast Precinct tabulators is randomized. There is no uniquely identifying information stored in the audit mark, or on the compact flash memory card for any ballot that is tabulated, whether from the ImageCastX BMD, or from a hand-marked ballot.

7.

Dominion has worked closely with Georgia election officials on the rollout of the new system. As part of the development and deployment of Democracy Suite statewide, no existing components of Georgia's legacy DRE system is being used. Dominion's Election Management System will completely replace the GEMS system used in Georgia. No information, software, or source code from the existing DREs is being utilized.

8.

The Poll Pads are completely replacing the old ExpressPoll units and no information from the existing ExpressPoll units will be placed into the Poll Pads. The Poll Pads will instead receive flat text files for each election containing the then-current voter information from the state's voter-registration database. Based upon information received from KNOWink's

President and CEO, the Poll Pad is certified for use in the following states: California, Idaho, Indiana, Minnesota, New York, Ohio, Pennsylvania, Texas, Virginia.

9.

Dominion's optical scanners (ICP) can be used with BMD-marked paper ballots or hand-marked paper ballots. The ICP units do not interpret the human-readable (text) portion of either type of ballot. Instead, the ICP units are programmed to read the QR Code for the BMD ballot or particular coordinates on hand marked ballot. Please see Exhibit A for a sample of coordinates on the hand-marked ballot. The target locations are then correlated to individual choices represented on the ballot. Pre-Logic and Accuracy Testing (Pre-LAT) is performed each election on every machine to verify that the target locations on hand-marked ballots, and the barcodes on BMD-marked ballots correspond correctly to the choices represented on the ballots and the digital cast-vote-records.

10.

Every ballot (hand-marked, or BMD generated) scanned on a Democracy Suite tabulator creates a digital image of the front and back of the ballot. In addition, the tabulators append an AuditMark to each image. The AuditMark is a text representation of how the tabulator interpreted the

ballot at scan time. This image can be used to audit the results of the election. Each AuditMark includes information on which tabulator the ballot was scanned. This unique feature of the ImageCast tabulators allows Election Officials to independently audit how the tabulator interpreted the QR Code or hand-marked ballot using a single image of the ballot with the appended AuditMark. It also includes a randomized sequence number when scanned on the ImageCast Precinct Tabulator. This randomized sequence number preserves voter anonymity as there is no way to correlate the sequence number to either an individual voter, or a specific point in time that the ballot was cast. When results and images are stored on the removable memory (Compact Flash cards), no date-timestamp information is included which prevents the ability to recreate the sequence of how the ballots were cast thus preserving voter anonymity.

11.

Any changes to the source code of any component of Democracy Suite would require new certification by the U.S. Election Assistance Commission and the State of Georgia.

12.

The Dominion BMDs being implemented in Georgia do not require calibration in the same way as Georgia's legacy DREs because they do not

use the same touch-screen technology. Instead, the ImageCastX BMD uses modern touch-screen technology, similar to an iPhone or Android phone, that does not require regular calibration.

13.

While all computers can be hacked with enough time and access, Dominion is not aware of any situation where an individual used the bar code on one of its units to launch software or to affect the operation of the unit.

I declare under penalty of perjury that the foregoing is true and correct. Executed this 13th day of November, 2019.

DR. ERIC D. COOMER

EXHIBIT A

Case 2:17-cv-00679-JAT Document 1358-2 Filed 06/18/19 Page 155 of 229

EXHIBIT G

The Ohio State Technology Law Journal

SEEKING THE SOURCE: CRIMINAL DEFENDANTS' CONSTITUTIONAL RIGHT TO SOURCE CODE

STEVEN M. BELLOVIN,[1] MATT BLAZE,[2] SUSAN LANDAU,[3] & BRIAN OWSLEY[4]

The right to a fair trial is fundamental to American jurisprudence. The Fifth Amendment of the Bill of Rights guarantees "due process," while the Sixth provides the accused with the right to be "confronted with the witnesses against him." But "time works changes, brings into existence new conditions and purposes." So it is with software. From the smartphones we access multiple times a day to more exotic tools—the software "genies" of Amazon Echo and Google Home—software is increasingly embedded in day-to-day life. It does glorious things, such as flying planes and creating CAT scans, but it also has problems: software errors.

Software has also found its way into trials. Software's errors have meant that defendants are often denied their fundamental rights. In this Article, we focus on "evidentiary software"—computer software used for producing evidence—that is routinely introduced in modern courtrooms. Whether from breathalyzers, computer forensic analysis, data taps, or even FitBits, computer code increasingly provides crucial

---

[1] Percy K. and Vida L.W. Hudson Professor of Computer Science and affiliate law faculty at Columbia University.
[2] Robert L. McDevitt, K.S.G., K.C.H.S. and Catherine H. McDevitt L.C.H.S. Professor of Law and Computer Science at Georgetown University.
[3] Bridge Professor in Cyber Security and Policy at The Fletcher School and the School of Engineering, Department of Computer Science, Tufts University.
[4] Assistant Professor of Law at University of North Texas Dallas College of Law. From 2005 until 2013, Brian Owsley served as a United States Magistrate Judge for the United States District Court for the Southern District of Texas.

Cite as: Steven M. Bellovin et al., Seeking the Source: Criminal Defendants' Constitutional Right to Source Code, 17 OHIO STATE TECHNOLOGY LAW JOURNAL (Dec. 2020), https://moritzlaw.osu.edu/ostlj/2020/12/22/seeking-the-source-criminal-defendants-constitutional-right-to-source-code/

trial evidence. Yet despite the central role software plays in convictions, computer code is often unavailable to examination by the defense. This may be for proprietary reasons—the vendor wishes to protect its confidential software—or it may result from a decision by the government to withhold the code for security reasons. Because computer software is far from infallible—software programs can create incorrect information, erase details, vary data depending on when and how they are accessed—or fail in a myriad of other ways—the only way that the accused can properly and fully defend himself is to have an ability to access the software that produced the evidence. Yet often the defendants are denied such critical access.

In this Article, we do an in-depth examination of the problem. Then, providing a variety of examples of software failure and discussing the limitations of technologists' ability to prove software programs correct, we suggest potential processes for disclosing software that enable fair trials while nonetheless preventing wide release of the code.

CONTENTS

I. INTRODUCTION 2
II. THE ROOT OF THE PROBLEM 17
III. SOURCE CODE AND CONSTITUTIONAL ISSUES 42
IV. CONCLUSION 71

I. Introduction

The right to a fair trial is fundamental to a democracy. In the United States, this right is guaranteed pursuant to the Fifth Amendment, which ensures "due process,"[5] and the Sixth Amendment, which provides the accused with the right "to be confronted with the witnesses against him."[6] These legal rights are a hallmark of the judicial process in the United States. Technology has transformed this, threatening the very

[5] U.S. CONST. amend. V.
[6] U.S. CONST. amend. VI.

right to a fair trial. The problem arises from the change in what constitutes a witness. At the time the Bill of Rights was written, witnesses were people, and it was well understood what it meant "to be confronted with the witness."[7] But times have changed.

Today we use complex software tools to measure certain types of activity, such as the level of blood alcohol a person has or how fast a car is being driven. We then use the results produced by these tools to provide evidence in court. The Federal Rules of Evidence enable defendants to question expert witnesses who present testimony on the validity of the evidence.[8] However, these protections do not extend to "questioning" computer code. Through three largely unrelated developments, these essential protections are in danger of being sharply curtailed, and therein lies a serious problem.

The first of the developments is the rise of "evidentiary software," that is, software whose output is itself evidence.[9] This has happened despite the prevalence of errors in large software systems, a concern well known to the computer science community but much less so to the public. The ubiquity of software errors is the second issue, since citizens may be convicted on the basis of what amounts to a witness bearing false testimony. The third arises from the increase of proprietary software; that and bureaucratic opacity have made it difficult for defendants to examine this software.[10] The result is a miscarriage of justice: conviction as a result of evidence that the defendants *are not able to cross-examine* and which, in fact, may be in error. Lest we lose the protections of the Fifth and Sixth Amendments, here we examine the confluence of these three issues.

---

[7] Interactive Constitution, *Sixth Amendment: Right to Speedy Trial by Jury, Witnesses, Counsel*, NAT'L CONST. CTR., https://constitutioncenter.org/interactive-constitution/amendment/amendment-vi [https://perma.cc/LAS7-2HPZ].
[8] FED. R. EVID. 702.
[9] While there are many examples, the simplest is the breathalyzer: complex software analyzes the breath sample for its alcohol content. *See, e.g.*, SEAN E. GOODISON ET AL., RAND CORPORATION, DIGITAL EVIDENCE AND THE U.S. CRIMINAL JUSTICE SYSTEM: IDENTIFYING TECHNOLOGY AND OTHER NEEDS TO MORE EFFECTIVELY ACQUIRE AND UTILIZE DIGITAL EVIDENCE (2015), https://www.ncjrs.gov/pdffiles1/nij/grants/248770.pdf.
[10] *See generally* Stephen W. Smith, *Policing Hoover's Ghost: The Privilege for Law Enforcement Techniques*, 54 AM. CRIM. L. REV. 233 (2017).

Others have discussed concerns about proprietary code[11] and the inability of defendants in civil and criminal cases to access the full evidence against them. While our fundamental concern is about fairness and upholding the rights pursuant to the Fifth and Sixth Amendments, our focus is somewhat different. We are concerned that the evidence is inaccurate—and because the defendants cannot access the underlying code, there is no way for the defendants to discover the falsehood of the "evidence" against them. What this means is that, although case law has moved to enable a defendant to cross-examine one who bears witness against him, the failure of the courts to understand the vagaries of evidentiary software and the great chance of errors in the evidence means that the right often fails in practice.

We will start our explanation of these issues with a simple example: the Breathalyzer.[12] The scientific principle behind these devices is straightforward: when someone drinks, alcohol is not digested, but is simply absorbed into the bloodstream where it does not undergo any chemical changes. When alcohol-laden blood reaches the lungs, the alcohol diffuses into the lungs proportionally to the amount in the blood. Thus, alcohol in exhaled breath provides a way to measure the level of alcohol in the blood.[13] The rest is elementary chemistry.

---

[11] *See generally* Rebecca Wexler, *Life, Liberty, and Trade Secrets: Intellectual Property in the Criminal Justice System*, 70 STAN. L. REV. 1343 (2018).

[12] "Breathalyzer" is a live trademark, serial number 72028025. That said, the word is often used generically; *see, e.g.*, Kashmir Hill, *Imagine Being on Trial. With Exonerating Evidence Trapped on Your Phone.*, N.Y. TIMES (Nov. 22, 2019), https://www.nytimes.com/2019/11/22/business/law-enforcement-public-defender-technology-gap.html [https://perma.cc/4TK5-9L8K] ("Law enforcement agencies get a new investigative technique — fingerprinting, DNA analysis, breathalyzer tests — and those representing the accused struggle to play catch-up.").

[13] Certain situations can change the validity of the reading, including whether the person being tested is diabetic or has been fasting—this may produce acetone, which is measured as alcohol—or whether they have recently used a mouth freshener, which may contain alcohol. Dentures can also trap alcohol in the mouth. *See* MURDO BLACK, ALCOLIZER TECHNOLOGY WHITE PAPER SERIES, SUBSTANCES THAT CAN AFFECT A BREATH TEST 3, https://www.alcolizer.com/wp-content/uploads/2017/01/12660_WhitePaper_Substances-that-Can-Affect-A-Breath-test_LR.pdf.

The explanation that follows is taken from the Breathalyzer's[14] inventor, Robert Borkenstein, once a member of the Indiana State Police and later a professor of forensic studies at Indiana University. The device shuttles a person's breath down a tube, eliminating the first portion of the breath. The breath is then combined with a mixture of sulfuric acid, potassium dichromate, silver nitrate, and water. The silver nitrate acts as a catalyst to speed up the chemical reaction transforming alcohol, potassium dichromate, and sulfuric acid into potassium sulfate, chromium sulfate, acetic acid, and water. The potassium dichromate and potassium sulfate are the chemicals to observe. Potassium dichromate is reddish-orange, while potassium sulfate is green. A photocell measures the decrease in yellow light, thus revealing the amount of alcohol present in the sample.[15] These systems are called "fuel-cell" breathalyzers.

Diagram taken from *How Stuff Works*.[16]

Although this chemistry is simple, chemistry itself is messy; it is much easier if one can dispense with "wet" devices. The development of infrared instruments for testing breath alcohol levels did so. The principle is again straightforward:

---

[14] "Breathalyzer" refers to Borkenstein's invention. When used with a small "b," the word denotes devices that chemically ascertain breath-alcohol levels from a person's breath. *See* A.W. Jones, *Physiological Aspects of Breath-Alcohol Measurement*, 6 ALCOHOL, DRUGS & DRIVING 1, 2 (Apr.-June 1990).

[15] R.F. Borkenstein & H.W. Smith, *The Breathalyzer and Its Applications*, 2 MED., SCI. & L., 13, 14 (1961).

[16] Craig Freudenrich, *How Breathalyzers Work*, HOW STUFF WORKS (Oct. 20, 2000), https://electronics.howstuffworks.com/gadgets/automotive/breathalyzer.htm [https://perma.cc/4KWT-DS27].

> Molecules absorb electromagnetic radiation at certain specific, unique wavelengths. Thus, it may be said that each molecule has its own 'infrared fingerprint.' Ethyl alcohol absorbs radiation at wavelengths of approximately 3.00, 3.39, 7.25, 9.18, 9.50, and 11.5 microns. No other compound absorbs radiation at all of those wave-lengths exclusively.[17]

Infrared and fuel-cell breathalyzers, which measure the current through the breath, are the most common types of breathalyzers.[18]

In order to accept the results of a breath-testing device, the judge and jury need to answer two questions: How does the device work? Does the device report alcohol breath levels accurately? The Breathalyzer was preceded by a simpler tool, the Drunkometer,[19] that operated on the same principles, and in 1955 Edwin Conrad, a senior attorney at the Federal Communications Commission, looked at how well evidence from various tools—radar guns, lie detectors, and Drunkometers—was being accepted by the courts.[20] He concluded that "scientific instrumentality of proof will not be accepted in evidence unless it has gained sufficient standing and scientific recognition, and has been demonstrated to be dependable."[21] Accepting the evidence required that it be:

> proved that qualified chemists made the test; the underlying theory and operation of the Drunkometer be explained in great detail; the accuracy of the machine in particular, and the accuracy of the method in general,

---

[17] AM. PROSECUTORS RSCH INST., BREATH TESTING FOR PROSECUTORS: TARGETING HARDCORE IMPAIRED DRIVERS 11 (2004), https://cdn.ymaws.com/mcaa-mn.org/resource/resmgr/files/tsrp/Resources/Breath_Testing_for_Prosecuto.pdf.
[18] *3 Types of Breathalyzers*, ELAWTALK, https://elawtalk.com/types-of-breathalyzers/ [https://perma.cc/AKT9-SY3L].
[19] Barron Lerner, *How Police Nab Drunk Drivers: From Drunkometer to Breathalyzer,* WBUR (Dec. 31, 2012), http://hereandnow.legacy.wbur.org/2012/12/31/breathalyzer-history [https://perma.cc/DAR6-LQLM].
[20] Edwin Conrad, *Push-Button Evidence*, 41 VA. L. REV. 217, 218 (1955).
[21] *Id.* at 219.

> were outlined by eminently qualified experts, and finally, the results of the test were interpreted medically.[22]

What constitutes such scientific "recognition"? The Federal Rules of Evidence Rule 702 on Testimony by Expert Witnesses states:

> A witness who is qualified as an expert by knowledge, skill, experience, training, or education may testify in the form of an opinion or otherwise if:
> (a) the expert's scientific, technical, or other specialized knowledge will help the trier of fact to understand the evidence or to determine a fact in issue;
> (b) the testimony is based on sufficient facts or data;
> (c) the testimony is the product of reliable principles and methods; and
> (d) the expert has reliably applied the principles and methods to the facts of the case.[23]

Pursuant to 1993 *Daubert v. Merill Dow Pharmaceuticals, Inc.*, the trial judge must determine "whether the testimony's underlying reasoning or methodology is scientifically valid and properly can be applied to the facts at issue."[24] In other words, the trial judge is the gatekeeper of scientific evidence determining what is admitted and what is excluded. It is important to note the role breathalyzers play in cases involving driving under the influence ("DUI") cases. Blood-alcohol concentration is what matters in American DUI cases, and thus, the breathalyzer measurements are critical.[25]

For breathalyzers, accuracy is determined by state and federal testing. The *Omnibus Transportation Employee Testing Act of 1991*[26] requires that pilots of various public transportation—aviation, mass transit, and

---

[22] *Id.* at 228.
[23] FED. R. EVID. 702.
[24] 509 U.S. 579, 580 (1993).
[25] Christopher Combs, *The Truth About Blood Alcohol Level & the Breathalyzer*, COMBS L. GRP. (June 3, 2018), https://www.combslawstl.com/blog/2018/06/03/the-truth-about-blood-alcohol-level-the-breathalyzer/ [https://perma.cc/BJ94-WZ3T].
[26] Department of Transportation and Related Agencies Act, Pub. L. No. 102-43, 105 Stat. 917, 953 (1992).

so on—be subject to alcohol testing programs, and the National Highway Traffic Safety Administration ("NHSTA") tests "evidential breath testers" on a regular basis[27] and publishes a "Conforming Products List of Evidential Breath Measurement Devices."[28] While states are not required to follow the federal list of approved devices, NHSTA provides funds to purchase approved devices, and thus many states adhere to the NHSTA list.[29] The devices' accuracy became trusted, and that trust lasted through several decades of cases.[30]

But there was always pressure against this trust. DUI cases form a peculiar subset of court cases; like jaywalking, cheating on income taxes, and failing to fully separate recyclable goods from trash, DUI is committed relatively frequently by people who are not considered criminals in the usual sense of the word. Yet the consequence of a DUI conviction—it may result in incarceration or the loss of a driver's license—can be high.[31] Accordingly, many drivers "lawyer up" when facing a potential DUI conviction. Legal attacks on breathalyzers' accuracy are thus not uncommon.[32]

---

[27] Highway Safety Programs; Model Specifications for Devices to Measure Breath Alcohol, 58 Fed. Reg. 48705 (Sept. 17, 1993).

[28] *See, e.g.*, Highway Safety Programs; Conforming Products List of Evidential Breath Alcohol Measurement Devices, 77 Fed. Reg. 35747 (June 14, 2012).

[29] U.S. GOV'T ACCOUNTABILITY OFF., GAO-08-477, TRAFFIC SAFETY: IMPROVED REPORTING AND PERFORMANCE MEASURES WOULD ENHANCE PERFORMANCE OF HIGH-VISIBILITY CAMPAIGNS (2008).

[30] As an example, in 2008 the New Jersey Supreme Court wrote that "[t]he accuracy and reliability of the breathalyzer itself has remained essentially unquestioned since our decision in *Romano v. Kimmelman*, 96 N.J. 66, 474 A.2d 1 (1984)." State v. Chun, 943 A.2d 114, 120 (N.J. 2008).

[31] With public transportation being minimal or even non-existent in many parts of the United States, loss of a driver's license can have a secondary, very severe penalty: loss of a job. *See* Alana Semuels, *No Driver's License, No Job*, ATLANTIC (June 15, 2016), https://www.theatlantic.com/business/archive/2016/06/no-drivers-license-no-job/486653/ [https://perma.cc/QEN6-C5E7]; Dana DiFillippo, *The Poverty Penalty: Should States Suspend Driver's Licenses for Scofflaws*, WHYY (Nov. 27, 2017), https://whyy.org/articles/poverty-penalty-states-suspend-drivers-licenses-court-scofflaws/ [https://perma.cc/2UKH-X5C3].

[32] *See generally* Ronald MacGregor, *Breathalyzers: Defects Then/Failures Still*, HG.ORG, https://www.hg.org/legal-articles/breathalyzers-defects-then-failures-still-30878 [https://perma.cc/L684-SUVE].

Modern devices all depend on software, but software unfortunately introduces new and potentially hidden failure modes into the technology. DUI convictions are per se violations based on breathalyzer results—a rating over .08 blood-alcohol level means the person is legally drunk[33]—making device accuracy critical. However, many of state testing labs were limited in their capabilities; the labs could only do "black box" testing, providing input to a breathalyzer and seeing the output, and not directly examining the software. Indeed, in many cases, even state lab testers lacked access to the code. In cases from Florida,[34] to Minnesota,[35] to New Jersey,[36] to Massachusetts, courts have said the denial of access to such source code[37] in criminal prosecution violates due process and the Confrontation Clause.[38] Thus, examinations of the software have started, and software in breathalyzers has been found to have errors.[39]

One important one was the 2008 New Jersey case of *State v. Chun*, a consolidation of twenty cases of individuals charged with DWI.[40] The counsel obtained the source code for the Alcotest 7110 MKIII-C breathalyzer.[41] The device works by using both infrared ("IR") and electrical chemical ("EC") oxidation technology to measure breath-alcohol levels.[42] Subjects are typically tested twice, for a total of four readings: IR_1 and EC_1 from the first breath and IR_2 and EC_2 from the

---

[33] *DUI Offense Basics*, FINDLAW, https://dui.findlaw.com/dui-charges/dui-offense-basics.html [https://perma.cc/F74G-G87S] (last updated Oct. 24, 2018).
[34] *See generally* State v. Allen, No. 08-CT-8840, slip op. at 2-3 (Fla. Cir. Ct. Apr. 24, 2009) (unpublished); State v. Atkins, No. 48-2008-CT-673-E, slip op. at 8-9 (Fla. Orange Cnty. Ct. June 20, 2008) (unpublished); State v. Lance Conley, No. 48-2012-CT-000017-A/A (Fla. Cir. Ct. Sept. 22, 2014).
[35] State v. Underdahl, 749 N.W.2d 117, 122-23 (Minn. Ct. App. 2008).
[36] *Chun*, 943 A.2d at 170.
[37] "Source code" is explained below. *See infra* Section II.
[38] Thomas E. Workman Jr., *Massachusetts Breath Testing for Alcohol: A Computer Science Perspective*, 8 J. HIGH TECH. L. 209, 232 (2008); *see also Chun*, 943 A.2d at 128-31.
[39] Stacy Cowley & Jessica Silver-Greenberg, *These Machines Can Put You in Jail. Don't Trust Them.*, N.Y. TIMES (Nov. 3, 2019) https://www.nytimes.com/2019/08/06/nyregion/crash-five-dead-long-island.html [https://perma.cc/K8PP-LV42].
[40] 943 A.2d. at 121.
[41] Evan Levow, *Summary of the Software House Finding for the Source Code of the Draeger Alcotest 7110 MKIII-C*, DWI.COM, https://www.dwi.com/new-jersey/state-v-chun/ [https://perma.cc/URH9-94KC].
[42] *Chun*, 943 A.2d. at 128.

second breath.[43] The measurements from the first breath are compared to the measurements from the second breath; if the measurements from the first breath are not within accepted levels of tolerance of the second breath, then the machine operator requires the subject to take a third breath; this will again provide two readings.[44]

The outside examiners noted that, "[w]hen the software takes a series of readings, it first averages the first two readings. Then, it averages the third reading with the average just computed. Then the fourth reading is averaged with the new average, and so on."[45] (Mathematically, that is not an average, as it weighs the first reading significantly more than the others, the second less than the first, but more than the others, etc.) The examiners were concerned that the error detection logic of the device needed a measurement error to register thirty-two times in a row for an error message to be displayed; otherwise the error was ignored.[46]

The court's special master deemed the device to be scientifically reliable,[47] but the court itself did not.[48] The New Jersey Supreme Court was not bothered by the way the device computed the so-called average, instead pointing out that what this did was give higher weight to the portion of the breath from the deepest air in the lung, the portion that most accurately reflects blood-alcohol levels.[49] But the court observed an error occurred when one of the original four readings fell outside accepted tolerance of the device.[50] In such a case, the subject is asked to do a third breath into the device,[51] and all six readings—two from each

---

[43] Jeffrey Lustick, *Getting to Know Washington's New DUI Breathalyzer: The German Made Dräger Alcotest 9510*, LUSTICK KAIMAN & MADRONE PLLC (Feb. 24, 2016), https://www.lustick.com/blog/getting-know-washingtons-new-dui-breathalyzer-german-made-drager-alcotest-9510/ [https://perma.cc/LN99-Q6QL].
[44] *Id.*
[45] Levow, *supra* note 41.
[46] *Id.*
[47] *Chun*, 943 A.2d. at 153; STEPHEN P. SMITH ET AL., IIT RESEARCH INSTITUTE, INDEPENDENT REVIEW OF THE CARNIVORE SYSTEM, FINAL REPORT xiii (2000), https://www.epic.org/privacy/carnivore/carniv_final.pdf.
[48] *Chun*, 943 A.2d. at 153.
[49] *Id.* at 156-57.
[50] *Id.* at 157-58.
[51] According to the court, as many as 5% of defendants might be tested on a third breath. *Id.* at 157.

breath—are included in its calculation (or at least, they should be). As the New Jersey Supreme Court discovered, this did not happen.

If the second EC reading was low—indicating the subject had low blood-alcohol levels—that evidence was simply ignored by the system. This was not because of malfeasance, but because of a programming error "due to [a] buffer overflow error"—an overwriting of the data.[52] Buffer overflow errors been known academically since 1981[53] (and in practice since 1988[54]). If the defense had an opportunity to examine the code, this error might have been found, in which case the evidence would have been dismissed.

That such an error should occur in a breathalyzer marketed in 2008 is both astonishing and highly disturbing. Avoiding such errors is always taught in first semester programming classes;[55] the existence of such an error demonstrates a serious lack of quality control by the vendor. An adversarial audit may have found the problem; without both source code and a suitable test environment (itself hard to construct without source code), there would be no way to find it.

The outside examiners found "thousands of programming errors" upon testing the Alcotest 7110 MKIII-C.[56] Drager, the manufacturer, claimed to have fixed the problems, but in 2019 the *New York Times* reported that the state failed to update the software in its devices.[57]

---

[52] *Id.* at 157-59.
[53] *See* C.A.R. Hoare, *The Emperor's Old Clothes*, 24 COMMC'NS ACM 75, 76-77 (1981).
[54] Jon A. Rochlis & Mark W. Eichin, *With Microscope and Tweezers: The Worm from MIT's Perspective*, 32 COMMC'NS ACM 689, 689 (1989).
[55] Buffer overflows are typically caused by lack of bounds-checking on what are known as "array indices." Improper indices are always wrong; using them can cause program crashes, erroneous results, or security problems. Consequently, all introductory classes and texts stress this point. *See, e.g.*, ROBERT SEDGEWICK & KEVIN WAYNE, INTRODUCTION TO PROGRAMMING IN JAVA: AN INTERDISCIPLINARY APPROACH §1.4 (2d ed., 2017) ("When programming with arrays, you must be careful. It is your responsibility to use legal indices when accessing an array element.").
[56] Cowley & Silver-Greenberg, *supra* note 39.
[57] *Id.*

Programming errors such as these are not limited to breathalyzers. A particularly striking example comes from Carnivore,[58] an FBI program for analyzing network traffic—this is called a packet sniffer—that was developed in the late 1990s and used on ISP's networks. Using filters to record traffic fitting some predetermined pattern, such as all email to or from a particular target, Carnivore would collect traffic pursuant to a pen/trap order or wiretap warrant.[59]

The software program was not properly designed, however, enabling incorrect collection of communications traffic. In 2000, an outside review committee discovered that, "Incorrectly configured, Carnivore can record any traffic it monitors."[60] The ability to record "any traffic" should not have been possible. In addition, the system permitted additional copies of intercepts to be made, even if they had not been minimized (a requirement for all lawful wiretaps); only FBI "procedures and professionalism" prevented that from occurring.[61]

One might imagine that the concerns the oversight committee raised were theoretical worries; surely the device would not actually exhibit such flaws in practice. In fact, serious problems occurred in the collection. An FBI anti-terrorism investigation by the team focusing on Osama bin Laden used Carnivore.[62] According to an internal memo, "[t]he FBI software not only picked up the E-mails under the electronic surveillance of the FBI's target XXX but also picked up E-mails on non-covered targets. The FBI technical person was apparently so upset that he destroyed all the E-mail take, including the take on XXX."[63]

---

[58] Carnivore was later renamed DCS-1000, for Digital Collection Service 1000. *DCS-3000 is the FBI's New Carnivore*, WIRED (Apr. 27, 2006, 7:04 PM), https://www.wired.com/2006/04/dcs3000-is-the-/ [https://perma.cc/U6VQ-EUCH].
[59] SMITH ET AL., *supra* note 47.
[60] *Id.* at xii.
[61] *Id.* at 4.2-4.3.
[62] The unit was known as the "UBL" unit; they spelled the target's name "Usama bin Laden." Michele Zanini & Sean J.A. Edwards, *The Networking of Terror in the Information Age*, *in* NETWORKS AND NETWARS: THE FUTURE OF TERROR, CRIME, AND MILITANCY, 39 (1999), https://www.rand.org/content/dam/rand/pubs/monograph_reports/MR1382/MR1382.ch2.pdf.
[63] Memorandum from the FBI on FISA Mistakes to Spike Bowman (April 5, 2000) (on file with Electronic Privacy Information Center) https://www.epic.org/privacy/carnivore/fisa.html [https://perma.cc/6FND-CHYV]. Context is given in the EPIC press release. *See* Press Release, Electronic Privacy Information Center, FBI's Carnivore System Disrupted Anti-

Carnivore was collecting digital evidence, which makes the fact that the Carnivore software was improperly designed particularly disturbing. Breathalyzers test a physical object—a human breath. While it is not standard to preserve the actual breath sample, it is possible to require that such samples be preserved until court hearings have occurred; in fact, the states of Alaska and New Hampshire do exactly that.[64] However, the digital artifacts that Carnivore collected *were* the actual evidence itself. One aspect of the Carnivore evidence collection process—overcollection—had gone wrong, but so could other aspects of collection. This raises the possibility that the collected evidence as seen by the court could be incorrect or it could appear misleadingly out of context. This makes clear that the ability of the defendant to cross-examine the "witnesses"—in this case, the digital code that did the collection—is crucial.

Computer forensic analysis provides yet a third example of problems with evidentiary software. Apple forensics expert Jonathan Zdziarski was hired by the United States Army to examine phone evidence in the case of a brigadier general who had been accused of sexual harassment, assault, and making murder threats.[65]

Terror Investigation (May 28, 2002) (on file with author) https://www.epic.org/privacy/carnivore/5_02_release.html [https://perma.cc/QM3T-KYBF].

[64] Best v. Mun. of Anchorage, 712 P.2d 892, 898–99 (Alaska Ct. App. 1985); Mun. of Anchorage v. Serrano 649 P.2d 256 (Alaska Ct. App. 1982) (state must preserve a breath sample). Current New Hampshire law requires the collection and preservation of an additional blood sample, and that someone who is asked to perform a breath test shall be informed that they have the right, at their own expense, to have a blood sample drawn and preserved. *See* In re Op. of Justices (Eliminating Requirements for Additional Breath Samples), 2 A.3d 1102 (N.H. 2010) (New Hampshire Supreme Court rejected proposed amendment to eliminate the requirement to save two breath samples). Other states have a duty to preserve, but it is not constitutionally mandated. Arizona has a duty to preserve breath samples for independent testing pursuant to its implied consent law. *See* RSA 265-A:7 II and III; State v. Vannoy, 866 P.2d 874, 878 (Ariz. Ct. App. 1993) (discussing ARIZ. REV. STAT. ANN. § 28-691). Failure to preserve breath samples did not violate due process or equal protection. *See also* People v. Molina, 468 N.Y.S.2d 551, 558 (N.Y. Crim. Ct. 1983) ("basic fairness and the duty to make of the trial a search for truth informed by all relevant material requires that the police preserve for the defendant's use a separate breath sample for testing and analysis"); People v. Trombetta, 219 Cal. Rptr. 637 (Cal. Ct. App. 1985).

[65] SUSAN LANDAU, LISTENING IN: CYBERSECURITY IN AN INSECURE AGE 145 (2017) (ebook).

Zdziarski was brought into the case late, just before the trial was to start. In the course of examining the phone, Zdziarski determined that the computer forensic tools initially used in the investigation did not work correctly. Timeline was a crucial issue in the case, and so Zdziarski needed to look at timestamps for device erasures, backup restores, and file accesses. The commercial forensic tools had reported these inaccurately. Problems included incorrect times for app deletion, basing the time on when the phone was next booted up rather than when the app was actually removed.[66] This meant that the forensic tools were miscalculating when the app was actually used. Zdziarski's analysis changed the case, with the Army dropping the most serious charges.[67]

Breathalyzers, packet sniffers and wiretaps, and computer forensic tools are far from the only examples of evidentiary software. Indeed, the errors we have discussed present but the tip of an iceberg. Other instances include radar guns, tools for conducting DNA analysis, and network investigative techniques (the last is government malware used, pursuant to court order, to intrude onto a system in order to investigate and/or obtain evidence).[68] Moreover, with our drive willy-nilly into the Digital Revolution, we are moving into a world in which law enforcement will increasingly rely on evidentiary software. From automatic toll booths that register when a car enters and exits a highway to personal devices that record where and when an individual is, the percentage of cases that rely on evidentiary software is increasing. The fact that such software may incorrectly report its findings makes cross-examining source code crucial to the basic protections laid down in the Bill of Rights.

Yet that capability, so necessary for protecting a defendant's rights, is disappearing. Even as we rely on computer software to bear witness, we are simultaneously moving to a criminal justice system in which "law

---

[66] Jonathan Zdziarski, *An Example of Forensic Science at Its Worst: US v. Brig. Gen. Jeffrey Sinclair*, ZDZIARSKI'S BLOG OF THINGS (Aug. 24, 2014), https://www.zdziarski.com/blog/?p=3717 [https://perma.cc/PR75-FW6A].
[67] LANDAU, *supra* note 65.
[68] *See, e.g.*, Jennifer Stisa Granic, *Challenging Government Hacking: What's at Stake*, ACLU (Nov. 2, 2017, 10:00 AM), https://www.aclu.org/blog/privacy-technology/internet-privacy/challenging-government-hacking-whats-stake [https://perma.cc/T6HC-FZ27].

enforcement privilege"[69] frequently prevents access to the details of investigative techniques and procedures.

Given the guarantees of the Fifth and Sixth Amendments, it may be difficult to understand how prosecutors are able to withhold details of the tools used in evidence gathering. This approach is a relatively new phenomenon.[70] Recently, United States Magistrate Judge Stephen Smith presented a compelling history of the law enforcement privilege's genesis.[71] The change began with the 1950 espionage trial of Judith Coplon, a Department of Justice employee in the Foreign Agents Registration Division.[72] Coplon was also a Soviet spy. The FBI had become suspicious of her and fed her a false document.[73] Coplon was arrested as she was handing over the fake document—and twenty-eight classified memoranda—to her accomplice.[74]

The twenty-eight other memoranda were a problem for FBI Director J. Edgar Hoover, for they showed evidence that the FBI was illegally wiretapping.[75] In 1937, ruling on the basis of the Federal Communications Act,[76] the Supreme Court deemed it illegal to "intercept" and "divulge" wired communications.[77] Initially, the Department of Justice interpreted the decision to mean interception and divulgence of content *outside the federal government* was unlawful.[78] In 1940 President Franklin Roosevelt authorized the practice against those suspected of subversive activities against the United States government, urging Attorney General Jackson to limit the investigations

---

[69] Smith, *supra* note 10, at 233.
[70] *Id.*
[71] *See generally id.*
[72] *See generally* United States v. Coplon, 185 F.2d 629 (2d Cir. 1950).
[73] *Id.* at 632-35.
[74] ATHAN G. THEOHARIS & JOHN STUART COX, THE BOSS: J. EDGAR HOOVER AND THE GREAT AMERICAN INQUISITION 256 (1988).
[75] Smith, *supra* note 10, at 237.
[76] 47 U.S.C. § 605.
[77] Nardone v. United States, 302 U.S. 379, 382 (1937).
[78] SELECT COMM. TO STUDY GOVERNMENTAL OPERATIONS WITH RESPECT TO INTELLIGENCE ACTIVITIES, S. REP. NO. 94-755, 94th Cong., 2d Sess., Book III, at 278 n.25 (letter from Attorney General Robert Jackson to Rep. Hatton Summers, March 19, 1941).

to a minimum and "limit them insofar as possible to aliens."[79] Nonetheless, the original restriction of the court still stood. The FBI had been wiretapping Coplon; although Hoover sought to conceal that fact during Coplon's trials, her defense attorney was able to get it into the record.[80] As a result, Coplon's convictions were overturned.[81] The more significant result from this was Hoover's response, which was effectively to hide wiretaps from view.[82]

The wily FBI director did not stop there. As Stephen Smith has detailed, Hoover publicly campaigned to establish an "evidentiary privilege covering law-enforcement techniques and procedures."[83] Evidentiary privilege was a new idea in 1950. However, as we can see from very public law-enforcement efforts to keep the techniques of investigative technologies hidden,[84] it is an oft-used practice.

The FBI director's efforts are not the only ones pushing in favor of evidentiary privilege. Judge Smith details how a small exemption for law-enforcement records in the Freedom of Information Act became a Mack Truck-sized hole that has permitted the withholding of evidentiary information by multiple courts.[85]

Sometimes legal problems arise from a single issue, as it did in the Coplon trial, where the withholding of evidence resulted in an unfair trial and Coplon's release. Sometimes legal problems develop from a

---

[79] *Id.* at 279 (Franklin D. Roosevelt, Confidential Memorandum for the Attorney General, May 21, 1940).
[80] *See* Smith, *supra* note 10, at 234.
[81] Coplon was tried in the District of Columbia on unauthorized possession of classified documents and in the Southern District of New York on espionage. The illegal wiretapping formed the basis for an appeal of Coplon's second trial, which won her a reversal. Smith, *supra* note 11, at 234-37.
[82] Wiretap evidence including requests for taps, etc. would be stored separately from the main files of a case, with only a single copy of the wiretap stored. This was held in a secure area at FBI headquarters. THEOHARIS & COX, *supra* note 74, at 259-60.
[83] Smith, *supra* note 10, at 243.
[84] Stephanie K. Pell & Christopher Soghoian, *Your Secret Stingray's No Secret Anymore: The Vanishing Government Monopoly over Cell Phone Surveillance and Its Impact on National Security and Consumer Privacy*, 28 HARV. J.L. & TECH. 1, 33-34 (2014).
[85] Smith, *supra* note 10, at 248-50.

confluence of issues. That is the situation here with the combination of evidentiary software, evidentiary privilege, and software error.

The issue of software error occupies a large part of computer science research and practice,[86] including security (a high percentage of security problems are due to buggy code[87]) and concerns about fairness and accuracy in machine learning. Here, we have focused narrowly on how the combination of evidentiary software, evidentiary privilege, and software error poses a threat to the fundamental right of criminal defendants to a fair trial. The good news is that, unlike the larger problem of software error, this problem posed by errors in evidentiary software appears relatively solvable. While our proposed solution challenges the present handling of the situation, our recommendation is technically feasible and is without high financial costs.[88]

We begin in Section II with an explanation of the nature of software and thus of coding errors, ending with an explanation of modern techniques used to find and correct these problems. In Section III we discuss the three Constitutional concerns that are raised by software errors: *Brady* violations (the prosecution must provide exculpatory evidence to a defendant),[89] the implications of the Confrontation Clause, and the Due Process clause. We conclude in Section IV with our recommendations.

II. The Root of the Problem

The essence of our argument is this: the nature of software, and hence of computer programming, is such that certain errors are more likely to

---

[86] Buggy code has long been a focus for academics and practitioners alike. "First, one must perform perfectly. The computer resembles the magic of legend in this respect, too. If one character, one pause, of the incantation is not strictly in proper form, the magic doesn't work. Human beings are not accustomed to being perfect, and few areas of human activity demand it. Adjusting to the requirement for perfection is, I think, the most difficult part of learning to program." FREDERICK P. BROOKS, JR., THE MYTHICAL MAN-MONTH 8 (1975).

[87] *See* WILLIAM R. CHESWICK & STEVEN M. BELLOVIN, FIREWALLS AND INTERNET SECURITY: REPELLING THE WILY HACKER 7 (1994).

[88] It is in fact not clear that there will be a net increase in costs, if early audits reduce the need for future litigation.

[89] Brady v. Maryland, 373 U.S. 83, 87 (1963).

be found by adversarial testing. We start by giving a brief overview of the nature and limitations of software development.

The software design process typically starts with the creation of "specifications," a detailed description of what the system should do.[90] It is not enough to say, for example, "measure breath alcohol level." What are the expected ranges? (A real breath sample will never consist of pure ethanol.) How much breath must be sampled? How close should it come to the results from an actual blood test? What sort of operational errors should be detected? What should the system do if it encounters anomalous situations? Should there be internet connectivity, to relay results immediately? Should there be a camera and a GPS, for better evidentiary value? If so, what if the device is used in a cellular dead spot, or if the camera or GPS are not working? How many samples from how many drivers must the device store?

Errors can creep in at any point. Unexpectedly high readings might erroneously show up as low because the device was never intended to handle such values.[91] The specification may have omitted important details, e.g., what to do if the battery voltage is below a certain level. Programmers use the specifications to guide writing the actual programs. Testers then use the same specification documents to guide their tests. An error in the specifications, then, can turn into erroneous code; such errors will not be detected by the testers, since their job is to ensure that the program matches the specifications.

Before we explain how coding errors can occur, we provide a brief note on terminology. As is widely known, computers "understand" only 0s and 1s. That is, the actual computer hardware is only capable of understanding "binary": a way to represent numbers or characters using only the values 0 and 1, commonly understood as an electrical circuit

---

[90] *See, e.g.*, Barry W. Boehm, *Verifying and Validating Software Requirements and Design Specifications*, IEEE SOFTWARE, Jan. 1984, at 76.

[91] Extreme values can occur legitimately. For example, in a recent New York criminal case, the defendant was described as having "the second highest level of marijuana that they've ever seen in a living specimen." Arielle Dollinger, *The Stolen Car Was Going 154 M.P.H. Five People Ended up Dead.*, N.Y. TIMES (Aug. 6, 2019), https://www.nytimes.com/2019/08/06/nyregion/crash-five-dead-long-island.html [https://perma.cc/3H9S-3E3S].

being off or on. This is the reason that the actual programs run by a computer chip are often referred to as "binaries" or "executables." Programmers, though, almost never directly code actual binary programs. Instead, they write in "source code," a (more-or-less) human-readable language.[92] Specialized programs known as "compilers" translate the source code to executables. While it is possible, with proper tools and considerable effort, for a human to understand an executable—indeed, analysis of computer viruses and other malware relies on this ability[93]—analysis is far easier and far more complete if the original source code is available.

A. Computer Programming

It is a truism that computers do only what they are told to do. The process of "telling" a computer what to do is called programming, and while programming is no longer the rare, arcane art that it once was, nevertheless many people have no exposure to it. Then, we explain programming errors that cause bugs as well as how programmers test to eliminate such bugs. Finally, we address the necessity of adversarial audits to safeguard defendants' rights.
Consider the following program fragment.[94]

```
int a, b;                /* Reserve two storage locations */

scanf("%d", &a);         /* Read the first value */
scanf("%d", &b);         /* Read the second */
printf("%d\n", a + b);   /* Print their sum */
```

It reads into two values and prints their sum. There are several things worth noting about these few lines of code.

---

[92] Some examples of source code are given. *See infra* Section II.A.
[93] *See generally* MICHAEL SIKORSKI & ANDREW HONIG, PRACTICAL MALWARE ANALYSIS: THE HANDS-ON GUIDE TO DISSECTING MALICIOUS SOFTWARE (Alison Law et al. eds., 2012).
[94] This fragment is written in a language known as C. *See generally* BRIAN W. KERNIGHAN & DENNIS M. RITCHIE, THE C PROGRAMMING LANGUAGE (2d ed. 1978). C is the ancestor of many currently used programming languages. Neil DuPaul, *The History of Programming Languages Infographic*, VERACODE (Apr. 15, 2013), https://www.veracode.com/blog/2013/04/the-history-of-programming-languages-infographic [https://perma.cc/ZPR2-HDR5].

The first, of course, is that the code itself—the first portion of each line—is completely incomprehensible to anyone but a programmer. The second thing to note is that the second half of each line, the portion enclosed between /* and */, is readable. These are known as "comments" and are ignored by the computer. This particular example is artificial, in that not only are there more comments than is customary, they're redundant to the code and would be seen as bad practice. That is, a comment like "Print their sum" is rather useless, since to any programmer it is apparent that is what the line does.[95] A better comment is one that explains why the sum is being printed, or what the input values mean.[96] Nevertheless, even to experienced programmers comments can be easier to read than code.

There is another, more subtle point: to a first approximation, every line of code interacts with every other. For example, in this case the first line says the storage locations "a" and "b" can only hold integers. It is straightforward to make them hold numbers with fractional parts, e.g., 1.5, by writing:

```
float a, b;
```

instead, but that would require certain changes to the corresponding scanf line to let it read so-called "floating point" numbers. Furthermore, and less obviously, the printf statement would require certain changes as well.

---

[95] Even non-programmers can see the root word "print" and "a + b". These mean exactly what they appear to mean. H. James de St. Germain, *Commenting*, JIM'S CS TOPICS, https://www.cs.utah.edu/~germain/PPS/Topics/commenting.html#:~:text=Comments%20should%20be%20useful%20high,easy%20to%20read%20as%20English. [https://perma.cc/M5QR-JE6D].

[96] Proper commenting style is routinely taught in introductory programming classes. *See id.*

Computers derive much of their power from two features: the ability to choose among alternatives and the ability to repeat activities. Suppose we wanted to change the above program to subtract two numbers, but always to subtract the lesser from the greater:

```
int a, b;               /* Reserve two storage locations */

scanf("%d", &a);            /* Read the first value */
scanf("%d", &b);            /* Read the second */
if (a > b)
            printf("%d\n", a - b);
    else
            printf("%d\n", b - a);
```

We could also do this repeatedly, for many pairs of numbers:

```
int a, b;               /* Reserve two storage locations */

while (1) {
    scanf("%d", &a);        /* Read the first value */
    scanf("%d", &b);        /* Read the second */
    if (a > b)
            printf("%d\n", a - b);
    else
            printf("%d\n", b - a);
}
```

We will forbear explaining these program fragments in detail, save to note that the last example has a bug: it will run forever, even when there is no more input available.[97]

---

[97] *See generally* KERNIGHAN & RITCHIE, *supra* note 94. There is an old joke about why programmers starve to death in the shower: the instructions on shampoo bottles often say "Lather, rinse, repeat," with no instruction on when to stop. Computers, after all, take things literally; programmers, perforce, must learn to do the same. Victor Raskin, *Verbal Play in Computer Jokes*, BELB, http://www.belb.info/studenti/mincheva/verbal%20play.htm [https://perma.cc/662L-P4LB].

B. Bugginess

Consider the following minor variant of the first programming fragment shown above, but where the two storage locations are called "k" and "l" instead of "a" and "b":

```
int k, l;                /* Reserve two storage locations */

scanf("%d", &k);         /* Read the first value */
scanf("%d", &l);         /* Read the second */
printf("%d\n", k + 1);   /* Print their sum */
```

In fact, the two are *not* identical; in the second fragment, the sum printed is not of the two values, but of the first value, "k", and the number one.[98] Depending on the font used, the difference between the digit one and the 12th letter of the alphabet can be extraordinarily hard to spot. A programmer whose eyes see the comment is likely to glance only fleetingly at the actual code, and thus not spot the error. Errors—and sabotage—from similarly confusing patterns are by no means unknown in the real world.[99]

Simple bugs can have drastic consequences. A missing hyphen contributed to the loss of Mariner 1, the first space probe launched toward Venus.[100] The first launch of the Ariane 5 rocket, after $7 billion and a decade of development, failed because of a simple software error.[101] The first launch of the Space Shuttle Columbia, before a worldwide live television audience, was aborted on the launchpad due

---

[98] One of the authors of this article copied and pasted the code fragments above into an actual program, to verify that the behavior is as stated.

[99] In what is likely the best-documented sabotage incident, Princeton computer science professor Ed Felten described what was an apparent attempt to plant a back door—a deliberate security hole known only to its authors—in Linux, relying on just such a trick. Ed Felten, *The Linux Backdoor Attempt of 2003*, FREEDOM TO TINKER (Oct. 9, 2013), https://freedom-to-tinker.com/2013/10/09/the-linux-backdoor-attempt-of-2003/ [https://perma.cc/5KLY-3J45].

[100] David R. Williams, *Mariner 1*, NASA SPACE SCI. DATA COORDINATED ARCHIVE, https://nssdc.gsfc.nasa.gov/nmc/spacecraft/display.action?id=MARIN1 [https://perma.cc/2T5J-PMX2].

[101] James Gleick, *Little Bug, Big Bang*, N.Y. TIMES MAG. (Dec. 1, 1996), https://www.nytimes.com/1996/12/01/magazine/little-bug-big-bang.html [https://perma.cc/4R2K-QGL3].

to a bug.[102] Another notable example was a misplaced "break" statement[103] that led to the failure of most of AT&T's long distance network.[104] These bugs were immediately visible. Some bugs, though, are silent; one must look (and look hard) for them.[105]

Bugs are an omnipresent hazard in software. All non-trivial software packages have bugs;[106] even large-scale efforts to prevent them have their limits. Three of us wrote:

> [I]t is important to know a fundamental tenet of software engineering: bugs happen. In his classic *The Mythical Man-Month*, Frederick Brooks explained why:
>
> > First, one must perform perfectly. The computer resembles the magic of legend in this respect, too. If one character, one pause, of the incantation is not strictly in proper form, the magic doesn't work. Human beings are not accustomed to

---

[102] John R. Garman, *The "BUG" Heard 'Round the World: Discussion of the Software Problem Which Delayed the First Shuttle Orbital Flight*, 6 ACM SIGSOFT SOFTWARE ENGINEERING NOTES, Oct. 1981, at 3.

[103] In computer programs, a fundamental construct is "looping": repeated execution of a segment of code, generally until some condition is met. A "break" statement is one way to terminate execution of a loop. *See generally* KERNIGHAN & RITCHIE, *supra* note 95.

[104] BRUCE STERLING, THE HACKER CRACKDOWN 35 (1992) ("The 'break' was *supposed* to 'break' the 'if clause.' Instead, the 'break' broke the 'switch' statement.").

[105] S*ee, e.g.*, the descriptions of the Juniper and telnet issues, *infra* Section II.C.

[106] The problem of buggy code has been known since the dawn of computing. STERLING, supra note 105 (internal citations omitted). *See also* Tim Menzies & Thomas Zimmermann, *Software Analytics: What's Next?*, IEEE SOFTWARE, Sept.-Oct. 2018, at 64, 64-65 ("As soon as people started programming, it became apparent that programming was an inherently buggy process. Maurice Wilkes, speaking of his programming experiences in the early 1950s, recalled the following: . . . the realization came over me with full force that a good part of the remainder of my life was going to be spent in finding errors in my own programs."). By the early 1970s, researchers had developed formulas to predict the number of bugs in a program, based on its size and complexity. *Id.* at 65; *see also* Ronald L Rivest, *On the Notion of "Software Independence" in Voting Systems*, 366 PHIL. TRANSACTIONS ROYAL SOC'Y A 3759, 3760 (2008) ("Finding all errors in a large system is generally held to be impossible in general or else highly demanding and extremely expensive. Our ability to develop complex software vastly exceeds our ability to prove its correctness or test it satisfactorily within reasonable fiscal constraints.").

> being perfect, and few areas of human activity demand it. Adjusting to the requirement for perfection is, I think, the most difficult part of learning to program.

> Because computers, of course, are dumb—they do exactly what they are told to do—programming has to be absolutely precise and correct. If a computer is told to do something stupid, it does it, while a human being would notice there is a problem. A person told to walk 50 meters then turn left would realize that there was an obstacle present, and prefer the path 52 meters down rather than walking into a tree trunk. A computer would not, unless it had been specifically programmed to check for an impediment in its path. If it has not been programmed that way—if there is virtually any imperfection in code—a bug will result. The circumstances which might cause that bug to become apparent may be rare, but it would nonetheless be a bug.[107]

Although it may be hard for non-programmers to count individual bugs or the result of efforts to prevent them, computer security vulnerabilities—which are mostly due to buggy code—are generally more visible.[108]

---

[107] Steven M. Bellovin et al., *Lawful Hacking: Using Existing Vulnerabilities for Wiretapping on the Internet*, 12 NW. J. TECH. & INTELL. PROP. 1, 27 (2014).

[108] All large software products have a "ticketing system" where bug reports and feature requests are noted. In principle, this could be used to assess the bug rate, and many companies do this internally. However, this information is often carefully guarded. Ticketing systems for open source software are often public; however, distilling the data down to useful information—eliminating the duplicates, figuring out which entries are user error and hence incorrect, even deciding what is an actual bug and what is simply code that doesn't work quite as desired, even if that is the intended behavior—requires a great deal of effort. Clint Fontanella, *What's a Ticketing System?*, HUBSPOT (July 14, 2020), https://blog.hubspot.com/service/ticketing-system [https://perma.cc/LK8Y-7MY4]. Furthermore, it is unclear if the behavior of open source developers is similar enough to that of commercial developers. *See, e.g.*, Steven M. Bellovin, *The Open Source Quality Challenge*, SMBLOG (Apr. 29, 2009), https://www.cs.columbia.edu/~smb/blog/2009-04/2009-04-29.html [https://perma.cc/7TN3-HU53]; *see also* Brooks, *supra* note 87.

Microsoft provides an excellent case study. By 2001, the company realized it was in the throes of a security crisis. "'No Microsoft executive could have any conversation with any enterprise customer about anything but Microsoft's bad security,' said Steve Lipner, who was the company's director of security assurance at the time."[109] The Gartner Group warned its clients against using IIS, Microsoft's web server.[110] In response, in 2003, the company initiated a massive effort to produce more secure software[111] and had all of its 8,500 Windows developers take security training.[112] Within months the effort showed notable successes. Steven Lipner, who was Microsoft's director of engineering strategy, told the *New York Times*, "Some of the tougher security standards . . . have shown measurable improvement in Windows Server 2003, which shipped earlier this year. The number of security vulnerabilities detected so far is half as many as at this stage after the release of Windows Server 2000."[113] Microsoft's security development process has vastly improved the security of the company's products (a measure of its effectiveness is that the process has been emulated by Cisco and Adobe).[114] Nonetheless, security flaws (again, due to bugs) still occur: in August 2018, "Microsoft pushed 17 updates to fix at least 60 vulnerabilities in Windows and other software, including two 'zero-day' flaws that attackers were already exploiting before Microsoft issued patches to fix them."[115] In other words,

---

[109] LANDAU, *supra* note 65, at 64.
[110] Jack Kapica, *Gartner Slams Microsoft IIS Server,* GLOBE & MAIL (Sept. 25, 2001), https://www.theglobeandmail.com/technology/gartner-slams-microsoft-iis-server/article22621594/ [https://perma.cc/8ZW2-EHZ6].
[111] Steve Lohr, *Fixing Flaws, Microsoft Invites Attack*, N.Y. TIMES (Sept. 29, 2003), https://www.nytimes.com/2003/09/29/business/fixing-flaws-microsoft-invites-attack.html [https://perma.cc/5HWN-4YEX] ("At Microsoft, much more time is now being set aside in the design cycle of products for security considerations, a mandate approved by senior management this spring. 'There is a shift from mainly an emphasis on working features to an emphasis on trustworthy and secure computing,' said Steven B. Lipner, director of security engineering strategy at Microsoft.").
[112] *See* LANDAU, *supra* note 65, at 64.
[113] Lohr, *supra* note 111.
[114] Tim Rains, *The Secret of the SDL*, MICROSOFT (July 2, 2014), https://cloudblogs.microsoft.com/microsoftsecure/2014/07/02/the-secret-of-the-sdl/ [https://perma.cc/9A4Z-W67H].
[115] Brian Krebs, *Patch Tuesday, August 2018 Edition*, KREBS ON SECURITY (Aug. 18, 2018), https://krebsonsecurity.com/2018/08/patch-tuesday-august-2018-edition/ [https://perma.cc/2K6H-5MER].

although good software development processes are a significant help, they are not a panacea; bugs (and security holes) have nevertheless persisted, more than fifteen years after Microsoft began its initiative.

C. Testing and Assurance

Naturally, software vendors are aware of this problem and try to find and eliminate bugs. The most common method is testing, which involves feeding assorted sample inputs to a program to see if the correct answer is produced. [116] This, however, has limitations. The most fundamental issue with testing was noted decades ago by Edsger Dijkstra, an esteemed computer scientist: "Program testing can be used to show the presence of bugs, but never to show their absence!"[117] That is, a test case might yield an incorrect result, thereby demonstrating a bug; however, it is impossible to tell if there is a bug that has not been triggered by a particular set of inputs: there are far too many possible situations to try them all. In fact, it is mathematically impossible to find all failures.[118]

Some problems fundamentally cannot be detected simply by testing. The most obvious example is encryption. By definition, the output of a good encryption algorithm is indistinguishable from a random string of bits.[119] Indeed, if an encryption algorithm's output can be distinguished

---

[116] *See generally* PAUL AMMANN & JEFF OFFUTT, INTRODUCTION TO SOFTWARE TESTING (Cambridge U. Press, 2d ed. 2017).

[117] Edsger W. Dijkstra, *Structured Programming*, *in* SOFTWARE ENGINEERING TECHS. 65, 66 (J. N. Buxton & B. Randell eds., 1970), http://homepages.cs.ncl.ac.uk/brian.randell/NATO/nato1969.PDF.

[118] AMMANN & OFFUTT, *supra* note 116, at 20 ("the problem of finding all failures in a program is undecidable."). "Undecidable" is a technical term in computer science and mathematics; it means that no possible algorithm can always find the correct answer. *Undecidability*, U. OF ROCHESTER, https://www.cs.rochester.edu/u/nelson/courses/csc_173/computability/undecidable.html [https://perma.cc/RV9A-759Z].

[119] The notion that ciphertext should be random is due to C. E. Shannon's article *Communication Theory of Secrecy Systems*, but his phrasing is quite mathematical and does not quite match current usage. *See* C.E. Shannon, *Communication Theory of Secrecy Systems*, 28 BELL SYS. TECH. J. 656, 656-715 (1949). A more accessible reference simply assumes that randomness is a requirement for a good cipher. JUAN SOTO, JR., U.S. DEPT. COM., RANDOMNESS TESTING OF THE ADVANCED ENCRYPTION STANDARD CANDIDATE ALGORITHMS (1999) ("One of the criteria used to evaluate the Advanced Encryption Standard candidate

from a random string, that algorithm is considered seriously flawed.[120] However, the presence of randomness is not itself sufficient for correctness; the encryption keys must also be chosen properly.[121] That is a result of the principle established by Auguste Kerckhoffs in 1883: one cannot assume that the encryption algorithm is secret.[122] Therefore, the security of an encryption algorithm must lie in the secrecy of the key, and ensuring that secrecy is thus critical to an encryption's algorithm working appropriately.[123] Unfortunately, testing will not necessarily reveal that secrecy. Two examples better illustrate this point. The first example concerned an encrypted version of the Telnet protocol.[124] Due to a bug in the key generation mechanism, most of the time a constant key would be used.[125] Anyone who knew this key could

---

algorithms was their demonstrated suitability as random number generators. That is, the evaluation of their output utilizing statistical tests should not provide any means by which to computationally distinguish them from a truly random source."). The actual modern definition of encrypting a message, though quite mathematical, assumes that an ideal encryption algorithm generates a sequence of random bits. *See*, *e.g.*, Shafi Goldwasser & Silvio Micali, *Probabilistic Encryption*, 28 J. COMPUT. & SYS. SCI. 270, 272 (1983) ("More specifically, a binary message will be encrypted bit-by-bit as follows: a "0" is encoded by randomly selecting an x such that $B(x) = 0$ and a "1" is encoded by randomly selecting an x such that $B(x) = 1$. Consequently, there are many possible encodings for each message. However, messages are always uniquely decodable.").

[120] Goldwasser & Micali, *supra* note 119, at 271.

[121] Today, an encryption key is generally a very large number; depending on the particular encryption algorithm used, the keys can be tens or even hundreds of digits long. Rob Stubbs, *Classification of Cryptographic Keys*, CRYPTOMATHIC (Feb. 19, 2018), https://www.cryptomathic.com/news-events/blog/classification-of-cryptographic-keys-functions-and-properties [https://perma.cc/JE5Y-ZSCP].

[122] Auguste Kerckhoffs, *La Cryptographie Militaire*, 9 J. DES SCI. MILITAIRIES 5, 12 (1883) (Fr.) ("Il faut qu'il n'exige pas le secret, et qu'il puisse sans inconvénient tomber entre les mains de l'ennemi." ["The system must not require secrecy and can be stolen by the enemy without causing trouble"]).

[123] *Id.*

[124] Telnet, a now-obsolescent protocol, was primarily used for command-line access to remote computers. Users thus had to enter a login name and password, hence the need for encryption. Tyson Supasatit, *Can You Encrypt Telnet?*, EXTRAHOP (Aug. 14, 2019), https://www.extrahop.com/company/blog/2019/telnet-security-how-to-encrypt-telnet-sessions/ [https://perma.cc/G4P5-2D2F].

[125] The fact of the problem was announced in CERT Advisory CA-1995-03, but no details were provided, though the true situation was well known in the security community. CERT DIVISION, CARNEGIE MELLON U., 1995 CERT ADVISORIES 20 (Carnegie Mellon U. Software Engineering I. ed., 2017). According to a comment on a blog, the actual cause was deliberately obfuscated. John Gilmore, Comment to *Vendors Are Bad for Security*, LINKS: BEN LAURIE BLATHERING (May 13, 2008, 2:09 PM), https://www.links.org/?p=327 [https://perma.cc/8NV7-WKSE].

trivially decrypt the traffic.[126] Simple testing would not have found the problem; the output would have looked random despite the constant key.[127] Only an examination of the code itself would have demonstrated the problem that the same encryption key was being used for all inputs.

The second incident was more sinister. The problems started with an algorithm for generating random numbers that was originally put forth by the National Security Agency (NSA). In NSA's words, the agency had "covertly influence[d] and/or overtly leverage[d] . . . commercial products designs . . . mak[ing] the systems in question exploitable."[128] The system in question was Dual EC_DRBG, a random number generator that provided bits for an encryption key.[129] Recall Kerckhoff's

---

[126] CERT DIVISION, *supra* note 125; Gilmore, *supra* note 125.

[127] Because of technical details of how the encryption actually worked, multiple encryptions of the same data would look different, despite the fact that the same key was used. *See generally*, Mihir Bellare & Phillip Rogaway, Introduction to Modern Cryptography (2005) (collection of class notes), available at https://web.cs.ucdavis.edu/~rogaway/classes/227/spring05/book/main.pdf; JASON GARMAN, KERBEROS (2003); PETER LOSHIN, TCP/IP CLEARLY EXPLAINED 221-35 (4th ed. 2002).

[128] *Secret Documents Reveal N.S.A. Campaign Against Encryption,* N.Y. TIMES (Sept. 5, 2013), https://archive.nytimes.com/www.nytimes.com/interactive/2013/09/05/us/documents-reveal-nsa-campaign-against-encryption.html [https://perma.cc/AYC3-FM5Z].

[129] Random numbers are used to generate encryption keys but finding genuinely random bits—that is to say, bits that are hard to predict—is a computationally difficult problem. Consequently, a common method instead is to start with some truly random bits, then use a mathematical function to expand these into a longer sequence of pseudo-random bits. The Elliptic Curve Digital Random Bit Generator (Dual EC-DRBG) is one such function; it uses elliptic curves to do so. The curve relies on two default parameters. The choice of these parameters is crucial, for anyone knowing the arithmetic relationship between the parameters would be able to predict the "random" numbers generated by the algorithm—making the random numbers highly nonrandom indeed. In particular, if encryption keys were based on these "random numbers," anyone knowing the relationship between the two parameters would have a significant head start in guessing the bits of the encryption key. Despite early concerns about a possible cryptographic backdoor—such a backdoor permits a much more efficient search for the key, making the encryption algorithm far less secure than it would otherwise be—in Dual EC_DRBG, the National Institute for Standards and Technology recommended this method for generating random numbers. *See* DAN SHUMOW & NIELS FERGUSON, ON THE POSSIBILITY OF A BACK DOOR IN THE NIST SP800-90 DUAL EC PRNG 2-7 (2007), http://rump2007.cr.yp.to/15-shumow.pdf; *see also* ELAINE BARKER & JOHN KELSEY, U.S. DEP'T. COM., NIST SP 800-90, RECOMMENDATION FOR RANDOM NUMBER GENERATION USING DETERMINISTIC RANDOM BIT GENERATORS 13-67 (2007) (withdrawn in Jan. 2012 and succeeded by SP 800-90A). This was after RSA Security LLC had made the algorithm the default random bit generator in its popular BSafe encryption toolkit. Joseph Menn, *Exclusive: Secret Contract Tied NSA and Security Industry Pioneer*, REUTERS (Dec. 20, 2013, 4:05 PM) https://www.reuters.com/article/us-usa-security-rsa/exclusive-secret-contract-tied-nsa-and-

Principle; the security of the system lies in the security of the key. But the NSA effort meant that the bits of the key were predictable and thus that any information encrypted through that system could be easily decrypted by the NSA.

In principle, the communication would, however, be secure against other interceptors, which did not know the relationship between a crucial two parameters—and thus could not easily guess the random bits and determine the encryption key. This situation did not last. A Juniper firewall used the DUAL_EC_DRBG random number generator. Apparently, someone switched the crucial parameters in the standard DUAL_EC_DRBG algorithm with other numbers; possibly, this allowed the party who did this to have its own back door.[130] But simple testing would not detect the problem. The output would, after all, appear to be properly encrypted.

But programmers had added a second back door to the Juniper firewall[131] that allowed anyone who knew a secret password to log into the firewall as a privileged user.[132] The password was "<<<

---

security-industry-pioneer-idUSBRE9BJ1C220131220. The algorithm was widely used, including in such applications as SSL/TLS, the protocol for securing web transmissions. Matthew Greene, *The Many Flaws of Dual EC-DRBG*, A FEW THOUGHTS ON CRYPTOGRAPHIC ENG'G (Sept. 18, 2013), https://blog.cryptographyengineering.com/2013/09/18/the-many-flaws-of-dualecdrbg/ [https://perma.cc/6YPJ-F4BA].

[130] Stephen Checkoway et al., *A Systematic Analysis of the Juniper Dual EC Incident*, *in* PROCEEDINGS OF THE 2016 ACM SIGSAC CONFERENCE ON COMPUTER AND COMMUNICATIONS SECURITY 468. The full story is vastly more complicated and includes other back doors and bugs. "The Dual_EC generated initial output that was supposed to then be run through the ANSI generator. The output from the second random generator would theoretically cancel out any vulnerabilities that were inherent in the Dual_EC output. . . . Except Juniper's system contained a bug, according to Willem Pinckaers, an independent security researcher in the San Francisco area who examined the system with Weinmann. Instead of using the second generator, it ignored this one and used only the output from the bad Dual_EC generator." Kim Zetter, *Researchers Solve Juniper Backdoor Mystery; Signs Point to NSA*, WIRED (Dec. 22, 2015, 1:29 AM), https://www.wired.com/2015/12/researchers-solve-the-juniper-mystery-and-they-say-its-partially-the-nsas-fault/ [https://perma.cc/2MEU-5EYW].

[131] Sean Gallagher, *Researchers Confirm Backdoor Password in Juniper Firewall Code*, ARS TECHNICA (Dec. 21, 2015, 1:15 PM), https://arstechnica.com/information-technology/2015/12/researchers-confirm-backdoor-password-in-juniper-firewall-code/ [https://perma.cc/H7YQ-GNHT].

[132] Checkoway et al., *supra* note 131, at 468. It is not known if the two back doors were inserted by the same party or by different parties. Zetter, *supra* note 131.

%s(un='%s') = %u," a value that resembles a string that might be used for debugging.[133] There is no conceivable testing regimen that would have found this flaw.

There are still other ways that programming bugs may lay hidden. Sometimes, bugs can be very rare or highly unlikely because they are timing-dependent. Consider the space shuttle bug.[134] The details are complex; let it suffice to say that it would only occur one out of sixty-seven times the shuttle's computers were booted.[135] The AT&T network flaw would only be triggered when certain events occurred within 1/100 of a second of each other.[136] Testing would be very unlikely to uncover this issue.

There have been inexplicable problems in forensic software too. In one case in Pennsylvania, three different experts concluded there was a DNA match to the defendant, but their probability estimates—all derived from software—differed wildly.[137] It is impossible to understand such differences without access to the source code, nor, for that matter, to know if the problem is a different mathematical model or simply code that implements the model incorrectly.

The limitations of testing have, of course, led to the development of other methods for finding and eliminating software bugs. Those have helped but have not solved the problem. In a previous work, several of us summarized the issue this way:

> We will not recount the myriad techniques other than testing that have been tried in an effort to eliminate bugs;

---

[133] Gallagher, *supra* note 131; *see also*, *infra* Section II.D. (discussing programmer blindness).
[134] *See* Garman, *supra* note 102.
[135] *Id.* at 9.
[136] *See* STERLING, *supra* note 104, at 48.
[137] Commonwealth v. Foley, 38 A.3d 882, 887 (2012) ("The experts differed in their estimates of the probability that someone other than Foley would possess DNA matching the DNA found in the sample—Conway testified that the probability that another Caucasian could be the contributor was 1 in 13,000; Dr. Cotton testified that the probability was 1 in 23 million; and Dr. Perlin testified that it was 1 in 189 billion."); *see also* Stephanie J. Lacambra et al., *Opening the Black Box: Defendants' Rights to Confront Forensic Software*, CHAMPION, May 2018, at 28, https://www.eff.org/files/2018/07/30/champion_article_-_lacambra_forensic_software_may_2018_07102018.pdf.

> let it suffice to say there have been many. These include formal mathematical methods, better programming and debugging tools, different organizational and procedural schemes, improved programming languages, and more. Many of these ideas have helped, but none have proved a panacea. The ability to produce error-free code is the Holy Grail of systems development: heavily desired but unattainable.[138]

In summary, software as it exists today (and likely for the foreseeable future) will always have bugs. Consequently, criminal defendants need access to source code to safeguard their constitutional rights.

D. Adversarial Audits

In light of the axiomatic notion that software will always have bugs, the question then becomes how courts should address this problem. Our basic thesis is that adversarial audits—examination and testing of software by defendants—is necessary for a fair trial. Put another way, we assert that outside testing will find flaws that the vendor did not. Why should this be?

In one sense, this is a statement that does not require proof. The American legal system is fundamentally based on the premise that the adversarial system is the best way to reach the truth. Indeed, as legal scholar Ralph Grunewald has written, "[t]he Supreme Court shares the view that facts are best proven dialectically through a complex process of persuasion and holds that truth 'is best discovered by powerful statements on both sides of the question.'"[139] Furthermore, "[t]he very premise of our adversary system of criminal justice is that partisan advocacy on both sides of a case will best promote the ultimate objective that the guilty be convicted and the innocent go free."[140]

---

[138] Bellovin et al., *supra* note 107, at 28.
[139] Ralph Grunewald, *Comparing Injustices: Truth, Justice, and the System*, 77 ALB. L. REV. 1139, 1156 (2014) (quoting United States v. Cronic, 466 U.S. 648, 655 (1984)).
[140] Herring v. New York, 422 U.S. 853, 862 (1975).

Even in technical matters, one side's expert witness testimony is rebutted by cross-examination and the other side's experts, rather than having the judge appoint and question a neutral witness. There is no *a priori* reason why software issues should be handled differently. There are also several technical reasons to treat software witnesses in the same manner as human expert witnesses. These include the inability of programmers to see their own errors, the fact that the program specifications[141] themselves may be in error (and thus although the program matches the specifications, the program does not do what it should be doing), inadequacy of testing data, and, finally, the proof history: too many programs viewed as correct have been shown to function incorrectly when subjected to adversarial audits that have uncovered serious errors. We discuss each of these phenomena in turn.

The first is a phenomenon we will call "programmer blindness." Just as writers are often bad at proofreading their own text, programmers are bad at reading their own code. Their eyes will skip over errors. This deficiency is often countered by peer review when someone else reads over the code before it is put into the production system,[142] but even that is not a fail-safe method. It is often the case that peers are not truly independent reviewers because programmers often have similar training—and thus tend to make the same mistakes.[143]

Just as there are excellent editors and proofreaders, there are also people who are skilled at avoiding conventional assumptions about software. This ability is at the heart of computer security, whether one is trying to protect a system or break into it.[144] Few companies have such people on their development teams, but a diligent defense attorney could and should seek one out. Indeed, as we increasingly rely on software for providing evidence and making decisions, such a person should be part of any criminal defense team.

---

[141] *See* discussion of erroneous specifications *infra* Section II.D.

[142] *See, e.g.*, Peter C. Rigby & Christian Bird, *Convergent Contemporary Software Peer Review Practices*, *in* PROCEEDINGS OF THE 2013 9TH JOINT MEETING ON FOUNDATIONS OF SOFTWARE ENGINEERING 202, 203 (2013).

[143] *See generally* John C. Knight & Nancy G. Leveson, *An Experimental Evaluation of the Assumption of Independence in Multiversion Programming*, 12 IEEE TRANSACTIONS ON SOFTWARE ENG'G 96, 97 (1986).

[144] *See generally* STEVEN M. BELLOVIN, THINKING SECURITY (2016).

Second, software testing is done according to the program's requirements and specifications.[145] In other words, a party external to the programming team defines the objective: what the inputs and outputs should look like, how errors should be handled, how fast the program must run, etc. The tester's job is to take each requirement and verify that it is met. Often, though, the specifications themselves are faulty. In one study of fifty actual security errors, researchers found that twenty two of them were due to errors in the requirements or specifications.[146] Ordinary testing will not uncover such problems because the program matches the specifications; the problem is that the specifications themselves are incorrect.[147]

The third reason is the nature of testing in the software development process. Software testing is not random; rather, the tester carefully crafts inputs. This testing is done in two ways. The older type is requirements-

---

[145] This is well understood by practitioners, though not always discussed in the academic literature. *But see* Gilles Bernot et al., *Software Testing Based on Formal Specifications: A Theory and a Tool*, 6 SOFTWARE ENG'G J. 387, 387 (1991) ("With the emergence of formal specification languages, it becomes possible to also start from the specification to define some testing strategies in a rigorous and formal framework.").

[146] *See* Carl E. Landwehr et al., *A Taxonomy of Computer Program Security Flaws*, 26 ACM COMPUTING SURV. 211, 216 (1994).

[147] Because the specifications for commercial programs are rarely published, and because these specifications are often extremely technical, we generally learn of such errors only when a security hole or serious bug is found that can be traced back by outsiders to a specification. A recently discovered flaw in Microsoft Windows—a flaw that is more than 20 years old—provides just such an example. In Windows, a program running in one window can send a message to a program running in another window. These messages can be things like "resize yourself," "terminate,", "open this URL," "here is some input," and more. Since some programs have more privileges than others—software installers need high privileges to change the system, while browsers (which are vulnerable to nasty web pages) have fewer permissions than normal—the ability to send such messages is normally restricted. However, Microsoft implemented the "Text Services Framework" to handle such functionality as Chinese language input. Obviously, a multilingual input program must be able to talk to all other running programs, so there were no restrictions for any such messages. This in turn allowed for improper messages to be sent by an attack program that pretended to be part of this framework, which in turn led to security problems. *See* Jim Salter, *A Look at the Windows 10 Exploit Google Zero Disclosed this Week*, ARS TECHNICA (Aug. 15 2019, 6:45 AM), https://arstechnica.com/information-technology/2019/08/a-look-at-the-windows-10-exploit-google-zero-disclosed-this-week/ [https://perma.cc/9ZXF-XBPB].

based: try inputs that stress or exceed requirements.[148] This approach tries to find what are known as "boundary conditions," which are places where the behavior of the program is likely to change.[149] For example, if a certain input field is allowed to be 200 characters long, a tester might try 199, 200, 201, and 2000 character inputs. Similarly, because 2,147,483,647 is the largest number some programs can handle, 2,147,483,648 is a useful test.[150]

Another testing strategy is code-based. There are certain technical aspects of the code that suggest certain inputs should be tried. For example, it is desirable to exercise as much of the program as possible. A mechanism known as a "code coverage tool"[151] tells the tester which lines of the software have or have not been exercised. Referring back to our subtraction example,[152] such a tool could verify that both branches of the "if" statement were tested. However, modern software is so complex that complete coverage during testing is not possible because there are just too many possibilities. In a testing trial, though, certain input values might be fixed to allow testers to vary other inputs in a way that would not have been feasible during the development process.

The Draeger Alcotest 9510 breathalyzer provides an example of how a code-based testing strategy affects criminal defendants. According to

---

[148] *See* AMMANN & OFFUTT, *supra* note 116, at 150 ("The input *domain* is defined in terms of the possible values that the input parameters can have. . . . The tester does not need to understand the implementation; everything is based on a description of the inputs.").

[149] The optimal strategy for generating good test data is in fact an active research area. *See, e.g.*, Mats Grindal et al., *Combination Testing Strategies: A Survey*, 15 SOFTWARE TESTING, VERIFICATION, AND RELIABILITY 167, 167 (2005) ("A literature search has revealed 16 different combination strategies described through more than 40 papers published between 1985 and 2004.").

[150] Generally, modern computers use 32-bit integers. One bit is used to indicate positive or negative; the range of integer values therefore ranges from -2,147,483,648 to 2,147,438,647. *See* Nickolas T. Lanza, *How Do You Get the Maximum and Minimum Values for Integer Data Types Based on the Operating System?*, MEDIUM (Jan. 29, 2018), https://medium.com/@nickolasteixeira/how-to-explain-to-my-wife-what-i-do-how-do-you-get-the-maximum-and-minimum-values-for-integer-befdc263a3a2 [perma.cc/9EJ2-26F7].

[151] *See, e.g.*, Mustafa M. Tikir & Jeffrey K. Hollingsworth, *Efficient Instrumentation for Code Coverage Testing*, 27 SIGSOFT SOFTWARE ENG'G NOTES 86, 86 (2002). The purpose of testing is, of course, to ensure that the code is correct. If test cases do not exercise certain lines of code, then the test cases cannot have verified whether those lines of code are correct.

[152] *See supra* Section II.A.

press reports, the unit is sensitive to variations in the outside temperature.[153] An adversarial examination of such a device could test it at the observed temperature during the police stop, rather than what was available in the vendor's test lab.

The fourth reason why software issues should be treated differently is purely empirical. In other scenarios, adversarial testing has found flaws in software that had been certified by outside parties. The best-documented example is voting machines, where outside auditors have *always* found flaws.

One of the first published independent audits of election systems was of leaked code to a Diebold electronic voting system.[154] The authors' conclusion, after analysis, was blunt: "Our analysis shows that this voting system is far below even the most minimal security standards applicable in other contexts."[155] They also noted problematic aspects that could only be learned by looking at the complete source code.[156] For example, they found no indications that any formal requirements documents or ticket tracker were used even though both of these are, as noted, part of most structured development processes.[157]

Another well-known example is California's "top-to-bottom review" of the electronic voting machines that were to be used in its 2008 elections. Independent teams were contracted to examine the system source code for vulnerabilities that might not have been discovered in the certification process.[158] The results cast significant doubt on the

---

[153] *See* Zack Whittaker, *Researchers Say a Breathalyzer Has Flaws, Casting Doubt on Countless Convictions*, ZDNET (May 10, 2018), https://www.zdnet.com/article/draeger-breathalyzer-breath-test-convictions/ [https://perma.cc/6QER-ZKJN].
[154] T. Kohno et al., *Analysis of an Electronic Voting System*, *in* PROCEEDINGS OF THE IEEE SYMPOSIUM ON SECURITY AND PRIVACY 27, 28 (2004), available at https://ieeexplore.ieee.org/document/1301313.
[155] *Id.* at 27.
[156] *Id.* at 37.
[157] *Id.* at 34.
[158] Cal. Secretary of State, Top-to-Bottom Review of Electronic Voting Systems Certified for Use in California Elections, (draft for public comment Mar. 22, 2007), https://www.sos.ca.gov/elections/ovsta/frequently-requested-information/top-bottom-review [https://perma.cc/2DM3-F2GJ] (archiving documents related to the review). The review was initiated by then-Secretary of State Debra Bowen, pursuant to her responsibilities

integrity of every one of the systems. A review of one evaluated system explained that "[v]irtually every important software security mechanism is vulnerable to circumvention."[159] Moreover, the previous certification processes[160] failed to catch bugs that one would expect to have been caught: "There is evidence in the documentation of vendor-initiated source code reviews as part of the independent testing process, however the broken cryptography architecture indicates that more audits by qualified security experts are required."[161]

Subsequent adversarial examinations of voting systems have shown similar results, as summarized by author Matt Blaze in his 2017 Congressional testimony, which reported that every current voting system examined over the course of a weekend at the DEFCON hacking conference was found to suffer from exploitable vulnerabilities due to unpatched or undetected software defects.[162]

Computerized voting systems provide interesting parallels to the software systems used to collect and process evidence in criminal cases. There is broad consensus among elections experts that modern software

---

in accordance with California law. *Id.*

[159] MATT BLAZE ET AL., SOURCE CODE REVIEW OF THE SEQUOIA VOTING SYSTEM 82 (2007), https://votingsystems.cdn.sos.ca.gov/oversight/ttbr/sequoia-source-public-jul26.pdf.

[160] There are no national standards for the certification of voting systems. Each state jurisdiction sets its own requirements, which are often administered on an ad hoc basis. California requires certification by the Secretary of State, Cal. Elec. Code § 19006(a) (West 2014) ("All voting systems be certified or conditionally approved by the Secretary of State, independent of voluntary federal qualification or certification, before they are used in future elections to ensure that the voting systems have the ability to meet accuracy, accessibility, and security standards."). The Federal standards are voluntary. *See* Help America Vote Act of 2002, 52 U.S.C. § 20961(b)(1) (2002) ("The Development Committee shall assist the Executive Director of the Commission in the development of the voluntary voting system guidelines"); *id.* § 21001(d) ("Nothing in this subpart may be construed to require a State to implement any of the voluntary voting system guidelines or any of the voluntary guidance adopted by the Commission with respect to any matter as a condition for receiving a requirements payment.").

[161] BLAZE ET AL., *supra* note 159, at 42.

[162] *Cybersecurity of Voting Machines: Hearing Before the H. Comm. on Oversight and Gov't Reform, Subcomm. on Info. Tech., & Subcomm. on Intergovernmental Affs.*, 115 Cong. 11 (2017) (statement of Matt Blaze, Assoc. Professor, U. Pa.).

systems are, by virtue of their design, too complex and unreliable to be relied upon for determining the outcomes of civil elections.[163]

The consensus that software is inherently unreliable and suspect has led elections experts to advocate for election system designs that do not depend on software for the correctness of election outcomes. A widely recognized statement of this criteria was proposed by computer scientist Ronald Rivest as Software Independence.[164] In particular:

> A voting system is strongly software-independent if an undetected change or error in its software cannot cause an undetectable change or error in an election outcome, and moreover, a detected change or error in an election outcome (due to change or error in the software) can be corrected without re-running the election.[165]

At first blush, a requirement for software independence might appear to preclude the use of any software (or computers) in elections altogether. But that is not necessarily the case. Software independence simply requires that any software-based system be designed in a way that allows for recovery of correct results even if the software had failed in some way. For example, a voting system that employs paper ballots (marked by a voter) might perform an initial tally by computerized scanners, but still retain the ability to use the original paper ballots (which reflect the true intent of the voters) for recounts and audits, which can be conducted by hand and without dependence on the software.[166] Such a system uses software, yet is still software independent under this definition.

---

[163] *See generally* NAT'L ACADS. OF SCIS., ENG'G, AND MED., SECURING THE VOTE: PROTECTING AMERICAN DEMOCRACY (2018), https://www.nap.edu/catalog/25120/securing-the-vote-protecting-american-democracy [https://perma.cc/B9PU-FDEQ].
[164] *See generally* Ronald L. Rivest & John P. Wack, *On the Notion of 'Software Independence' in Voting Systems*, 366 PHIL. TRANSACTIONS SERIES A, MATHEMATICAL, PHYSICAL, AND ENG'G SCIS. 3759 (2008), https://people.csail.mit.edu/rivest/RivestWack-OnTheNotionOfSoftwareIndependenceInVotingSystems.pdf.
[165] *Id.* at 3763.
[166] *See id.*

Flaws in voting systems threaten the integrity of an election; flaws in DNA analysis software can put the wrong person in jail. In one widely publicized incident, defense attorneys in New York City were granted access to the source code to the Forensic Statistical Tool ("FST") developed by the office of the chief medical examiner.[167] On a motion from ProPublica, the software was made publicly available and affidavits from a defense expert witness were unsealed.[168] He explained that by examining the source code he was able to learn of behavior in the match calculation that was never documented anywhere:

> An instance of undocumented behavior of the FST program is noted in the calculation of likelihood ratios performed in "Comparison.cs". A routine included in the source code file ("Comparison" class), named "CheckFrequencyForRemoval" appears to perform the following behavior:
> 1. Check all replicate (evidentiary) genotypes for any locus that contains alleles whose frequency sums to ≥ 0.97 in any of the four subpopulations ("Asian," "Black," "Caucasian," and "Hispanic).
> 2. Remove these loci from the likelihood ratio calculations.
> During this review, I encountered no notice, either intended or actual, provided to the user of FST that any loci were removed from the likelihood ratio calculation. I found no indication that this behavior is intended during my examination of FST-related publications and the FST Validation materials.[169]

---

[167] Lauren Kirchner, *Federal Judge Unseals New York Crime Lab's Software for Analyzing DNA Evidence,* PROPUBLICA (Oct. 20, 2017, 8:00 AM), https://www.propublica.org/article/federal-judge-unseals-new-york-crime-labs-software-for-analyzing-dna-evidence [https://perma.cc/4BPK-RUJD].
[168] Lauren Kirchner, *Traces of Crime: How New York's DNA Techniques Became Tainted,* N.Y. TIMES (Sept. 4, 2017), https://www.nytimes.com/2017/09/04/nyregion/dna-analysis-evidence-new-york-disputed-techniques.html [https://perma.cc/8A7T-MPFU].
[169] Memorandum in Support of Defendant's Motion in Limine, Exhibit C at 20, United States v. Johnson, S.D.N.Y 2016 (1:15-cr-00565-VEC).

Furthermore, the behavioral change in the match calculation was done after the validation study of the FST package: "There is at least one indication of a deviation in behavior between the version of FST used during its validation study and the version of FST provided to me." [170]

Beyond that example, there may be other lurking problems. Deviation from good coding practice is a red flag. Systems engineer Nathaniel Adams wrote:

> Martin Fowler describes code smells as, "A code smell is a surface indication that usually corresponds to a deeper problem in the system." In this sense, a smell is not a defect in itself but is a deviation from good coding practices, which can indicate underlying software defects. Coding conventions and style guides purposely prevent code smells. Coding practices such as writing long routines or complex classes can obfuscate underlying issues due to the difficulty of comprehending large segments of code.[171]

He goes on to note that "[t]he FST source code presents a number of basic smells such as routine length and complex classes."[172] In other words, the style of coding suggests that the program is below normal professional standards and may have other, not yet detected problems. It is extremely difficult to detect such "smells" without access to source code.

We thus see that, as an empirical matter, adversarial audits can uncover relevant issues in software relied upon in prosecutions. We are not aware of any studies of why this should be—why independently certified software should prove so vulnerable when subjected to an adversarial audit. We speculate that, in the absence of such audits, there is little basis for comparison other than price, and hence little opportunity for market forces to produce improvement. This is similar

---

[170] *Id.* at Exhibit A, at 9.
[171] *Id.* at Exhibit C, at 13 (citing Martin Fowler, *Code Smell,* MARTINFOWLER.COM (Feb 9, 2006), https://martinfowler.com/bliki/CodeSmell.html [https://perma.cc/9ZRP-AHG5]).
[172] *Id.*

to code developed in a competitive market, where base functionality is all-important and the winner is often the first player to the market: security often takes a back seat.[173] If price is in fact the primary differentiator, vendors' primary incentives are to reduce their costs, e.g., by reducing the cost of the entire software development process.

We do not mean to suggest that adversarial audits are a panacea. Nevertheless, experience shows that they do help.

E. Machine Learning

Machine learning algorithms raise a different set of issues.[174] Machine learning algorithms, popularly known as "artificial intelligence" or "AI," are at the heart of many of today's technologies, including all major search engines.[175] However, these algorithms have to be "trained," i.e., fed known initial data.[176] If the training data is deficient, the output of the algorithm will be incorrect.

This problem has arisen a number of times. Professor Ignacio Cofone wrote:

> An illustrative employment example is a machine learning system that Amazon recently developed to rank job candidates. The system displayed a significant bias against female candidates, justifiably triggering public outcry. Because the algorithm was trained using Amazon's existing hiring data under the idea that Amazon's current employee choices are a good proxy for

---

[173] NAT'L RSCH. COUNCIL ET AL., TRUST IN CYBERSPACE 67 (Fred B. Schneider ed., 1998) ("For software-intensive products, early arrival in the marketplace is often critical to success in that marketplace. This means that software development practice becomes distorted to maximize functionality and minimize development time, with little attention paid to other qualities. Thus, functionality takes precedence over trustworthiness.").

[174] See generally Steven M. Bellovin et al., *Privacy and Synthetic Datasets,* 22 STAN. TECH. L. REV. 1 (2019), for a description of machine learning.

[175] Gabriel Jiménez, *Artificial Intelligence Applications in Search Engines,* MEDIUM (Dec. 18, 2018), https://medium.com/aimarketingassociation/artificial-intelligence-applications-in-search-engines-437c57f8b265 [https://perma.cc/H77S-A55W].

[176] Bellovin et al., *supra* note 174, at 4.

> Amazon's desired employee choices, the algorithm reflected existing hiring practices. These practices, however, to the surprise of the algorithm's programmers, ended up being sexist.[177]

Microsoft had similar problems with its Tay "chatbot." A chatbot is designed to respond to users who send it messages, learning as it goes. However, Tay learned from the worst of the Internet, going "full Nazi" in less than 24 hours.[178] Microsoft realized that the problem was training data, saying "[i]n an emailed statement given later to Business Insider . . . : 'The AI chatbot Tay is a machine learning project, designed for human engagement. As it learns, some of its responses are inappropriate and indicative of the types of interactions some people are having with it.'"[179]

In other words, for software based on machine learning, defendants should have access not only to the source code but also to the training data. Without that data, it is impossible to understand what output a program might produce. In explaining problems with automated analysis of computer programs, Professor Kroll et al. wrote:

> Perhaps the most obvious approach is to disclose a system's source code, but this is at best a partial solution to the problem of accountability for automated decisions. The source code of computer systems is illegible to nonexperts. In fact, even experts often struggle to understand what software code will do: inspecting source code is a very limited way of predicting how a computer program will behave. Machine learning, one increasingly popular approach to automated decision making, is particularly ill-suited to source code analysis because it

---

[177] Ignacio Cofone, *Algorithmic Discrimination Is an Information Problem,* 70 HASTINGS L.J. 1389, 1397-98 (2019).

[178] James Vincent, *Twitter Taught Microsoft's AI Chatbot to Be a Racist Asshole in Less Than a Day*, VERGE (Mar. 24, 2016, 6:43 AM), https://www.theverge.com/2016/3/24/11297050/tay-microsoft-chatbot-racist [https://web.archive.org/web/20190312132945/https://www.theverge.com/2016/3/24/11297050/tay-microsoft-chatbot-racist].

[179] *Id.*

> involves situations where the decisional rule itself emerges automatically from the specific data under analysis, sometimes in ways that no human can explain.[180]

The same, of course, is true for evidentiary software.

III. Source Code and Constitutional Issues

Software and algorithms now play a critical role in many criminal prosecutions. Depriving such defendants access to the underlying code risks depriving defendants of their rights. For example, there are many ways for criminal defendants to challenge breathalyzer results, including attacks of the device's margin of error, its calibration, and the breath sample itself.[181] However, there are not many examples in which courts analyzed the use of algorithms and software, let alone whether any errors in specific program should be the basis for a challenge by criminal defendants. Indeed, courts have often been challenged by new technological developments.[182]

Some state courts have received challenges to convictions using various types of breathalyzers.[183] The defendants typically argued for the ability to examine the source code for these devices. They generally make two types of arguments. First, they argue that the source code must be provided pursuant to *Brady v. Maryland*, in which the Court held that prosecutors must provide defendants with all exculpatory evidence.[184] Second, defendants assert the failure to provide the source code violates the Confrontation Clause of the Sixth Amendment.[185] Third, they

---

[180] *See* Joshua A. Kroll et al., *Accountable Algorithms*, 165 U. PA. L. REV. 633, 638 (2017).
[181] Paul A. Clark, *The Right to Challenge the Accuracy of Breath Test Results Under Alaska Law*, 30 ALASKA L. REV. 1, 6-8 (2013).
[182] *See generally* Riley v. California, 573 U.S. 373 (2014); Kyllo v. United States, 533 U.S. 27 (2001).
[183] Cowley & Silver-Greenberg, *supra* note 39.
[184] Brady v. Maryland, 373 U.S. 83, 87 (1963) (holding that the prosecution must provide exculpatory evidence to a defendant). The case involved a murder committed by one of two men being tried in separate trials; the prosecution had a confession from the murderer but failed to share this evidence with the other.
[185] U.S. CONST. amend. VI; *see also* People v. Umpierre, 951 N.Y.S.2d 382, 383 (Sup. Ct. 2012); State v. Lindner, 252 P.3d 1033, 1036 (Ariz. Ct. App. 2010) (finding against the

maintain that denial of access to source code constitutes a due process violation.[186] We discuss each of these approaches in turn.

A. *Brady* Violations

The Supreme Court in *Brady* was very clear: the prosecution must provide exculpatory evidence to a defendant. In *Brady*, two people, the petitioner and another person, were charged with first degree murder, but tried separately.[187] At trial, Brady conceded his guilt in the murder, but argued that he did not kill the victim.[188] In preparation for trial, his defense attorney sought the extrajudicial statements by the other person.[189] The prosecution withheld one in which he admitted to committing the killing.[190] The Supreme Court held "that the suppression by the prosecution of evidence favorable to an accused upon request violates due process where the evidence is material either to guilt or to punishment, irrespective of the good faith or bad faith of the prosecution."[191]

In cases where software provides evidence, courts have had varying interpretations of *Brady*. In some sense, the issue is what constitutes "suppression"—for if the prosecution does not have access to the source code behind evidence in the case, the prosecution cannot be considered to be suppressing such evidence.

The 2008 decision *City of Fargo v. Levine* before the North Dakota Supreme Court involved Glenn Levine, whom the police had arrested for driving under the influence based in part on the results of an Intoxilyzer.[192] Levine retained Dr. Robert Howard as an expert because he was "chief operating officer of Medscan Laboratory and Advanced

---

defendant's assertion that failure to provide breathalyzer source code violates the Confrontation Clause).

[186] *See Lindner*, 252 P.3d at 1034, n.1 (stating that defendant also raised due process right to discover source code); *Brady*, 373 U.S. at 87.

[187] *Brady*, 373 U.S. at 84.

[188] *Id.*

[189] *Id.*

[190] *Id.*

[191] *Id.* at 87.

[192] 747 N.W.2d 130, 131-32 (N.D. 2008).

Drug Testing, a company providing third-party drug and alcohol testing for employers."[193] In his work, Howard was familiar with the Intoxilyzer as well as having professional experience writing source code.[194] Howard explained that "the source code used in Intoxilyzer machines varies among jurisdictions and, when purchasing a machine, the buyer specifies certain types of programming."[195] He testified that without the specific Intoxilyzer's source code, he could not properly analyze Levine's results, but the trial court nonetheless denied the motion to compel production of the source code.[196]

The North Dakota Supreme Court explained that the prosecutor neither possessed nor controlled the Intoxilyzer source code.[197] Levine had furthermore failed to provide any evidence that that city possessed or controlled the source code in its Intoxilyzer.[198] The state supreme court concluded that the prosecutor did not commit a *Brady* violation by failing to produce source code that was not in the city's possession.[199]

In 2008, the Supreme Court of Montana also heard a challenge to a denial of a request for the source code for the Intoxilyzer 8000.[200] In *Peters*, CMI, the manufacturer of the Intoxilyzer 8000, raised concerns about trade secrets in Kentucky state court.[201] However, CMI agreed to provide the defense expert access to the source code in its Kentucky corporate headquarters subject to a protective order that required the defense expert to destroy his computer hard drive when he completed the review and left CMI headquarters.[202] Although the defense expert did not object to signing a protective order, he considered the CMI

---

[193] *Id.* at 132.
[194] *Id.*
[195] *Id.*
[196] *Id.*
[197] *Id.* at 133.
[198] *Id.*
[199] *Id.*
[200] State v. Peters, 264 P.3d 1124, 1127 (Mont. 2011).
[201] *Id.* at 1127; *see also* Wexler, *supra* note 11, at 1419 (citing Kewanee Oil Co. v. Bicron Corp., 416 U.S. 470, 490 (1974)) ("The U.S. Supreme Court has even recognized that the 'substantial risk' of undetectable theft of trade secrets may be socially useful in that it encourages innovators to seek patents, which require disclosure.").
[202] *Peters*, 264 P.3d at 1129.

agreements to be too restrictive and declined to sign them.[203] Specifically, the defense expert explained that he feared signing the CMI agreements because the company "'threaten[ed] my license to practice law if I violated'" the agreements.[204] He further explained that "'it's not that I won't sign it. *I don't believe I can sign it—without subjecting myself to sanctions*.'"[205] In other words, the criminal defendants were not able to present any defense to the charge against him because CMI essentially cowed the expert witness by threatening his livelihood and liberty.

In *Peters*, the Supreme Court of Montana explained that the prosecution did not have the Intoxilyzer 8000 source code.[206] It determined that because CMI did provide the defense expert reasonable access to the source code the district court did not abuse its discretion in limiting the information that the prosecution was required to provide.[207] Consequently, the Supreme Court of Montana declined to address claims based on either a *Brady* violation or the Confrontation Clause.[208] However, this approach ignored the legitimate concerns raised by the defense expert.

Meanwhile in 2012, in *State v. West*, the Court of Appeals of Oregon considered Donald West's appeal of his drunk driving conviction.[209] Defense counsel challenged the results of an Intoxilyzer 8000.[210] Oregon law establishes a presumption of admissibility of the breathalyzer results provided that they comply with methods approved by the Oregon Department of State Police.[211] Rules issued by Oregon's State Police explicitly authorize the use of the Intoxilyzer 8000.[212]

---

[203] *Id.*
[204] *Id.*
[205] *Id.* (emphasis added).
[206] *Id.* at 1131.
[207] *Id.* at 1132.
[208] *Id.*
[209] 279 P.3d 354, 356 (Or. Ct. App. 2012).
[210] *Id.*
[211] OR. REV. STAT. § 813.160(1) (2011); *see also West*, 279 P.3d at 356.
[212] OR. ADMIN. R. 257-030-0120 (2006); *see also West*, 279 P.3d at 357.

In *West*, the appellate court explained that the prosecution had met its *Brady* obligations because it provided all information that it had regarding the Intoxilyzer 8000.[213] Defense counsel had furthermore failed to demonstrate that the source code was material, but instead simply posited that it "'could be used to show the manufacturer's bias toward producing evidence that will lead to convictions.'"[214] Consequently, the appellate court determined that "*Brady* is not authority for a defendant obtaining evidence of unknown import to test whether it helps or hurts his case."[215]

In 2013, in *State v. Marino*, the Court of Appeals of North Carolina heard an appeal challenging a jury verdict finding Jory Marino guilty of driving while intoxicated.[216] Defense counsel filed a motion seeking *Brady* material, which the trial judge essentially granted ordering that the prosecution provide defense counsel "with 'all downloaded and non-downloaded data in its possession that was generated from [the] Intoximeter [used to analyze defendant's breath.]'"[217] But the trial judge did not order the production of the Intoximeter's source code.[218] This meant that the defense was not in a position to determine if the Intoximeter breath analysis software worked correctly.[219]

Pursuant to *Brady*, a "prosecutor is not required to deliver his entire file to defense counsel, but only to disclose evidence favorable to the accused that, if suppressed, would deprive the defendant of a fair trial."[220] Missing from the decision is any clarification of who decides what suppressed information might deprive the defendant of a fair trial. How do they make that determination?

In *Marino*, the appellate court determined that defense counsel had not established that the Intoximeter source code would be favorable or material to his defense regarding either the guilt phase or the punishment

---

[213] *West*, 279 P.3d at 359.
[214] *Id.*
[215] *Id.*
[216] 747 S.E.2d 633, 634 (N.C. Ct. App. 2013).
[217] *Id.* at 635.
[218] *Id.*
[219] *See id.*
[220] United States v. Bagley, 473 U.S. 667, 675 (1985).

phase of his prosecution.[221] The court moreover characterized defense counsel's interest in this source code and the possibility that it could result in exculpatory evidence to be speculative at best.[222] The court did not acknowledge—and perhaps did not know—that software often lies, not out of programmers' malfeasance, but as a result of design or coding errors. In *Marino*, the appellate court noted that *Brady* evidence must be favorable or material to guilt or punishment.[223] However, it ignored the significant factor that the prosecution must actually possess evidence in order to be able to determine whether the evidence is favorable.[224] Without the code to examine, the defense is not a position to make this determination either. The court is effectively setting up a Catch-22 proposition.

The fact that prosecutors often do not have any copies of the breathalyzer source code is a significant issue regarding *Brady* concerns. In Minnesota, a trial judge ordered the prosecution to produce to the defense attorney "the complete computer source code for the operation of the Minnesota model of the Intoxilyzer 5000 currently in use."[225] Specifically, the trial judge determined that Minnesota "owned the source code for the Intoxilyzer 5000EN model created exclusively for the state."[226]

In response to this order, the Commissioner filed a writ of prohibition with the appellate court seeking to bar the trial judge from enforcing this order.[227] The fact that Minnesota owned part of the source code pursuant to its contract with CMI was significant for the Minnesota Supreme Court's decision to uphold the appellate court.[228] Specifically, the state supreme court rejected the Commissioner's argument that he did not possess or control the source code based on this contractual language.[229] This decision did not concern *Brady* materials because the court

---

[221] *Marino*, 747 S.E.2d at 638.
[222] *Id.*
[223] *Id.*
[224] *See Brady*, 373 U.S. at 87.
[225] *In re* Comm'r of Pub. Safety, 735 N.W.2d 706, 709 (Minn. 2007).
[226] *Id.*
[227] *Id.*
[228] *Id.* at 712.
[229] *Id.*

determined that Minnesota had the source code and should produce it consistent with the trial judge's order. It stands as a contrast to the situations where prosecutors do not have access to the source code.

Defendants have generally failed in obtaining the source code for breathalyzers based on arguments pursuant to *Brady*. For example, they often cannot demonstrate the need for the source code. It is, of course, extremely difficult to prove that the source code has errors if the code is not available for examination. The other wrinkle is that often the prosecution can establish that it does not have the source code either. Consequently, a court cannot very well compel the prosecution to produce something that it does not possess.[230]

B. Confrontation Clause

In addition to *Brady* challenges, criminal defendants may argue that the failure to provide the source code for breathalyzers also violates the Confrontation Clause. Specifically, the Sixth Amendment of the Constitution provides that "[i]n all criminal prosecutions, the accused shall enjoy the right to . . . to be confronted with the witnesses against him."[231]

One might think that this is a straightforward rule, but in recent years it has become anything but. The confusing situation had its genesis in a relatively uncomplicated murder case, *Crawford v. Washington* that, due to issues of admissibility of evidence from the defendant's wife, made its way to the Supreme Court.[232] The evidence in that decision concerned testimony from a living person, but the implications of

---

[230] One problem is that CMI may not have the source code for a given Intoxilyzer. *See* Affidavit of Brian Faulkner at ¶ 5, State v. Bonakoske (Fla. Charlotte County Ct. June 10, 2011) (No. 08-1726-T) (In response to requests for source code, CMI's Manager of Engineering swore that "CMI is unable to produce, with any degree of certainty, each and every section of source code from software revisions that were not put into service by the customer. Consequently, CMI cannot reliably reconstruct version 8100.10 which directly corresponds to software version 8100.10 included in the instrument initially tested, but not utilized, by the Florida Department of Law Enforcement."). Of course, if the source code is non-existent for CMI as well, it begs the question of how it can reliability be assessed at all.

[231] U.S. CONST. amend. VI.

[232] 541 U.S. 36, 36 (2004).

Crawford stretch all the way to software. We discuss the Confrontation Clauses in two parts: (i) how the courts have ruled about expert witness testimony post *Crawford* and (ii) the implications of *Crawford* on evidentiary software.

1. Expert Witness Testimony post-*Crawford*

Interpreting the Confrontation Clause, the United States Supreme Court determined that the clause afforded criminal defendants the right to cross-examine any witnesses who "bear testimony" against the defendant.[233] In *Crawford*, the defendant, Michael Crawford, confronted a man he believed had attempted to rape his wife, Sylvia Crawford.[234] Believing the alleged rapist was reaching for a weapon, Michael Crawford stabbed the man.[235] Sylvia Crawford's statement to the police did not corroborate her husband's claim of self-defense, and Michael Crawford was charged with attempted murder.[236]

The prosecution sought to have Ms. Crawford testify, but Washington's spousal privilege law prevented her from testifying against her husband without his consent.[237] The prosecution was able to introduce the wife's statement because it was admissible pursuant to a hearsay exception.[238] The defendant argued that the statement's admission violated his Confrontation Clause rights.[239] The trial court admitted the statement, finding that it had "adequate 'indicia of reliability.'"[240] After the jury found Crawford guilty, the state appellate court reversed finding that the statement was inadmissible before the Washington Supreme Court reinstated his conviction.[241]

---

[233] *Id.* at 51.
[234] *Id.* at 38.
[235] *Id.* at 38-39.
[236] *Id.* at 39-40.
[237] *Id.* (discussing WASH. REV. CODE § 5.60.060(1) (1994)).
[238] *Id.* (citing State v. Burden, 841 P.2d 758, 761 (Wash. 1992)).
[239] *Id.*
[240] *Id.*
[241] *Id.* at 41.

The United States Supreme Court grappled with whether the admission of Ms. Crawford's out-of-court statement violated the Confrontation Clause because the defendant could not cross-examine the statement.[242] In a unanimous decision, the Court determined that its admission violated Crawford's Sixth Amendment rights.[243] Justice Scalia wrote that the Framers of the Constitution intended the Confrontation Clause to bar these types of out-of-court statements.[244] In finding Ms. Crawford's statement inadmissible, the Court overruled *Ohio v. Roberts*,[245] and determined that analysis of the statement's reliability was inadequate in preserving the Sixth Amendment rights and that courts must address the evidence's testimonial nature.[246]

Evidence deemed to be testimonial may not be introduced against a criminal defendant unless the witness is unavailable *and* the defendant had a previous opportunity to cross-examine the witness regarding the substance of the proposed evidentiary statement.[247] Legal scholar Jennifer Mnookin explained that in *Crawford* a critical issue concerned *whether* "the statement [was] made in circumstances that suggest[s] its likely future relevance as testimony in a criminal [trial]?"[248] Thus, business records, Call Detail Records, web searches, and the like would not be covered by *Crawford* decision's analytical reasoning[249]—but breathalyzer tests and DNA tests are.

In *Crawford*, the Court dealt with testimonial evidence in the case of a human witness whose testimony the prosecution introduced during the trial, but who was not available for cross examination by the defendant during trial.[250] Our interest is in (particular types of) expert-witness testimony, and so we turn to a series of post-*Crawford* Supreme Court

---

[242] *Id.* at 38.
[243] *Id.* at 68-69.
[244] *Id.* at 43-50.
[245] 448 U.S. 56 (1980).
[246] *Id.* at 67-68.
[247] *Id.* at 53-54.
[248] Jennifer Mnookin, *Expert Evidence and the Confrontation Clause After Crawford v. Washington*, 15 J.L. & POL'Y 794 (2007).
[249] *Id.*
[250] *Crawford*, 541 U.S. at 38.

rulings on the admissibility of testimony—or not—by expert-witnesses in which the Court subsequently ruled.

In *Melendez-Diaz v. Massachusetts*, the Supreme Court heard arguments about a prosecution involving an alleged drug dealer.[251] Specifically, the state presented certificates from an in-state laboratory that tested whether the numerous plastic bags containing a white powder the defendant allegedly secreted within a police car as he was being transported to the station were cocaine.[252]

The state laboratory tested all the bags from the scene as well as those located in the police cruiser.[253] At trial, the prosecution introduced the bags of cocaine into evidence along with three certificates of analysis establishing that the bags tested positive for cocaine.[254] Specifically, "[t]he certificates reported the weight of the seized bags and stated that the bags '[h]a[ve] been examined with the following results: The substance was found to contain: Cocaine.'"[255] Pursuant to Massachusetts law, the lab analysts swore before notaries public to the findings in the certificates.[256]

Instead of the laboratory technicians who analyzed whether the packages contained cocaine and the amount of narcotics, the prosecution simply presented the certificates.[257] These certificates were, as required by state law, prepared by the State Laboratory of the Massachusetts Department of Health.[258] The lab technician who prepared the certificates did not appear in court, but the certificates were sworn to by analysts from the lab in front of a notary public.[259] Relying on the requirements put forth in *Crawford,* the defendant, Luiz

---

[251] 557 U.S. 305, 307-08 (2009).
[252] *Id.* at 308. The amount of cocaine was significant, as that would determine the charge. *Id.*
[253] *Id.*
[254] *Id.*
[255] *Id.*
[256] *Id.* (citing MASS. GEN. LAWS ANN. ch. 111, § 12 (West 2006)).
[257] *Id.* at 308-09.
[258] *Id.* at 308.
[259] *Id.*

Melendez-Diaz, objected, claiming that the documents were testimonial in nature.[260] The Court concurred.[261]

Justice Scalia, writing for a majority that included Justices Stevens, Souter, Thomas, and Ginsburg, concluded that Melendez-Diaz's Confrontation Clause right was violated when he was barred from confronting these technicians.[262] Addressing the certificates of analysis, the Court explained that as they were affidavits, "[t]here is little doubt that the documents at issue in this case fall within the 'core class of testimonial statements.'"[263] To the state's argument that the analysts are not subject to confrontation, the Court responded that "[t]he Sixth Amendment guarantees a defendant 'the right to be confronted with the witnesses *against him*.'"[264] Specifically, the Court explained that "Confrontation is one means of assuring accurate forensic analysis . . . designed to weed out not only the fraudulent analysis, but the incompetent as well."[265] The Court cautioned that "[s]erious deficiencies have been found in the forensic evidence used in criminal trials."[266]

One such situation in Massachusetts became public a few years later. A chemist employed in a Massachusetts state drug testing lab, Annie Dookhan, had been falsifying drug reports in an attempt to build an impressive number of tests she was analyzing.[267] In many cases, she failed to run analyses that she reported to the courts.[268] Dookhan pleaded guilty to "27 counts of misleading investigators, tampering with

---

[260] *Id.* at 309.
[261] *Id.* at 329.
[262] *Id.*
[263] *Id.* at 310 (Thomas, J., concurring in part and concurring in the judgment) ("[T]he Confrontation Clause is implicated by extrajudicial statements only insofar as they are contained in formalized testimonial materials, such as affidavits, depositions, prior testimony, or confessions" (quoting *White v. Illinois,* 502 U.S. 346, 365 (1992))).
[264] *Id.* at 313.
[265] *Id.* at 318-19.
[266] *Id.* at 319.
[267] Katie Mettler, *How a Lab Chemist Went from "Superwoman" to Disgraced Saboteur of More Than 20,000 Cases,* WASH. POST (Apr. 21, 2017), https://www.washingtonpost.com/news/morning-mix/wp/2017/04/21/how-a-lab-chemist-went-from-superwoman-to-disgraced-saboteur-of-more-than-20000-drug-cases/?noredirect=on [https://perma.cc/4YV4-TV6Z].
[268] *Id.*

evidence, and filing false reports." [269] Ultimately, prosecutors in eight Massachusetts counties dismissed charges in over 21,000 drug cases as a result.[270]

After *Melendez-Diaz*, the Supreme Court ruled on another case in which lab certificates were introduced, but the technician who prepared it did not appear in court: *Bullcoming v. New Mexico*.[271] The police arrested Donald Bullcoming for DWI after he caused an accident.[272] After he refused to take a breathalyzer, the police obtained a search warrant to obtain a blood sample at a local hospital.[273] However, the blood was analyzed at a state laboratory.[274] At Bullcoming's trial, the prosecution introduced the blood alcohol test results, but the technician who conducted the test was unavailable to testify.[275] Instead, another lab technician who was familiar with the test equipment and procedures testified.[276] The trial court determined that although the analysis of blood-alcohol levels was testimonial, the certifying analyst did not need to appear in court; in-court testimony from another analyst from the same lab had sufficed to meet the requirements of the Confrontation Clause.[277]

Relying on *Melendez-Diaz*, the Supreme Court of New Mexico had already ruled that the lab analysis was testimonial. Despite that determination, the court concluded that the lab report was nonetheless "functionally identical to live, in-court testimony, doing precisely what a witness does on cross-examination."[278] Making clear that testimonial is testimonial, the United States Supreme Court was having none of this argument: "if an out-of-court statement is testimonial, it may not be introduced against the accused at trial unless the witness who made the

---

[269] *Id.*
[270] *Id.*
[271] 564 U.S. 647, 651 (2011).
[272] *Id.* at 652.
[273] *Id.*
[274] *Id.* at 652-53.
[275] *Id.* at 655.
[276] *Id.*
[277] *Id.* at 655-56.
[278] State v. Bullcoming, 226 P.3d 1, 8 (N.M. 2008) (quoting Melendez-Diaz v. Massachusetts, 557 U.S. 305, 308 (2009)).

statement is unavailable and the accused has had a prior opportunity to confront the witness."[279] In an observation that is important in considering evidentiary software, the majority opinion noted that, "[i]n any event, the comparative reliability of an analyst's testimonial report does not overcome the Sixth Amendment bar. This Court settled in Crawford that the 'obviou[s] reliab[ility]' of a testimonial statement does not dispense with the Confrontation Clause."[280]

At its core, the Confrontation Clause safeguards defendants against testimonial hearsay statements.[281] The Supreme Court has explained that "[t]estimonial statements of witnesses absent from trial have been admitted only where the declarant is unavailable, and only where the defendant has had a prior opportunity to cross-examine."[282] Nonetheless, the Court has hesitated in defining all types of testimonial statements.[283] Indeed, in assessing whether a statement is testimonial, the Supreme Court has explained that "courts should look to all of the relevant circumstances."[284] Moreover, in analyzing *Crawford*, the Court noted that the decision "did not offer an exhaustive definition of 'testimonial' statements. Instead, *Crawford* stated that the label 'applied at a minimum to prior testimony at a preliminary hearing, before a grand jury, or at a former trial; and to police interrogations.'"[285] Such opaque standards hardly give courts much guidance.

Legal scholars Jennifer Mnookin and David Kaye have noted that:

> [f]or the most part, [lower courts] were reluctant to read *Crawford* as requiring any significant changes to the preexisting methods for introducing forensic science testimony. In the first years following the *Crawford* decision, most (though certainly not all) trial courts instead endeavored to shoehorn the traditional methods

---

[279] *Bullcoming*, 564 U.S. at 657.
[280] *Id.* at 661 (quoting Crawford v. Washington, 541 U.S. 36, 61 (2004)).
[281] *See* U.S. CONST. amend. VI.
[282] *Crawford*, 541 U.S. at 59.
[283] *See* Davis v. Washington, 547 U.S. 813, 822 (2006).
[284] Michigan v. Bryant, 562 U.S. 344, 369 (2011).
[285] *Crawford*, 541 U.S. at 68.

> for presenting forensic science testimony into this new framework without requiring modifications.[286]

For example, in the *Levine* case, the defendant retained a highly qualified expert who sought the source code, but the court had concluded that, as the government did not have the source code, the prosecutor had not committed a *Brady* violation by failing to provide the code to the defendant.[287] This reasoning was faulty. At issue was not a Brady violation—providing potentially exculpatory evidence in possession of the government to the defendant—but rather a failure to satisfy the requirements of the Confrontation Clause.

The North Dakota Supreme Court determined that its state law allows criminal defendants to cross-examine either the director of the state crime laboratory or one of its employees.[288] However, this decision preceded *Melendez-Diaz* by about a year, and the state supreme court rejected the Confrontation Clause argument because the defendant chose not to call any such witness, but instead pled guilty.[289] In other words, the defendant waived his claim to a Sixth Amendment violation.

In Arizona, state law prevents criminal defendants from challenging the admissibility of breathalyzer results. Specifically, the legislature mandated that "[t]he inability of any person to obtain manufacturer's schematics and software for a quantitative breath testing device that is approved . . . shall not affect the admissibility of the results of a breath test."[290] Thus, when Michael Lindner asserted that the failure to require the prosecution to provide the source code and a witness to cross-examine violated the Confrontation Clause, the Arizona Court of Appeals rejected Lindner's arguments, distinguishing *Melendez-Diaz* as

---

[286] Jennifer Mnookin & David Kaye, *Confronting Science: Expert Evidence and the Confrontation Clause*, 2012 SUP. CT. REV. 99, 105.
[287] City of Fargo v. Levine, 747 N.W.2d 130, 132-33 (N.D. 2008).
[288] *Id.* at 135 (first discussing N.D. CENT. CODE § 39-20-07(5) (2019); and then discussing Berger v. State Highway Comm'r, 394 N.W.2d 678, 686 (N.D. 1986)).
[289] *Id.*
[290] ARIZ. REV. STAT. ANN. § 28-1323(A)(5) (2006).

only requiring the technician who performed the blood tests as opposed to the designer of the equipment used by that technician.[291]

In *Lindner*, the appellate court determined that because the officer who conducted the breath tests appeared at trial it was not necessary to require the breathalyzer's designer to appear also in order to discuss the source code.[292] The court concluded that a constitutional violation did not occur.[293]

In *West*, yet another DWI prosecution, the defendant sought the source code for the Intoxilyzer 8000. The 2012 decision by the Oregon Court of Appeals rejected this argument, finding that there was "an opportunity to cross-examine witnesses and to impeach the state's evidence, including the Intoxilyzer test results."[294] The appellate court reasoned that the admission of the Intoxilyzer's certificates of accuracy prepared by technicians, but without their testimony, did not violate the Confrontation Clause because these certificates were not testimonial in nature.[295]

In *Marino,* a 2012 North Carolina case, the defendant asserted he needed the source code in order to cross-examine "his primary accuser, the Intoximeter."[296] Looking to *Melendez-Diaz*, the North Carolina Court of Appeals explained that the Supreme Court's decision provided defendants with the right to "cross-examine those individuals involved in the production of testimonial documents to be introduced at trial, *such as the technician operating the Intoximeter in the present case*," but held that *Melendez-Diaz* did not provide a basis for the defendants to examine the breathalyzer source code.[297] This reasoning in *West* flies in the face of the *Melendez-Diaz*. This is especially so in light of the fact

---

[291] State v. Lindner, 252 P.3d 1033, 1035-36 (Ariz. Ct. App. 2010) (citing Melendez-Diaz v. Massachusetts, 557 U.S. 305, 310-11 (2009)).
[292] *Id.* at 1036.
[293] *Id.*
[294] State v. West, 279 P.3d 354, 361 (Or. Ct. App. 2012).
[295] *Id.*
[296] State v. Marino, 747 S.E.2d 633, 638 (N.C. Ct. App. 2013).
[297] *Id.* at 639 (emphasis added).

that not all state technicians can be trusted—as the conviction of Annie Dookhan in Massachusetts clearly establishes.[298]

The situation was different in Minnesota; there, a state court determined that source code is essential to safeguarding a defendant's rights. In *State v. Underdahl*, the Minnesota Supreme Court determined that defendants who failed to provide any information or documentary evidence that the breathalyzer source code was relevant to their criminal defense would be denied such source code.[299] However, the Minnesota Supreme Court further concluded that a defendant who provided evidence that "an analysis of the source code may reveal deficiencies that could challenge the reliability of the Intoxilyzer" should be entitled to engage in discovery regarding that source code.[300]

Interpreting *Underdahl*, the Court of Appeals of Minnesota addressed a defendant who challenged the revocation of his driver's license based on an implied consent statute and an alcohol concentration in excess of the legal limit.[301] James Lund submitted an affidavit from a forensic scientist who explained the significance of access to the source code for the defendant.[302] Moreover, the forensic scientist opined that an analysis of such code is necessary to determine whether they functioned

---

[298] The Dookhan scandal is not the only one to rock the criminal justice system leading to the overturning of numerous convictions. *See generally* Keith L. Alexander, *Prosecutors Criticize D.C. Crime Lab's Handling of Some DNA Evidence*, WASH. POST (Mar. 5, 2015), https://www.washingtonpost.com/local/crime/dc-prosecutors-criticize-city-crime-labs-handling-of-some-dna-cases/2015/03/05/b5244f88-bea4-11e4-b274-e5209a3bc9a9_story.html [https://perma.cc/77Q5-9AGW]; Kyle Swenson, *Broward Crime Lab Scandal Could Taint Many Cases*, MIA. NEW TIMES (Aug. 7, 2014), https://www.miaminewtimes.com/news/broward-crime-lab-scandal-could-taint-many-cases-6396401 [https://perma.cc/CS7U-BJA4]; Mark Hansen, *Crime Labs Under the Microscope After a String of Shoddy, Suspect and Fraudulent Results*, ABA JOURNAL (Sept. 1, 2013), http://www.abajournal.com/magazine/article/crime_labs_under_the_microscope_after_a_string_of_shoddy_suspect_and_fraudu [https://perma.cc/RG6M-36LX]; Maurice Chammah, *After Drug Lab Scandal, Court Continues to Reverse Conviction*, TEX. TRIBUNE (Mar. 27, 2013), https://www.texastribune.org/2013/03/27/after-drug-lab-scandal-court-reverses-convictions/ [https://perma.cc/R27P-3NHN] (reporting that "a former employee with a DPS crime laboratory in Houston may have fabricated the results of thousands of drug tests").

[299] 767 N.W.2d 677, 685 (Minn. 2009).

[300] *Id.* at 685-86.

[301] Lund v. Comm'r of Pub. Safety, No. A08-1408, 2009 WL 1587135, at *1 (Minn. Ct. App. June 9, 2009).

[302] *Id.*

accurately.[303] Notwithstanding this expert affidavit, the trial court denied the request for discovery of the source code.[304] The *Lund* appellate court reversed and remanded, finding that discovery was appropriate based on *Underdahl* because "a showing that discovery of the source code is relevant to the accuracy of a petitioner's test results is sufficient to support a motion for discovery of the source code."[305]

The Court of Appeals of Minnesota heard similar challenges seeking the breathalyzer source code in criminal prosecutions for driving while intoxicated. For example, the appellate court reversed and remanded the trial court's decision to deny discovery of the source code when the defendant submitted a declaration from a computer forensics expert[306] as well as a complaint filed by Minnesota against the manufacturer of the Intoxilyzer.[307] Similarly, in *State v. Veldhuizen*, the Court of Appeals of Minnesota concluded that the trial court abused its discretion in denying a request for discovery of the source code where defendant submitted an expert affidavit describing the source code and its effect of test results as well as information regarding "two inconsistent data reports produced from the same underlying intoxilyzer test."[308]

Yet the Court of Appeals of Minnesota affirmed criminal convictions for driving while intoxicated when the defendants failed to demonstrate before the trial court how the source code would assist them in challenging the charge against them: "Cornish did not present a proffer or analysis in support of either his pretrial motion to compel or his timely motion for a new trial that is comparable to the submission found sufficient in *Underdahl II*."[309] The Cornish court determined that his

---

[303] *Id.*
[304] *Id.*
[305] *Id.* at *3; *see also* Duncan v. Comm'r of Pub. Safety, No. A08-2237, 2009 WL 2366280, at *4 (Minn. Ct. App. Aug. 4, 2009) (holding that the trial court abused its discretion in denying discovery of the source code).
[306] State v. Granse, No. A09-2192, 2010 WL 4451243, at *2 (Minn. Ct. App. Nov. 9, 2010).
[307] Minnesota *ex rel.* Campion v. CMI of Kentucky, Inc., No. 08-603, 2009 WL 2170134, at *4 (D. Minn. July 16, 2009).
[308] Nos. A08-2110, A08-2112 & A08-2113, 2009 WL 1684494, at *2 (Minn. Ct. App. Nov. 9, 2010).
[309] State v. Cornish, No. A08-1228, 2010 WL 2035610, at *4 (Minn. Ct. App. May 25, 2010).

proffers were insufficient to establish that the source code would be useful in challenging the breathalyzer evidence.[310]

Similarly, in *State v. Garberg*, the Court of Appeals of Minnesota heard another appeal challenging the denial of access to a breathalyzer source code.[311] The court determined that the defendant "failed to make a threshold evidentiary showing that the source code information may relate to his guilt or innocence, negate his guilt, or reduce his culpability" before the trial court as mandated in *Underdahl*.[312] This inconsistency is troubling. As we have shown, the problem of buggy (and hence unreliable) code is endemic in computer software; one cannot assume, absent substantial evidence, that some particular piece of software is correct.

The issue of correct software occurs in a myriad of different situations, including in situations of far greater import than an individual DWI case. For example, recently, the federal government engaged in a nationwide prosecution of individuals who distributed child pornography images amongst themselves on a secured website known as Playpen.[313] Criminal defendants around the country had varying degrees of success in challenging the evidence used against them.[314]

In 2016, in *United States v. Michaud*, the federal government indicted Jay Michaud for offenses related to child pornography.[315] Defense counsel moved for the prosecution to produce the source code for the network investigative technique ("NIT") used by the FBI during its investigation.[316] The technique was designed to activate a "computer—wherever located—to send to a computer controlled by or known to the government, network level messages containing information that may assist in identifying the computer, its location, [and] other

---

[310] *Id.*
[311] No. A09-914, 2010 WL 772622, at *3 (Minn. Ct. App. Mar. 9, 2010).
[312] *Id.*
[313] *See generally* Brian L. Owsley, *Network Investigative Source Code and Due Process*, 14 DIGIT. EVID. & ELEC. SURVEILLANCE L. REV. 39 (2017).
[314] *See id.*
[315] No. 3:15-cr-05351, 2016 WL 337263, at *1-2 (W.D. Wash. Jan. 28, 2016).
[316] *Id.*; Order on Procedural History and Case Status in Advance of May 25, 2016 Hearing at 1, *Michaud*, 2016 WL 337263 (No. 3:15-cr-05351).

information."[317] The district court concluded that it would be prejudicial to the defendant if the government were allowed to enter evidence to which the defendant could not meaningfully respond because of lack of access to the NIT source code.[318] The court further explained given that "the discovery withheld implicates the defendant's constitutional rights,"[319] any "evidence of the NIT and the search warrant issued on the basis of the NIT should be suppressed, and the fruits of that search must also be suppressed."[320]

The prosecution then moved to dismiss the charges against Michaud without prejudice,[321] resulting in the dismissal of the charges against him.[322] However, Michaud was an exception; most defendants caught through the Playpen investigation failed to gain access to the source code despite filing motions to do so.[323]

Most recently, in a prosecution reliant upon DNA software for identification, the New York Supreme Court Appellate Division considered whether the Confrontation Clause required the prosecution to provide the defendant with source code.[324] Law enforcement had located DNA on the cord used to strangle the decedent.[325] After the defendant's arrest, the police took a buccal swab to gather his DNA for testing using a software program known as TrueAllele Casework System.[326] The defendant asserted that the state violated his

---

[317] *Michaud*, 2016 WL 337263, at *2.
[318] Owsley, *supra* note 313, at 45; *see also* Order Denying Dismissal and Excluding Evidence at 1, *Michaud*, 2016 WL 337263 (No. 3:15-cr-05351).
[319] *Id.*
[320] *Id.*
[321] Government's Unopposed Motion to Dismiss Indictment Without Prejudice at 3, *Michaud*, 2016 WL 337263 (No. 3:15-cr-05351).
[322] Order Dismissing the Indictment Without Prejudice at 1, *Michaud*, 2016 WL 337263 (No. 3:15-cr-05351).
[323] *See, e.g.*, United States v. Harney, No. 16-38, 2018 WL 1145957, at *12 (E.D. Ky. Mar. 1, 2018); United States v. Stepus, No. 3:15-cr-30028, 2018 WL 1257804, at *4 (D. Mass. Mar. 12, 2018); United States v. Zak, No. 16-CR-65, 2017 WL 4358140, at *4 (W.D.N.Y. Oct. 2, 2017); United States v. Spicer, No. 1:15-cr-73, 2018 WL 635889, at *5 (S.D. Ohio Jan. 31, 2018).
[324] *See* People v. Wakefield, 107 N.Y.S.3d 487, 495 (App. Div. 2019).
[325] *Id.* at 491.
[326] *Id.*

Confrontation Clause right by denying him access to TrueAllele's source code.[327]

The appellate court characterized the argument by the defendant, Wakefield, as novel because it was based on the source code being the out-of-court declarant.[328] Moreover, it determined that "the TrueAllele report [was] testimonial in nature."[329] Nonetheless, the court rejected the argument that the source code should be available to the defendant because the creator of TrueAllele who wrote its source code had testified about the pertinent algorithm.[330] In the court's opinion, there was no violation of Wakefield's Confrontation Clause right.[331] Having the algorithm's designer testify about the code did not answer, however, whether the software performed correctly; only an examination of the code could do that.

The decision in *Wakefield*, along with *Michaud*, *Underdahl*, and its progeny, demonstrates that on a case-by-case basis, some courts are starting to acknowledge that source code may be essential for defendants to exercise their Confrontation Clause rights. These cases, however, have largely not addressed the issue of correctness of evidentiary software.

2. Implications for Evidentiary Software

The limited Confrontation Clause jurisprudence in this area has developed based on the determination that criminal defendants do not really need access to the source code—and thus their constitutional rights are not violated when they lack this ability. As Jennifer Mnookin explained:

> [A] great many lower court opinions have wrestled with the potential Confrontation Clause implications of expert evidence that includes statements that might be classified

---

[327] *Id.* at 494.
[328] *Id.* at 495.
[329] *Id.* at 497.
[330] *Id.*
[331] *Id.*

> as testimonial. Most of these courts have endeavored to find ways around *Crawford's* dictates; unfortunately, most of the arguments proffered by these courts are deeply intellectually unsatisfying.[332]

The previously discussed Intoxilyzer decisions further demonstrate that point.[333]

Since *Melendez-Diaz* and *Bullcoming*, courts have wrestled with appropriately handling the Confrontation Clause when the expert witness is not a person but rather a result determined by computer software. As Professor Mnookin indicated, the key issue in *Crawford* is, "[w]as the statement made in circumstances that suggest its likely future relevance as testimony in a criminal prosecution?"[334] For evidentiary software, the answer is an unqualified yes. Its purpose is exactly to produce testimony to be relied upon in a criminal trial. The expert is not the policeman proffering the results of a breathalyzer test or a radar gun; the expert is the reasoning exemplified in the software that processes the data—and the implementation of that reasoning. In this regard, *Wakefield* reached exactly wrong conclusion.

To get it right, we need to go back to *Daubert*, which lays out very clearly how to determine the admissibility of expert witness testimony:

> The Rules—especially Rule 702—place appropriate limits on the admissibility of purportedly scientific evidence by assigning to the trial judge the task of ensuring that an expert's testimony both rests on a reliable foundation and is relevant to the task at hand . . .

---

[332] Mnookin, *supra* note 248, at 796.
[333] *See* City of Fargo v. Levine, 747 N.W.2d 130 (N.D. 2008); State v. Peters, 264 P.3d 1124 (Mont. 2011); Wexler, *supra* note 11; State v. West, 279 P.3d 354 (Or. Ct. App. 2012); *In re* Comm'r of Pub. Safety, 735 N.W.2d 706 (Minn. 2007); Affadivit of Brian Faulkner, State v. Bonakoske (Fla. Charlotte County Ct. June 10, 2011) (No. 08-1726-T); State v. Underdahl, 767 N.W.2d 677 (Minn. 2009); State v. Granse, No. A09-2192, 2010 WL 4451243 (Minn. Ct. App. Nov. 9, 2010); Minnesota *ex rel.* Campion v. CMI of Kentucky, Inc., No. 08-603, 2009 WL 2170134 (D. Minn. July 16, 2009); State v. Veldhuizen, Nos. A08-2110, A08-2112 & A08-2113, 2009 WL 1684494 (Minn. Ct. App. Nov. 9, 2010).
[334] Mnookin, *supra* note 248, at 794.

> [T]he trial judge, pursuant to Rule 104(a), must make a preliminary assessment of whether the testimony's underlying reasoning or methodology is scientifically valid . . . .[335]

The Supreme Court's analysis in *Daubert* warrants repeating: the trial judge is assigned the task of ensuring that an *expert's testimony rests on a reliable foundation* and must make a preliminary assessment of *whether the testimony's underlying reasoning is scientifically valid*.[336] As we already have documented, despite the best intentions of programmers, source code often has errors. Yet there is no way for a defendant to demonstrate the existence of any errors without having the code examined. Therefore, any approach that eschews access to the code for such an examination presents a serious gap in the protection of defendants' rights.

A trial judge is not in a position to determine whether the underlying reasoning provided by the program is valid or whether, if the underlying reasoning is correct, it has been properly implemented. Indeed, when it comes to technological developments that impact a criminal defendant's constitutional rights, many judges are ill-equipped to analyze the nuances related to such technology.[337] This realization begs the question of how best to address this issue and safeguard constitutional rights.

We do not live in a perfect world where there are no criminals, where the criminals that exist conduct only minor criminal activities that cause no serious, long-term harm, where all computer code works perfectly the first time, or where software errors are easy to find. Consequently, we do not have a solution that can completely solve the problem of

---

[335] Daubert v. Merrell Dow Pharms., Inc., 509 U.S. 579, 579-80 (1993).
[336] *Id.*
[337] *See, e.g.*, Brian L. Owsley, *Triggerfish, Stingrays, and Fourth Amendment Fishing Expeditions*, 66 HASTINGS L.J. 183, 210 (2014) ("[O]ther magistrate judges may have received applications using the pen register application and not realized that they were authorizing or denying use of a cell site simulator."); Brian L. Owsley, *The Fourth Amendment Implications of the Government's Use of Cell Tower Dumps*, 16 U. PA. J. CONST. L. 1, 18 (2013) ("[I]n my own informal survey of magistrate judges nationwide, many have informed me that they were unfamiliar with cell tower dumps.").

convictions based on faulty software. However, we do have a solution posed by evidentiary software that can level the playing field and enable defendants "to confront the witnesses against [them]."[338] It is a solution that lies already in the words of *Melendez-Diaz*, where the Court enunciated that "[c]onfrontation is designed to weed out not only the fraudulent analyst, but the incompetent one as well."[339]

In one regard, *Crawford's* purpose was rooting out hearsay ("even if the Sixth Amendment is not solely concerned with testimonial hearsay, that is its primary object, and interrogations by law enforcement officers fall squarely within that class"[340]) but in essence, the courts relying on source code that the defendant cannot cross examine is analogous to the courts relying on hearsay. In analyzing hearsay, Mnookin wrote that, post-*Crawford*, it is not acceptable to simply cross examine an expert witness if she herself has relied on hearsay; the defendant must be also be afforded the opportunity to cross examine those upon whom the expert relied in preparing her testimony.[341]

One might imagine an objection to this two-prong approach posited by Mnookin on the grounds that this process can lead to an almost infinite regression of required witnesses, each one suggesting that they have learned some small fact relevant to the case from the next, who must then be called to testify.[342] Mnookin and Kaye call into question that only the expert needs to be available and questioned as a witness by defense counsel:

> Take the testimony about the chemical makeup of a drug in *Melendez-Diaz*. Justice Scalia implies that the person who establishes the accuracy of the testing device is not required. But why not? Is that not a prerequisite to being able to be confident about the accuracy of the result? Arguably the person who interpreted the result is making the most central bottom-line judgment relevant to the

---

[338] *Levine*, 747 N.W.2d at 134.
[339] Melendez-Diaz v. Massachusetts, 557 U.S. 305, 319 (2009).
[340] Crawford v. Washington, 541 U.S. 36, 53 (2004).
[341] Mnookin, *supra* note 248, at 832-33.
[342] *Id.* at 833.

> defendant. Is this individual therefore the only one who needs to testify? What if the technician who prepared the sample somehow contaminated it? Should that technician be required to testify as well? None of these technicians is merely engaged in documenting a chain of custody. They are all taking steps that contribute to the final conclusion about the chemical composition of the substance at issue.[343]

However, the issue they raise (which they also answer) is not of concern here: the "witness" is only one deep. It is the software upon which the expert witness relied in order to provide her testimony.

The issue of hearsay is exactly the point here. The expert witness testifying about the results of a breathalyzer or DNA test is not testifying as to what the software does. Instead, that expert is testifying as to what the program designer and coder intended the software to do. Nor, for the same reason, is it adequate for the program designer or coder to testify—for they know only what they intended the software to do, not that the program does exactly that, no more, no less, in all circumstances. The harm that *Crawford* seeks to prevent—the inability of the defendant to confront a witness testifying against him—is denied when he, or his delegated authorities, cannot examine the code. Only by viewing the code can the defendant have the rights afforded to him by the Confrontation Clause. Given the prevalence of errors in software, defendants must have that right.

We propose a simple test (analogous by voting system software independence) for determining whether software-based systems that produce criminal evidence should be subject to adversarial analysis. If the relevant inputs to the system are retained, and all other information and algorithms required to reproduce the software-based evidence are available to opposing parties, then there is likely no compelling benefit to adversarial analysis of the software. In such cases, the correctness of the software's behavior can be confirmed or refuted simply by independently recalculating what the software was supposed to do and

---

[343] Mnookin & Kaye, *supra* note 286, at 152-53.

comparing the result with the evidence in question. If, on the other hand, such an independent recalculation cannot be done (either because some relevant inputs are not available or because the software relies on proprietary algorithms or other behavior), there is no way for an opposing party to determine whether the evidence is correct without expert adversarial analysis of the software. Such a test has the benefit of being relatively straightforward to apply by courts.

We must admit that our proposed solution is not quite as simple as those words might make it appear. Evidentiary code is typically developed by the private sector with the intent of profit. In other words, making the underlying code public is implausible. However, there are many ways for the software to be examined by the defense that do not require the code to accessible to the general public. The most obvious answer is a protective order, similar to those used in patent cases, be used in regard to source code. This approach was considered in *Peters*, but unfortunately for the criminal defendant the expert refused to sign the proposed protective order, resulting in an adverse decision for the criminal defendant.[344] Provisions of such orders commonly include requirements that analysis be done on computers not connected to the Internet, and that the analysts not have any means, *e.g.*, external disks, with which they can make copies of the software at issue.[345] Other mechanisms, including limited courtroom closures, are discussed by Wexler.[346]

C. Due Process Clause

The Fifth Amendment mandates that "[n]o person shall . . . be deprived of life, liberty, or property, without due process of law."[347] Similarly, the Fourteenth Amendment serves as a constraint on action by states: "nor shall any State deprive any person of life, liberty, or property,

---

[344] State v. Peters, 264 P.3d 1124, 1129, 1131 (Mont. 2011).
[345] Lydia Pallas Loren & Andy Johnson-Laird, *Computer Software-Related Litigation: Discovery and the Overly-Protective Order*, 6 FED. CTS. L. REV. 75, 99-100 (2012).
[346] Wexler, *supra* note 11, at 1409-10.
[347] U.S. CONST. amend. V.

without due process of law."[348] Due process is not easily defined, but at its core, the right of due process is about fundamental fairness.[349]

Courts have rejected criminal defendants' due process claims for source code when the government does not have the source code.[350] "Due process does not require the police to seek and find exculpatory evidence."[351] The increased move by society to outsourcing services might well lead to a situation in which law enforcement increasingly does not actually have the software that was used in the conviction.[352] Thus, any requirement that a defendant has the right to examine software might need to be addressed not only judicially, but also legislatively.

While courts might be not be willing to compel discovery of the source code in civil proceedings, such as driver's license revocation hearings, discovery might be warranted in criminal proceedings.[353] In *Crandall*, the court indicated that due process was not an issue because it was a summary civil proceeding, but "[d]iscovery of the source code might be permitted for [a] criminal trial."[354] In the civil context, legislative solutions may be the only way for individuals to obtain this necessary information. Nonetheless, courts have a role in determining whether due process was violated. In the end, there is a fundamental unfairness for the government to rely on a technology and prevent a defendant from challenging its reliability when addressing the adverse consequences. Ultimately, courts have a role in determining whether a due process violation has occurred.

---

[348] U.S. CONST. amend. XIV, § 1.
[349] *See, e.g.*, Lassiter v. Dep't of Soc. Servs., 452 U.S. 18, 24-25 (1981); *see also* State v. Davis, 898 N.E.2d 281, 287 (Ind. 2008).
[350] *See* Mahan v. Bunting, No. 1:13-CV-00165, 2014 WL 1154054, at *3 (N.D. Ohio Mar. 20, 2014).
[351] *Id.* (quoting Stadler v. Curtin, 682 F. Supp. 2d 807, 818 (E.D. Mich. 2010)).
[352] *See, e.g.*, City of Fargo v. Levine, 747 N.W.2d 130, 133-34 (N.D. 2008).
[353] *See In re* Crandall, No. 2008–951, 2008 WL 4615633, at *3 (N.Y. Sup. Ct. Aug. 25, 2008), *aff'd sub nom.* Crandall v. Brovetto, 891 N.Y.S.2d 761 (2010).
[354] *Id.*

Machines based on software are susceptible to potential errors that lead to testimony that can wrongly convict individuals.[355] Without allowing defendants to have access to the source code related to their cases, they will be unable to have experts who can analyze whether the source code operates as it is intended to do or whether there are flaws that might lead to the conviction of an innocent defendant. Indeed, in 2008, in *State v. Kummer*, the Court of Appeals of Minnesota explained that failing to provide a defendant the source code for the Intoxilyzer 5000EN might violate her due process rights.[356]

Notwithstanding the *Kummer* court's decision, courts generally do not address the merits of whether the denial of access to source code for criminal defendants violates due process. Some reasons make sense. For example, in 2010, another Minnesota appellate court declined to address a due process violation based on the use of a breathalyzer when the defendant was convicted for driving under the influence based on a urine sample as opposed to the use of a breathalyzer.[357] Indeed, appellate courts regularly decline to address the issue because defendants raised it for the first time on appeal.[358]

In *Peters*, criminal defendants also asserted a claim based on a due process violation regarding convictions for driving under the influence.[359] In 2011, the Supreme Court of Montana declined to address the due process claim.[360] The court determined that CMI had offered to make the source code available to the defendants, but the defense expert refused the offer because he did not agree with CMI's

---

[355] Andrea Roth, *Machine Testimony*, 126 YALE L.J. 1972, 1989-90 (2017).
[356] No. A08-0533, 2008 WL 4472610, at *1 (Minn. Ct. App. Oct. 7, 2008).
[357] State v. Sterling, 782 N.W.2d 579, 581 n.2 (Minn. Ct. App. 2010).
[358] *See* Christian v. Comm'r of Pub. Safety, No. A13-1921, 2014 WL 1758374, at *2 (Minn. Ct. App. May 5, 2014); State v. Garberg, No. A09-914, 2010 WL 772622, at *3 (Minn. Ct. App. Mar. 9, 2010); Lund v. Comm'r of Pub. Safety, No. A08-1408, 2009 WL 1587135, at *3 (Minn. Ct. App. June 9, 2009); *see also* State v. Lindner, No. 1 CA-CR 09-0583, 2010 WL 2103532 at *1 (Ariz Ct. App. May 25, 2010) (Lindner "waived these claims . . . by failing to argue them" in his appellate brief) (citations omitted), *superseded by* 252 P.3d 1033 (Ariz Ct. App. 2010).
[359] State v. Peters, 264 P.3d 1124, 1129, 1132 (Mont. 2011).
[360] *Id.* at 1132 ("Courts should avoid constitutional questions whenever possible." (citing Kulstad v. Maniaci, 244 P.3d 722 (Mont. 2010))).

terms.[361] Specifically, he expressed concerns that he could be sanctioned and lose his law license based on the overly strict approach by CMI in its take-it-or-leave-it agreement.[362] Moreover, CMI expected the expert to work exclusively at its headquarters and to destroy his computer hard drive when the review was completed.[363] These demands by CMI also served to create a much more burdensome and costly review process.

In that context, the *Peters* court found that it was not necessary to review the due process claims.[364] However, this determination ignored the fact that it left the criminal defendants without any means to present a defense to the charges against them.

In *West*, the criminal defendant was indigent and sought state funds so that he could retain an expert to analyze the breathalyzer's source code.[365] After the trial court denied his request for such funds, West argued that this denial violated his due process rights.[366] Specifically, he asserted that the prosecution failed to demonstrate that the breathalyzer was reliable and that an expert to analyze the source code would assist him in obtaining exculpatory evidence.[367] In 2012, the Court of Appeals of Oregon rejected the due process argument, concluding that it constituted a fishing expedition.[368] It noted that West "relies instead on the assertion that, *if* various governmental agencies produced everything defendant requested . . . *then* an expert *might* be able to demonstrate that the documents reveal something useful."[369] This approach did not establish a basis for finding a due process violation.[370]

---

[361] *Id.*
[362] *Id.* at 1129.
[363] *Id.*
[364] *Id.* at 1132.
[365] State v. West, 279 P.3d 354, 360 (Or. Ct. App. 2012).
[366] *Id.*
[367] *Id.*
[368] *Id.*
[369] *Id.*
[370] *Id.* at 361.

In *Collins v. State*, the defendant was charged with DUI per se and DUI less safe, but the trial court only sentenced her based on a conviction for DUI less safe.[371] On appeal, she argued that her due process rights were violated because she was not able to access the source code for the Intoxilyzer 5000 for trial.[372] In 2014, the Court of Appeals of Georgia concluded that Collins' due process claim was moot because she was only convicted of DUI less safe, which was unrelated to the source code.[373]

In *Commonwealth v. Camblin*, the defendant challenged the denial pursuant to state and federal due process rights.[374] In 2015, the Massachusetts Supreme Judicial Court, which is the state's highest court, addressed the trial court's decision denying a motion in limine seeking to bar the breath test machine because it was scientifically unreliable.[375] Due process mandates that evidence presented by the prosecution must be reliable.[376] The court reversed, finding that the defendant was addressing a new technology and thus should have be able to challenge its reliability before the trial court.[377]

---

[371] 760 S.E.2d 606, 608 (Ga. Ct. App. 2014); *see also* GA. CODE ANN. § 40-6-391(a)(5) (2014) (establishing six ways in which a driver may be driving while impaired in some manner.) The statute defines DUI per se as "driv[ing] or be[ing] in actual physical control of any moving vehicle while . . . [t]he person's alcohol concentration is 0.08 grams or more at any time within three hours after such driving or being in actual physical control from alcohol consumed before such driving or being in actual physical control ended." *Id.* In other words, this definition fits the standard blood alcohol concentration. The statute also defines DUI less safe as "driv[ing] or be[ing] in actual physical control of any moving vehicle while . . . [u]nder the influence of alcohol to the extent that it is less safe for the person to drive." *Id.* § 40-6-391(a)(1). In order to meet this standard, it is not necessary to provide evidence of blood alcohol concentration. Instead, the prosecution simply needs to establish an impairment due to alcohol based on circumstantial evidence. *See, e.g.*, Davis v. State, 687 S.E.2d 854, 858 (Ga. Ct. App. 2009).
[372] *Collins*, 760 S.E.2d at 608.
[373] *Id.* at 608-09.
[374] 31 N.E.3d 1102, 1108 (Mass. 2015).
[375] *Id.* at 1104-05.
[376] *Id.* at 1111 (discussing Commonwealth v. Given, 808 N.E.2d 788 (Mass. 2004)).
[377] *Id.* at 1112.

IV. Conclusion

As the above legal discussion reflects, there are not many cases addressing a defendant's right to access of the relevant source code consistent with a constitutional right. In most of the breathalyzer cases, the defendants and their attorneys failed to properly present the various constitutional challenges such that there were very few discussions on the merits. Even in the few instances of decisions on the merits, the courts fail to understand the implications of the requests. For example, in *Camblin*, the court granted the request because it was a new technology; that, though, is not the fundamental issue.

Overall, the problem is that the courts have applied the wrong standard. This is exemplified by the Minnesota ruling in *Garberg*, where the court wrote that the defendant "failed to make a threshold evidentiary showing that the source code information may relate to his guilt or innocence, negate his guilt, or reduce his culpability."[378] In other words, the court felt that software should be presumed reliable, and that without specific indications to the contrary Garberg could not have access to the source code. This is in stark contrast to how human witnesses are treated, where defendants have a constitutional right to try to impeach their credibility. Software does not deserve elevated status. Defendants must have access to evidentiary source code.

Implementing this policy requires several modifications to current practice. First, of course, is a rule on when defendants should have such access. We suggest adapting the definition of "software independence" proposed by Professor Rivest for elections: "A voting system is *software-independent* if an undetected change or error in its software cannot cause an undetectable change or error in an election outcome."[379] Note carefully that this does not rule out the use of software; systems where software proposes an answer but a manual process verifies it can still be software-independent.[380] Here, we propose that if evidence is not

---

[378] State v. Garberg, No. A09-914, 2010 WL 772622, at *3 (Minn. Ct. App. Mar. 9, 2010).
[379] Rivest & Wack, *supra* note 164, at 3761.
[380] In election systems, this can be accomplished by risk-limiting audits, a statistical process that involves manual counting of a small subset of the ballots cast, even if they were tallied by

analogously "software-independent", the defense should be entitled to examine the software that produced it.

A second crucial point is that the source code must be available for production by the prosecution. Procurement contracts must specify delivery of source code for each version of the code or device. This has to be part of acceptance of the product by the government, with no payment tendered or use of the product until the source code has been delivered. Such a policy would avoid situations as in *In re Commissioner of Public Safety,* where the Commissioner had to be ordered to obtain the code,[381] or *Bonakoske*, where the vendor asserted that it no longer possessed the code.[382]

Finally, defendants must have reasonable access. Auditing a program is not done simply by staring at printouts of the code; rather, a variety of specialized tools are used, both to examine the code[383] and to do test executions.[384] Audits may require teams; furthermore, it must often be contracted out to specialist companies. Reasonable protective orders may be required,[385] but they should not be overly onerous, as in *Peters.*[386] Even these may not be enough. As we noted, training data must be provided if machine learning algorithms are involved.[387] Professor Bratus et al. suggest the need for information on external inputs, e.g., the time of day, the configuration of the operating system and supporting software, and the specific version of the evidentiary software in use.[388]

---

a computer. *See, e.g.*, Mark Lindeman & Philip B. Stark, *A Gentle Introduction to Risk-limiting Audits*, 10 IEEE SEC. & PRIV. 42, 42 (2012).

[381] 735 N.W.2d 706, 709 (Minn. 2007) (The Commissioner had to be ordered to "*obtain* and provide to [Underdahl's] counsel the complete computer source code.") (emphasis added).

[382] Affidavit of Brian Faulkner at ¶ 5, State v. Bonakoske (Fla. Charlotte County Ct. June 10, 2011) (No. 08-1726-T).

[383] A simple example of such a tool is one that finds each location where each variable is used.

[384] One well-known technique for finding bugs at runtime is known as "fuzzing." *See, e.g.*, Andy Greenberg, *Hacker Lexicon: What is Fuzzing?*, WIRED (June 2, 2016), https://www.wired.com/2016/06/hacker-lexicon-fuzzing/ [https://perma.cc/4L2N-D786].

[385] *But see* Wexler, *supra* note 11.

[386] State v. Peters, 264 P.3d. 1124 (Mont. 2011).

[387] *See* Vincent, *supra* note 178.

[388] Sergey Bratus et al., *Software on the Witness Stand: What Should It Take for Us to Trust It?*, *in* TRUST AND TRUSTWORTHY COMPUTING 396, 406 (Alessandro Acquisti et al. eds., 2010).

Arguably, providing access to source code will impose a burden on the prosecution. Many defendants will be economically unable to take advantage of such opportunities. But neither consideration is unique.

From a technical perspective, it is a virtual certainty that any significant software system contains bugs. As Christian Chessman wrote, "[i]t is difficult to overstate the fallibility of computer programs."[389]

As shown in *Chun* and *Foley*, breathalyzers and DNA-matching devices are not immune.[390] These bugs may or may not affect the results in any given case, but in our judicial system, the Constitution and fundamental considerations of due process demand that defendants should have the opportunity to find out—and that is only feasible with access to source code.

---

[389] Christian Chessman, *A "Source" of Error: Computer Code, Criminal Defendants, and the Constitution*, 105 CALIF. L. REV. 179, 228 (2017).
[390] State v. Chun, 943 A.2d 114, 126-31 (N.J. 2008); Commonwealth v. Foley, 38 A.3d 882, 887 (Pa. Super. Ct. 2012).