OFFICE OF THE DIRECTOR OF NATIONAL INTELLIGENCE

STONESOUP

Securely Taking On Software of Uncertain Provenance

Intelligence Advanced Research Projects Activity

IARPA
BE THE FUTURE

LEADING INTELLIGENCE INTEGRATION

STONESOUP Phase 3
Test and Evaluation Execution and Analysis System (TEXAS)
System Design Document
12 December 2014

This report was prepared by TASC, Inc., Ponte Technologies LLC, and i_SW LLC. Supported by the Intelligence Advanced Research Projects Activity (IARPA), Research Operational Support Environment (ROSE) contract number 2011-110902-00005-002. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation hereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA or the U.S. Government.

Table of Contents

List of Figures

List of Tables

1 Overview

The scope of this document is to cover the system design of the "Test and Evaluation, eXecution, Analysis System" (TEXAS) for the STONESOUP Phase 3 Test and Evaluation activity.

This System Design Document (SDD) includes these sections:

- **Section 2 TEXAS System**: An overview of TEXAS, its processes and the descriptions of roles and actors.
- **Section 3 Test System Components**: A view of the infrastructure, test host on the performer network, and components of the administrative network.
- **Section 4 Test Cases**: Description and characteristics of Test Cases.
- **Section 5 Test System Operation**: Describes the specific steps of testing as the systems works its way through the selection, running and scoring of a test case.
- **Section 6 Cloud Based Test Environment**: Describes the cloud based IT infrastructure services and TEXAS' Amazon Web Services Implementation.
- **Appendix A Document Map/References**: Demonstrates how this System Description Document works with the other STONESOUP documents to provide the appropriate context and detail to understand TEXAS.
- **Appendix B Applicable Department of Defense Architecture Framework (DoDAF) Views**: Defines the DoDAF views which are applicable to the TEXAS system design.

2 TEXAS System

2.1 System Overview

TEXAS was designed and developed to test the ability of software prototype applications created to detect and mitigate software vulnerabilities and exploits through static analysis and run time countermeasures. These applications, by the STONESOUP Performers, were developed to provide automated techniques that allow end users to securely execute software without basing risk mitigations on characteristics of provenance that have a dubious relationship to security. The TEXAS system will exercise the technologies and evaluate the success and statistically measure performance through high volume testing.

An overview diagram of how TEXAS interacts with the Performer's technologies is shown in **Figure 1** below. Bold solid lines reflect data and information flow, whereas dashed lines reflect commands initiated by actors.

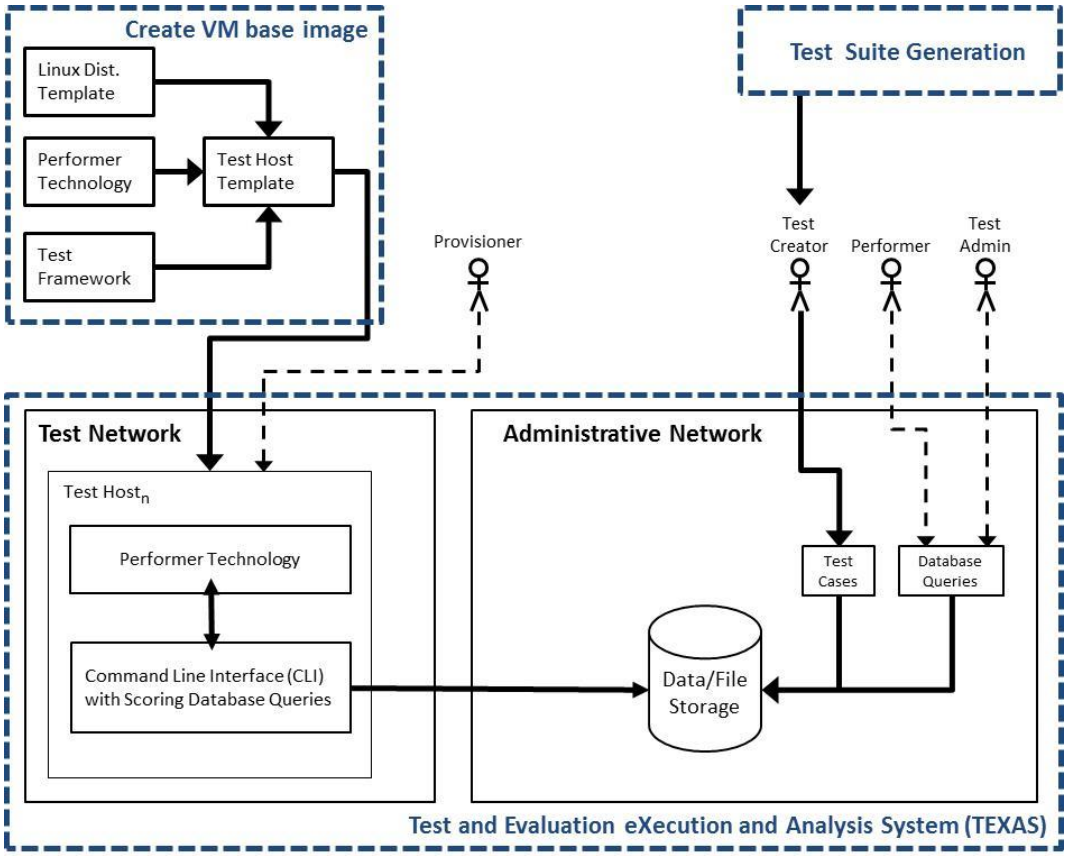


Figure 1. TEXAS Functional Architecture showing Administrative and Performer Networks

The Functional Architecture to support Test and Evaluation is composed of multiple components and actors that must work together to execute the tests necessary to validate the technology. In alphabetical order these include:

- **Administrative Network**: Where the test cases are managed, presented to the performer network for execution, and results are stored and presented to the test administrator (**Section 3.3**)
- **Command Line Interface (CLI)**: Allows a test engineer or performer technology developer to run the "analyze" or "execute" commands, and receive immediate feedback in regards to the results (**Section 2.2.2**)
- **Data Base Queries**: Query scripts are written to get scored test results from the administrative subnetwork database (**Section 0)**
- **Data/File Storage Component**: A data repository for test case setup information, scoring metrics, sufficient information to rescore tests, and review/tracing of execution issues (**Section 0**)
- **LINUX Distribution Templates**: A Linux Distribution with Performer Technology and a Test Framework installed (**Section 3.2.1.1**)
- **Performer**: An actor who develops the technology application which mitigates vulnerabilities tested during T&E and who can view scored test results from the administrative subnetwork database (**Section 2.4.4**)
- **Performer Technology**: Applications created to detect and mitigate software vulnerabilities and exploits through static analysis and run-time countermeasures (**Section 3.2.1.3**)
- **Provisioner**: An actor who adds new workstations, servers, or entire subnetworks to the test infrastructure as needed, configuring each new platform from a predefined virtual machine template for the role it will play in the overall architecture (**Section 2.4.1**)
- **Scoring Database Queries**: Computes statistics across the database of completed tests (**Section 3.3.2**)
- **Test Administrator**: An actor who conducts testing according to the Rules of Engagement and tuning and exercising oversight of the test infrastructure to successfully complete testing (**Section 2.4.3**)
- **Test Cases**: Specification of a vulnerability test, including parameters and test approach that is inserted into the Test Framework and produces an outcome and a score indicating the acceptability of the Performer Technology in mitigating the vulnerability (**Section 4**)
- **Test Creator**: User/system component that creates Test Cases and provides them (base program with injections, metadata, I/O pairs) to TEXAS (**Section 2.4.2**)
- **Test Framework**: Automated system that interacts with other test functions to install and configure a test case, invoke the performer's technology, collect the test results, and send the results data to storage (**Section** Error! Reference source not found.)
- **Test Host**: Instantiation of a Test Host Template on the Amazon Web Service framework built using an image provided by the performers to the T&E team (**Section 3.2**)
- **Test Host Template**: Snapshot of a machine instance that has both a Performer Technology and a Test Framework installed that can be quickly provisioned (**Section 3.2.1**)

- **Test Network:** Where software packages developed by the performers are subjected to test cases designed to evaluate the efficacy of the mitigation techniques under a variety of environments and inputs (**Section 3.1.1**)
- **Test Suite Generation**: Based on the Test Data Generation Plan (TGP), the Test Suite Generation component generates a list of test case identifiers following criteria for a uniform distribution of test cases (**Section 3.3.1**)

2.2 TEXAS Processing

2.2.1 Test Stages

The primary requirement for a successful test and evaluation of performer technologies is TEXAS executing test cases and returning results. The full execution of test cases occurs in three main steps: Analysis, Execution, and Scoring. See **Figure 2. TEXAS Stage 1 and Stage 2 Processing**.

- **Analysis**: The source code or binary of a program is scanned looking for CWE code patterns and applying diversification techniques to harden the resulting binary. The output of the Analysis phase is a hardened binary executable.
- **Execution**: The Execution step is run for each I/O, as defined in the Test Case's metadata, and involves actually invoking the hardened binary created in the Analyze step with known inputs (both benign and exploiting). Performer technology may also monitor the execution of the binary to look for execution patterns indicative of an attack in progress or software vulnerability.
- **Scoring**: Scoring executed immediately after the Execution step and looks at the environment for the known outputs defined in the metadata for the given I/O pair that was executed.

The Analysis, Execution, and Scoring steps of the test case are collectively known as a stage. Test cases execute through two stages, Stage 1 and Stage 2.

Stage 1 occurs without performer technology and confirms the effectiveness of the fault injection on the base program. If Stage 1 is scored as completely valid, Stage 2 occurs with performer technology. This stage tests the efficacy of the Performer Technology in mitigating the injected fault without altering the behavior of the program.

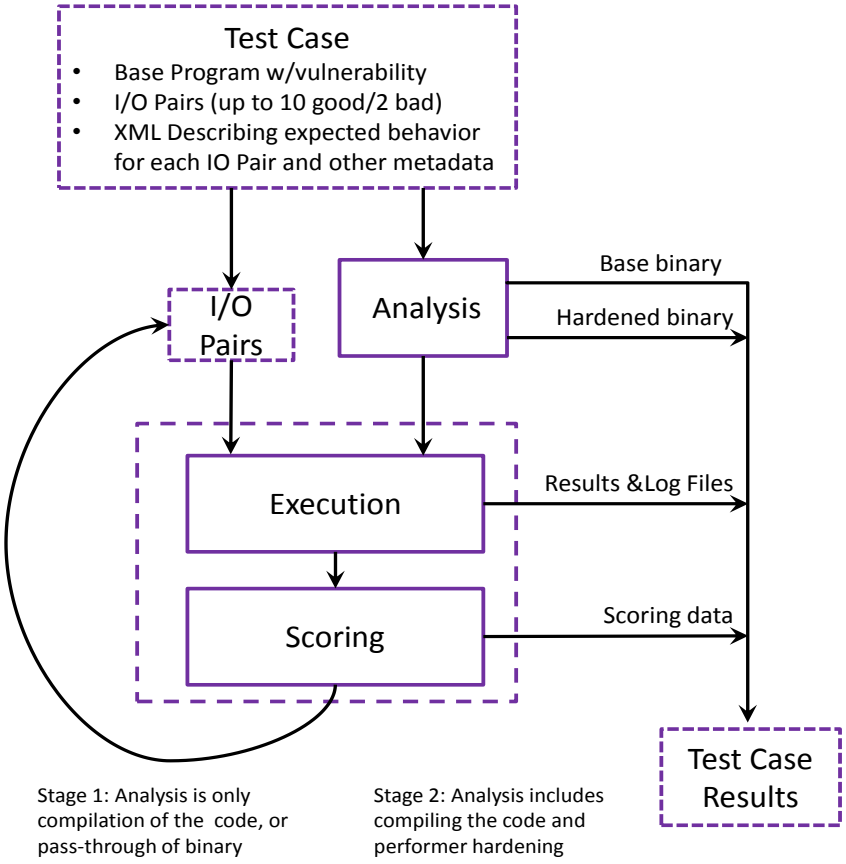


Figure 2. TEXAS Stage 1 and Stage 2 Processing

Scoring of successful mitigation is an essential part of this operation and will be built in during design and development. These and other components are described in detail in the following sections. See **Table 1. STONESOUP Stages 1 and 2 Attributes for Scoring,** for additional details on scoring.

Table 1. STONESOUP Stages 1 and 2 Attributes for Scoring

Stage 1 Attributes	Stage 2 Attributes
Fault Injected Base Program	Fault Injected Base Program after Analysis by Performer Technology
Operated "Normally"	Operated "Normally"
Success is measured by the T&E Team by:	
Bad I/O pairs trigger expected Vulnerability	Bad I/O pairs should trigger expected Vulnerability but Performer technology may mitigate the Vulnerability
Good I/O pairs trigger no Vulnerability	Good I/O pairs should not trigger a Vulnerability
No unusual behaviors detected	No unusual behaviors detected
Measured execution time reasonable	Measured execution time – increase < limit
Measured Vulnerabilities	Measured Vulnerability issues

For a high level view of the expected functions, sequencing and some data and control linkages expected, see **Figure 3. TEXAS Process Control Flow for Stage 2 of the Testing Process.**

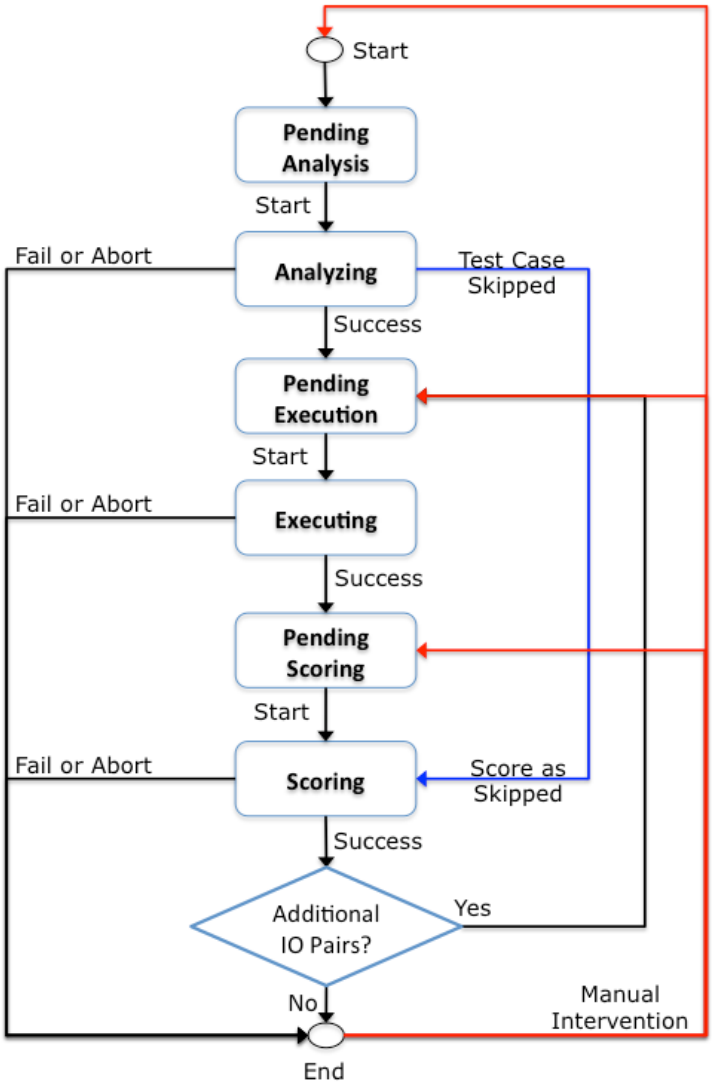


Figure 3. TEXAS Process Control Flow for Stage 2 of the Testing Process

The individual steps are described in **Table 2. STONESOUP Logical Components.**

Table 2. STONESOUP Logical Components

Logical Operation	Outcome	Details
Start		User initiates a test case run
Pending Analysis	Start	Analysis of a test case is started
Analyzing	Success	Stage 1: Regular binary was produced Stage 2: Hardened binary was produced
	Fail or Abort	Binary not produced, operation aborted by user, or timeout

Logical Operation	Outcome	Details
	Test Case Skipped	Stage 1: Not Applicable Stage 2: Performer sends Skip message via Communications API
Pending Execution	Start	A queued execution of an I/O Pair is started
Executing	Success	All processes and run command were able to execute; timeout can still result in a Success outcome
	Fail or Abort	A pre-, co-, or post-process or the run command failed to actually execute; or the operation was aborted by the user
Pending Scoring	Start	A queued scoring of an I/O Pair is started
Scoring	Publish	All output checks evaluated, scoring formula results successfully calculated, and score published
	Fail or Abort	An output check failed to evaluate or was missing; scoring formula failed to evaluate; or operation aborted by user
Additional IO Pairs?	Yes	There are additional I/O Pairs to be executed
	No	There are no additional I/O Pairs to be executed
End	Manual Intervention	After a test case has reached the end of run, the user may choose to manually re-queue entire test case, an execution, or a scoring
	Completion	Test case run is complete

2.2.2 Command Line Interface

TEXAS is designed and developed to test a Performer technology's ability to detect and mitigate software vulnerabilities and exploit through static analysis and run time countermeasures. To this end, the TEXAS system includes a Command Line Interface (CLI) to support the two major testing stages, Stage 1 and Stage 2, and the three major testing workflows within each stage, i.e. Analyze, Execute, and Score.

The provided TEXAS CLI exists to allow a test engineer or performer technology developer to run the "analyze" or "execute" commands, and receive immediate feedback in regards to the results. If performing an execution workflow, one will also receive a score. In addition to allowing execution of individual workflows, the TEXAS CLI also provides a validate command for convenience that performs both an "analyze" and multiple "executes" in a single invocation.

The TEXAS CLI is designed and developed to execute on a Linux operating system. While there are no specific Linux distribution requirements, all system testing has been performed against those listed below. Other distributions may also work, but are not specifically tested.

- Ubuntu 12.04 LTS (x86_64)
- CentOS 6.5 (x86_64)

TEXAS is implemented largely in Python and requires a Python 2.7.5 execution environment. All TEXAS BASH scripts are developed for BASH 3 or greater. For more information see the "STONESOUP TEXAS CLI Users Guide".

2.4 Actors/Roles

Actors and roles are used to describe actions performed within TEXAS to accomplish various T&E related actions.

- Actor: someone or something that "acts" on or with the system.
- Roles: a set of needs, behaviors, and expectations.

The four actors/roles in STONESOUP are:

- Provisioner
- Test Creator
- Test Administrator
- Performer

2.4.1 Provisioner

The Provisioner ensures that TEXAS has enough resources for optimum performance. Utilizing Cloud Platform-as-a-Service (PaaS) services, this actor will configure the initial setup and also add new workstations, servers, or entire subnetworks to the test infrastructure as needed. Each new platform will be configured from a predefined virtual machine template for the role it will play in the overall architecture. Once leased, configured, and initialized, the installed TEXAS client software on each new platform will begin communicating with the appropriate TEXAS services and requesting analysis tasks to perform. While configuring resources, the Provisioner also assesses costs and assists the Test Administrator to ensure resources stay within cost targets.

2.4.2 Test Creator

The Test Creator creates Test Cases and provides them to the TEXAS. The methodologies for creating the Test Cases are covered in the “Test Data Generation Plan.” For more information on Test Cases, see **Section 4**.

2.4.3 Test Administrator

The Test Administrator is responsible for conducting testing according to the Rules of Engagement. The Test Administrator needs to be able to:

- Execute individual test cases to run
- Tuning and exercising oversight of the test infrastructure
- Launch Test Cases on Test Hosts
- View results that are scored and summarized in Dashboard views
- Use the system interface to pause and resume testing
- Review automatic scoring
- Select the next set of test cases to be run
- Modify the size of the queue and number of available virtual machines - speed up or slow down adaption to test case results and influence the orchestrator attempts to

maintain a flat distribution of completed tests across performers, weaknesses and other criteria

2.4.4 Performer

The Performer can interact with the system to reject a certain number of test cases as described in the STONESOUP Broad Agency Agreement. The Performer can also instruct sequential execution of a test case to allow their technology to learn how to mitigate an exploit.

3 Test System Components

Physical or virtual system that contains the Linux distribution, modified with the requisite test framework components and the performer technology. To accommodate the performers, the Test Hosts will be modified to meet their specific operating system requirements, technology and supporting artifacts.

An instance of a Test Host Template is an operating system image that includes the base programs and Test Harness that has been provisioned and is available for running Test Cases.

3.1 Infrastructure

There will be multiple networks to accomplish testing. The main T&E network contains the major components contains the primary components (e.g. Provisioner, Orchestrator, and Scorer) and the performer network is for the exclusive use of the performers. The network configuration is shown in **Figure 1. TEXAS Functional Architecture showing Administrative and Performer Networks**.

3.1.1 Test Network

The test network is where software packages developed by the performers are subjected to test cases designed to evaluate the efficacy of the mitigation techniques under a variety of environments and inputs. The performer networks will be limited subsets of the main T&E network. There will be a limited number of test hosts at any one time.

3.1.2 Administrative Network

The Administrative Network controls access via the Internet to TEXAS for the performers and the T&E Team. The administrative network also controls the distribution of test cases to the performer network, stores the results of execution, and provides the dashboard interface to the Test Administrator.

3.1.3 Communications API

For Phase 3 the T&E team extends the Communications Application Programming Interface (API). This was done to account for shortfalls in Phase 2 testing. The Communications API provides functionality for input and output, control functions and general interactions between the Test Harness and the performer technology. The Communications API is explained in more detail in the Communications API document. Key reasons for the upgrade were:

- New messages
- Confirmation of specific actions
- Better control and understanding of test state

The goal of the extension/changes are to run more tests more accurately, acquire better T&E results, and minimize API changes to reduce code rework on both the test team and the performers.

3.2 Test Host on Performer Network

The Test Host is an instantiation of a Test Host Template on the Amazon Web Service framework built using an image provided by the performers to the T&E team. The Test Host is modified prior to testing to include base, Testing Framework and modified base programs for executing tests. The AMI contains the test framework to enable communications with the test broker, scorer, etc.

3.2.1 Test Host Template

3.2.1.1 Linux Distribution Template

This is a Linux Distribution with Performer Technology and a Test Framework installed.

External Dependency: Any external dependencies, or software component or system accessible from a Test Host that is required for the execution of the Performer Technology but is external to the Performer Technology and thus is not part of the Test and Evaluation.

3.2.1.2 Command Line Interface (CLI) with Scoring Database Queries

The Command Line Interface (CLI) with Scoring Database Queries is the TEXAS software that provides the capability to execute 'analyze' and 'execute' steps of a test case in both Stage 1 and Stage 2.

3.2.1.3 Performer Technologies

Software developed by the performers designed to accomplish the STONESOUP goals of securely using executable of unknown provenance and rendering any exploits or bad behavior benign to the host operating system automatically.

Each Performer technology will be hosted on a machine of appropriate capabilities. Performers will be asked to provide their technology suitable for loading onto the appropriate virtual machine image. These images will be stored for use during test execution and used to initiate test operations without modification and to scale capability as required to meet test load. Each Performer may have unique target machines and/or special requirements to be considered. This information is captured in **Section 3.1.1**.

3.3 Components Associated with Administrative Network

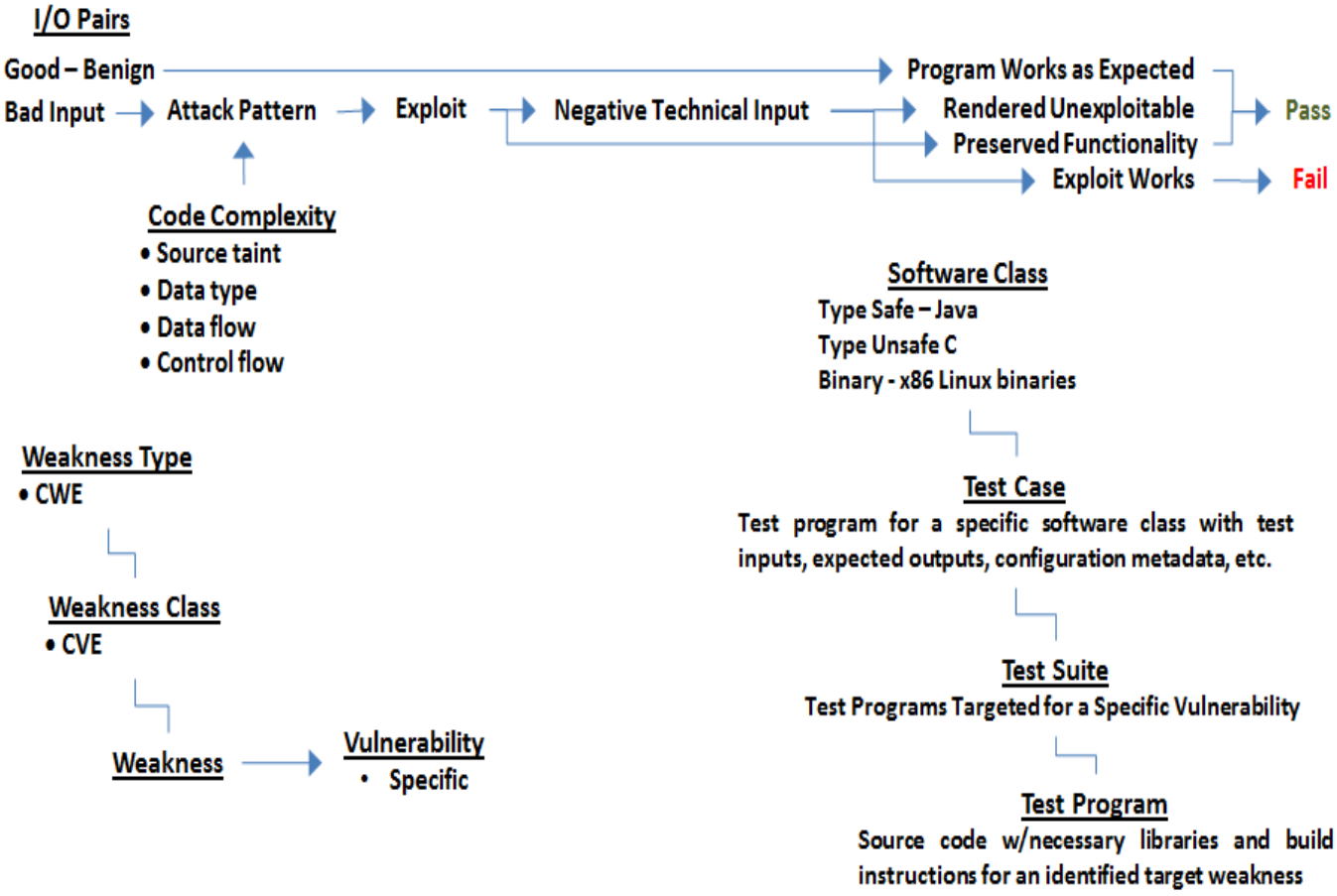
The Test Suite Generation, Scoring Database Queries and Data/File Storage Services are components supporting functions in the Administrative Network.

3.3.1 Test Suite Generation Component

Based on the Test Data Generation Plan (TGP), the Test Suite Generation component generates a list of test case identifiers following criteria for a uniform distribution of test cases. This component attempts to maintain an even distribution of executed test cases across the performers and is spread across the criterion of base programs, common weakness enumerations (CWE), algorithmic variants, code complexity features, and injection point. This test suite list is then used to generate a corpus of test cases for the performers that is queued up and placed in the Data/File Storage database.

The Test Suite Generation component is designed to ensure that if testing were to be terminated at any time the resultant test results would have a flat distribution and provide for a fair assessment for all participants based on completed results. It prioritizes tests based on the history of tests executed so far, in order to maintain a flat distribution of executed tests. The overall guiding principle for the Test Suite Generation component is the mandate that "at any moment we stop, the distribution of test cases executed should be flat." **Figure 4. Test Composition Criteria Factors that Lead to a Flat Distribution of Test Cases,** illustrates the overall structure and depicts how the Test Suite Generation component maintains a flat response of test articles across performers, weaknesses and other criteria.

Figure 4. Test Composition Criteria Factors that Lead to a Flat Distribution of Test Cases



3.3.2 Scoring Database Queries

The Scoring Database Queries computes the results of an individual test case and optionally uploads the results to the administrative subnetwork database. Query scripts are provided to then get scored test results from the administrative subnetwork database. Queries could also be used to rescore tests based on alternate algorithms or the presence of filters defined by the Test Administrator based on review of anomalous results.

The Scoring Database Queries generates the result of a completed Test Case on a Test Host. It is made up of the binaries (both the base Performer binary that is built before analysis and the output binary that is result of the Test Case execution), the outcome of the Test Case, and the Score. The Scoring algorithm returns results of:

- "Program Works as Expected"
- Exploit was "Preserved Functionality (PF)"
- Exploit was "Rendered Unexploitable (RU)"
- Exploit had "Controlled Exit (CX)"
- Exploit was "Rendered Unexploitable with Learning (RUL)" with some negative impacts, or
- "Exploit Works" and it was a failure

An example of the Scoring Database Queries is shown in **Figure 5. Scoring Algorithm Example of Performer Learning through Successive Test Runs**.

During test execution, the performers are provided latitude in asking that some results be excused to account for differences in test result interpretation or need for minor re-work. Tests eliminated at the request of the performers are added back into the queue and the queue is adjusted based on the above paragraph.

- Good I/O $_{i,j}$ where i= Test Case ID and j = Good I/O pair test number
- Bad I/O $_{i,j}$ where i= Test Case ID and

  j = a (i.e., test case$_i$ run first time *without* learning mitigation)or j = b (i.e., test case$_i$ run second time *after learning* mitigation)

- Steps 1-6 are the First Run of Test Case$_i$ with Learning mitigation in steps 5 and 6
- Steps 7-12 is First Run with Learning mitigation in steps 11 and 12
- Steps 13+ Passed Preserved Functionality Check

1.	Good I/O$_{1,1}$	Pass	PF
2.	Good I/O$_{1,2}$	Pass	PF
3.	Good I/O$_{...}$	Pass	PF
4.	Good I/O$_{1,10}$	Pass	PF
5.	Bad I/O$_{1,a}$	Fail – Controlled Exit	CX
6.	Bad I/O$_{1,b}$	Normal Termination	CX
7.	Good I/O$_{2,1}$	Pass	PF
8.	Good I/O$_{2,2}$	Pass	PF
9.	Good I/O$_{...}$	Pass	PF
10.	Good I/O$_{2,10}$	Pass	PF
11.	Bad I/O$_{2,a}$	Pass	RUL
12.	Bad I/O$_{2,b}$	Pass	RUL
13.	Good I/O$_{3,1}$	Pass	PF
14.	Good I/O$_{3,2}$	Pass	PF
15.	…		

PF = $\wedge_i$ Good$_i$

RU = $\wedge_{i,j}$ ~Bad$_{i,j}$

Figure 5. Scoring Algorithm Example of Performer Learning through Successive Test Runs

3.3.2.1 Scoring Test Cases

There is a hierarchy when setting up test cases. The performers will be able to reject test cases where they determine some condition has prevented their technology from performing properly and affecting its evaluation. For some test cases the performer can choose to reject a test before it becomes a completed test and is added to their scoring criteria.

All test cases are tracked in a matrix and rolled up to the CWE level (the level test case development is assigned) as well as weakness class level. Areas that impact this selection and tracking process include working through base program selection, success working with ROSE, finalizing the metadata specification, determining TEXAS file structure and others. Performers are provided preliminary test results to make their own decisions on their technologies performance. They are allowed to skip or ask a certain percentage of tests be re-executed, and for some performer technology, there is a learning component that enables the performer technology to not only avoid a controlled exit, but to use sequential execution to train their technology to render the exploit unexploitable and continue execution.

Data/File Storage Component

Mongo databases are used to store test data, test results and the archive files produced by runs using GridFS. It also stores information used by the Test Administrator to control the overall evaluation effort, such as the current processing status of each test case and the aggregate results from all tests. Data storage services are accessible to the performer subnetworks and the administrative network. By using a single data store, movement of data around the test network will be minimized. The information stored in the Data File Storage will include:

- Test case setup information
- Scoring metrics
- Sufficient information to rescore tests
- Review/tracing of execution issues.

Specific data will be described in implementation documentation.

4 Test Cases

Test Cases are a specification of a vulnerability test, including parameters and test approach that is inserted into the Test Framework and produces an outcome and a score indicating the acceptability of the Performer Technology to the vulnerability. Test Cases are programs merged with I/O pairs and with XML that describes the test case and how it is to be used and scored.

Test Cases are created by the Test Data Generation Team. See the Test Data Generation Plan for more information on Test Cases.

4.1 Input-Output (I/O) Pairs

Each individual Test Case is associated with multiple I/O-pairs that drive the evaluation. These IO-pairs are split into GOOD I/O-pairs and BAD I/O-pairs.

- A GOOD I/O-pair refers to an I/O-pair with a well-formed input and an expected output. The input of a GOOD I/O-pair is used to verify that a performer technology does not alter the application's expected functionality.
- A BAD I/O-pair refers to an I/O-pair with input that is expected to result in an exploit. This I/O-pair could be malformed or malicious data specifically crafted to exploit the target weakness. Each BAD I/O-pair is associated with a technical impact that helps define the expected exploit.

A test case will have multiple eight GOOD and two BAD I/O-pairs.

5 Test System Operation

STONESOUP test system operation is shown in **Figure 6. TEXAS Concept of Operations Operational View 1 (OV-1),** below. The steps listed here describe the specific steps as the T&E environment works its way through the selection, running and scoring of a test case. The overall process is similar for Stage 1 and Stage 2 processing discussed earlier.

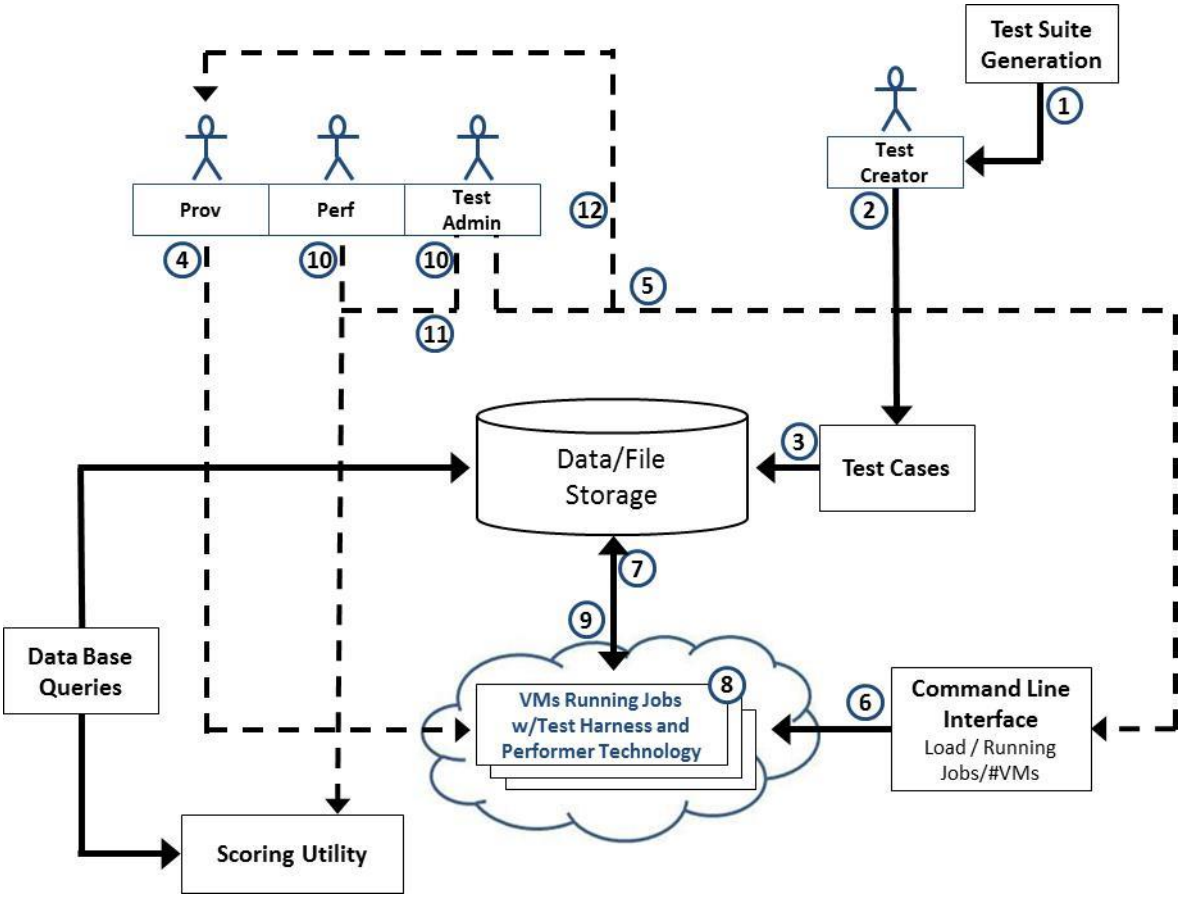


Figure 6. TEXAS Concept of Operations Operational View 1 (OV-1)

1. Test Suite Generation component generates list of Test Case identifiers for Test Suite
2. Test Creator validates uniform distribution in the Test Suite
3. Test Creator generates Test case metadata and loads in Data/File Storage as Test Corpus
4. Provisioner initiates host Virtual Machines (VM)
5. Test Administrator activates test queues in Data/File Storage
6. Test Administrator initiates Job Runs through Command Line Interface (CLI)
7. Test meta data needed for Job Runs passed from Data/File Storage
8. Job runs on host VM
9. When Job completes, sends results to Data/File Storage
10. Data Base queries (Performer, Test Admin) go through Scoring Database Queries

11. Test Administrator reads information about previous results from Scoring Database Queries, which reads test results from Data/File Storage query and generates score
12. Test Administrator requests Provisioner to adjust test infrastructure, as required to complete T&E

6 Cloud-Based Test Environment

Cloud computing as a scalable architecture offers distributed IT infrastructure services. Commercial cloud computing services provides the opportunity to avoid procuring servers and other IT infrastructure and instantly provision additional servers in minutes to respond to TEXAS testing needs. TEXAS will utilize Amazon Web Services to satisfy these needs.

6.1 Infrastructure

As seen in **Figure 1. TEXAS Functional Architecture showing Administrative and Performer Networks**, TEXAS sub-domains are setup with private virtual private network (VPN access) for the T&E team and performers. The test network provides secure authenticated communication channels between all TEXAS components. Performers see a single virtual network connecting the performer subnetwork and the administrative subnetwork.

TEXAS will utilize subnets for the main T&E enclave and smaller sites for each performer and the Independent Verification and Validation teams to use. All of these will be connected to the Internet and to each other to share resources via virtual private networks (VPN).

The administrative subnetwork consists of virtual machines that host TEXAS servers that implement the test manager, orchestrator, broker, scoring Database Queries, and analysis functions, as well as data and file storage services. These functions and services facilitate the management of test execution, collection, and analysis of test results.

Each performer will have a dedicated subnetwork of virtual machines running TEXAS clients and their technology. The subnetwork will also include any external dependencies needed by the performer's technology or for the execution of individual tests, e.g., a SQL database, DNS server, Web server, or IRC Chat Server. Each host workstation in this subnetwork running a performer's technology will be configured as either an analysis or execution host for that specific performer technology, with appropriate memory and computing resources. Performers can install the latest version of their technology on a set of networked host machines in their subnet with TEXAS client software. Select individual test cases will be available and the TEXAS client can be scripted to run individual analysis. This permits each performer to perform regression tests on new releases of their software or new releases of the TEXAS client software.

6.2 Amazon Web Services Implementation

6.2.1 Elastic Compute Cloud Hosts

Elastic Compute Cloud (Amazon EC2) is a web service that provides resizable compute capacity in the cloud. It is designed to make web-scale computing easier for developers. The benefits of TEXAS utilizing Amazon EC2's capabilities include:

- Lower Cost: No up-front expenses or long-term commitments with the ability to build and manage a global infrastructure at scale. Pay for capacity as used.

- Agility and Instant Elasticity: The T&E team and performers can innovate, experiment and iterate and instantly deploy new technology versions in protected virtual machines as workload grows, and instantly scale down based on demand
- Open and Flexible: Language and operating system agnostic platforms with built in support for performer platforms/programming models and the ability to add TEXAS specific images as needed.
- Secure: Secure, durable platform with industry-recognized certifications and audits: PCI DSS Level 1, ISO 27001, FISMA Moderate, HIPAA, and SAS 70 Type II. Multiple layers of operational and physical security to ensure the integrity and safety of performer and TEXAS data.
- Through scalable architecture controls, the T&E team can monitor usage while allowing performers virtually unfettered access to independently test their technology in the actual test environment.
- Complete control of computing resources
- Proven computing environment.

6.2.2 TEXAS Amazon Storage

Storage is a software component that provides storage and retrieval of Test Case Metadata, Score Results, Results Archive, and Test Case Archive.

The archive is a snapshot of a set of files or binaries that represent the state of the Performer Technology and other relevant files on the Test Host. Archives are taken after the Analysis Task and the Execute Task.

A high-capacity database will be used to store test data and test results. It also stores information used by the Test Administrator to control the overall evaluation effort, such as the current processing status of each test case and the aggregate results from all tests. The data storage services will be accessible to the performer subnetworks and the administrative network. By using a single data store, movement of data around the test network will be minimized.

Storage will utilize Amazon Elastic Block Storage (EBS) storage. EBS is a Network-Attached Storage (NAS) system that can be mounted as a file system and accessed from within an EC2 instance as a virtual storage device (e.g., a virtual hard disk drive). EBS will be used for TEXAS data and file storage. Only a relative modest amount will be required at the beginning of the project, ramping up sharply for the later dry runs and the actual final T&E testing. Based on predicted Performer usage, storage is projected to peak at approximately 75 terabytes (TB), based on the number of test cases and the size of the logging data (~2 GB) that will be captured for each test execution. This is almost twice the storage used by MITRE in Phase 2 (40 TB), but the increase is justified by the five-fold increase in the size of each test case base program and increase in logging to mitigate the risk of ambiguous results. The cost of 75 TB of EBS storage

for the final T&E month is only $7,500.00, so for any risk that can be mitigated by increased storage, the cost of that approach would not be prohibitive.

All possible AWS configurations (for the type of Amazon Elastic Compute Cloud (EC2) pricing model) are available on the Amazon web site at http://aws.amazon.com/ec2/instance-types/.

6.2.3 Linux Amazon Machine Instances

Amazon Machine Instances (AMI) are images of Linux operating system installations suitable for instantiation by a virtual machine. These could be standard AWS Amazon Machine Instances (AMI) or custom images based on performer technology use and made public for their use. AMI's are pre-built instances that contain the base operating system, the Test Framework with test cases and require application libraries and the Performer technology. They are used to instantiate virtual machines for testing. Upon beginning a test sequence the AMI is used to set up the system for testing.

Appendix A—Document Map/References

The document map shown in **Figure 7. Program Documentation Relationships and Dependencies,** demonstrates how this System Description Document works with the other STONESOUP documents to provide the appropriate context and detail to understand TEXAS. The reader is encouraged to review these other documents as needed to complete or supplement the information contained herein.

A limited set of documents are envisioned for the System Design. All documents marked as "Needed, to be developed" are expected as part of the implementation documentation. Specifics of these documents will be developed as part of the implementation; other similar documents may be needed as well. Because Collaboration is a required functionality, a guide will be needed, in addition to the implementation plans developed. Additional documentation regarding Test cases and data generation will be created as well and will be included in the Test and Evaluation Plan. A full set of STONESOUP related documentation can be found in the Test and Evaluation Plan.

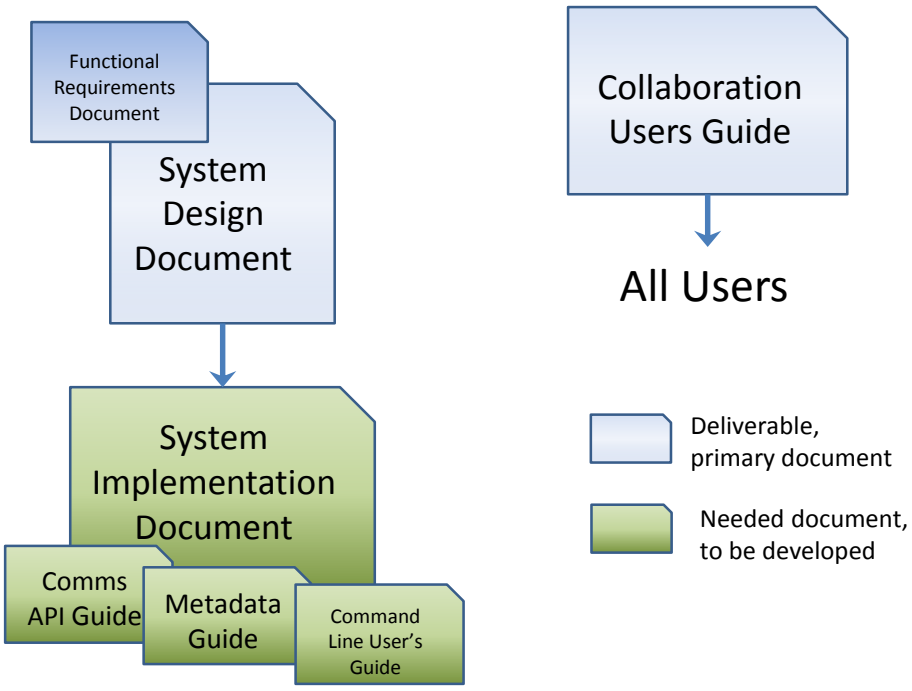


Figure 7. Program Documentation Relationships and Dependencies

Table 3. STONESOUP Document List and Descriptions, lists these documents with descriptions of their purpose and source documents.

Table 3. STONESOUP Document List and Descriptions

Document	Purpose	Sources
Collaboration User Guide v1	Describes the procedures and expectations of each participant for sharing information. Relies on data in this SDD.	Any Draft Collaboration Guidance
Functional Requirements Document	Describes requirements for the Test System, derived from Customer description of expected functions and controls.	SOO, RFP, other contractual documents
System Design Document (SDD)	High Level description of Test System operation, components, inputs, outputs, functional processes of the System, and interfaces required to operate.	SOO, RFP, other contractual documents
Communications API	Describe to performers the commands and interaction sequence for communication with TEXAS.	SDD, Technical Information from selected components
Metadata Guide	Describes the format and use of TEXAS metadata.	SDD, Technical Information from selected components
Command Line Interface Users Guide	Describes the operation and commands for the TEXAS command line interface.	SDD, Technical Information from selected components

Appendix B—Applicable Department of Defense Architecture Framework (DoDAF) Views

B.1. DoDAF Overview

DoDAF[1] is a family of related views into a system design that provides a common lexicon and format to describe components and relationships and this family is shown in **Table 4. DoDAF Views and their Relationships.**

Table 4. DoDAF Views and their Relationships
(Views Used for STONESOUP are Bolded)

Viewpoint/ Category	Tabular	Structural	Behavioral	Mapping	Taxonomy	Pictorial	Timeline
All Viewpoint	**AV-1***				**AV-2***		
Capability	CV-1	CV-4		CV-6 CV-7	CV-2		CV-3 CV-5
Operational	OV-3	OV-2 OV-4	OV-6a OV-6b OV-6c		OV-5	**OV-1***	
System	SV-6 SV-7 SV-9	SV-1 SV-2	SV-4 SV-10 a/b/c	SV-3 SV-5a/b			SV-8
Standards	StdV-1 StdV-2						
Data and Information		DIV-1/2/3					
Service	SvcV6 SvcV-7 SvcV-9	SvcV1/2	SvcV-4 SvcV-10 a/b/c	SvcV-3a SvcV-3b SvcV-5			SvcV-8
Project		PV-1		PV-3			PV-2

* The DoDAF views used in this document are as follows:

AV-1*: Overview and Summary - Describes a Project's Visions, Goals, Objectives, Plans, Activities, Events, Conditions, Measures, Effects (Outcomes), and produced objects

AV-2*: Integrated Dictionary - An architectural data repository with definitions of all terms used throughout the architectural data and presentations

OV-1*: High-Level Operational Concept Graphic - High-level graphical/textual description of the operational concept.

[1] Office of the Assistant Secretary of Defense Networks and Information Integration (OASD/NII), "Reference Architecture Description", June 2010, website accessed 11 June 2014.

B.2. TEXAS DoDAF Operational View 1 (OV-1)

Figure 8. STONESOUP End-User Oriented Operation View 1 (OV-1), shows a concept of operations for the end-user. STONESOUP operates in two Phases, shown as Steps 1 and 2 where they respectively, harden and modify the subject application, and Step 3 where the user performs normal operations as they would with any other application on their desktop or server.

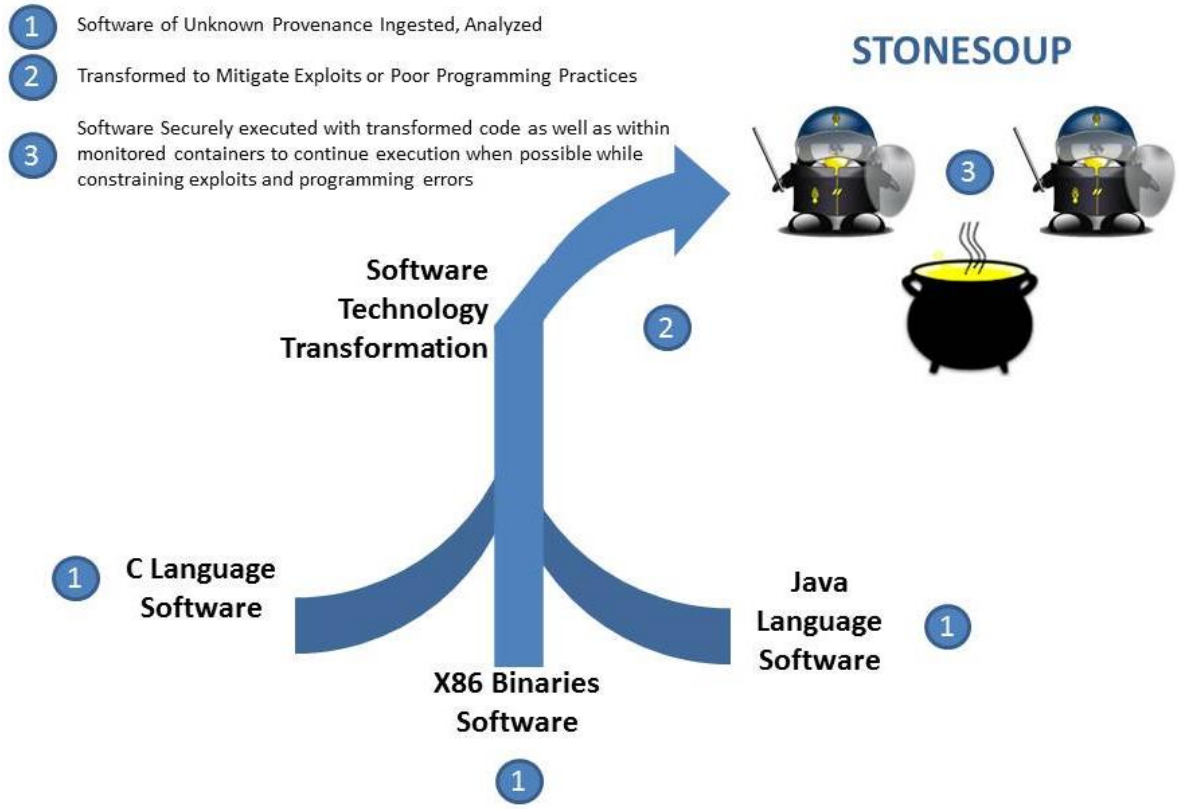


Figure 8. STONESOUP End-User Oriented Operation View 1 (OV-1)

B.3. TEXAS DoDAF Exhibit (AV-1)

Table 5. DoDAF All Views 1 (AV-1) Exhibit, describes the STONESOUP Vision, Goals, Objectives, Plans, Activities, Events, Conditions, Measures, Effects (Outcomes), and produced objects.

Table 5. DoDAF All Views 1 (AV-1) Exhibit

Method	Description	Advantages
Identification	Name	Securely Taking On New Executable Software Of Uncertain Provenance (STONESOUP) Program
Details	Security Classification	FOUO, Unclassified, or No Classification
PO	Approving Organization	Intelligence Advance Research Projects Agency (IARPA)
Scope	Views	N/A
Purpose	STONESOUP develops and demonstrates comprehensive, automated techniques that allow end users to securely execute software without basing risk mitigations on characteristics of provenance that have a dubious relationship to security. Existing techniques to find and remove software vulnerabilities are costly, labor-intensive, and time-consuming. Many risk management decisions are therefore based on qualitative and subjective assessments of the software suppliers' trustworthiness. STONESOUP develops software analysis, confinement, and diversification techniques so that non-experts can transform questionable software into more secure versions without changing the behavior of the programs.	

B.4. Terms and Definitions (AV-2)

Table 6. Terms and Definitions (AV-2), provides definitions of terms used in this document.

Table 6. Terms and Definitions (AV-2)

Term	Definition
Administrative Subnetwork	The administrative subnetwork consists of virtual machines that host the TEXAS servers which implement the test manager, orchestrator, broker, scoring, and analysis functions, as well as data and file storage services. These functions and services facilitate the management of test execution, collection, and analysis of test results.
Amazon Elastic Block Storage (EBS) storage	EBS is a Network-Attached Storage (NAS) system that can be mounted as a file system and accessed from within an EC2 instance as a virtual storage device (e.g., a virtual hard disk drive). EBS will be used for TEXAS data and file storage.
Amazon Elastic Compute Cloud (EC2)	Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides scalable compute capacity in the cloud. It is designed to make web-scale computing easier for developers.
Amazon Machine Instance	An Amazon Machine Image (AMI) provides the information required to launch an instance of a virtual server in the cloud. You specify an AMI when you launch an instance, and you can launch as many instances from the AMI as you need. An AMI includes the following: • Template for root volume instance (e.g., an operating system, application server, and applications) • Launch permissions that control which accounts can use the AMI to launch instances • Block device mapping that specifies volumes to attach to instance when launched
Amazon Web Services (AWS)	Amazon Web Services (AWS) is a preferred method for computing infrastructure. AWS offers IT infrastructure services in the form of web services - now commonly known as cloud computing.
Amazon Web Services (AWS) Virtual Private Cloud (VPC), Subnet and Virtual Private Networking (VPN)	A virtual private cloud (VPC) is a virtual network dedicated to the STONESOUP AWS account. It is logically isolated from other virtual networks in the AWS cloud. You launch AWS resources, such as Amazon EC2 instances, into your VPC. When the test team creates VPC for the T&E environment and performer work areas, a set of IP addresses will be specified for the VPCs in the form of a Classless Inter-Domain Routing (CIDR) block (for example, 10.0.0.0/16). This is shown in Figure 8. Currently it is anticipated the IP addresses will be allocated as listed below with VPNs to redirect participants to their work are using a Virtual Private Network address accessible over the Internet.
AV-1	Overview and Summary - Describes a Project's Visions, Goals, Objectives, Plans, Activities, Events, Conditions, Measures, Effects (Outcomes), and produced objects
AV-2	Integrated Dictionary - An architectural data repository with definitions of all terms used throughout the architectural data and presentations
Base Programs	A collection of programs in Phase 3 that should approach 500K source lines of code and that are injected with faults for representative tests of the performer technologies.
Broker	A Test Broker is an automated system that accepts jobs to be performed on the performer test systems and publishes them to the appropriate queue for a performer host system to request from the Test Harness. These jobs are queued based on priorities assigned by the Test Orchestrator.

Term	Definition
Communications Application Programming Interface (API)	The Communications application-programming interface (API) specifies how TEXAS components and the performer technology should interact with each other. The API comes in the form of a library with specifications for routines, data structures, object classes, and variables. In some other cases, notably for SOAP and REST services, the API comes as just a specification of remote calls exposed to the API consumers.
Dashboard	Displays testing progress and monitoring of test cases being processed through the architecture
Data/File Storage	To store the data files generated by the performer and base programs that can include database program created files of information, source files and generated executable files, system event data, internal program trace data that can be in the form of text or binary files.
External Dependency	Any external dependencies, or software component or system accessible from a Test Host that is required for the execution of the Performer Technology but is external to the Performer Technology and thus is not part of the Test and Evaluation.
I/O Pairs	I/O Pairs are the input files and descriptions of expected output files. Input files can have benign or malicious commands embedded in them and are designed to provide a working problem to the base program with different program command line switches in them to exercise different portions of the program and potentially the different exploits embedded in the base program. As the performer technologies correct and fix, or miss, the exploits and execute the base programs the program and system output are captured and evaluated against the I/O pair metadata to determine the performer technology score.
Metadata	Metadata or "data about data is used to hold all the information about a test case and how it's to be executed.
OV-1	High-Level Operational Concept Graphic - High-level graphical/textual description of the operational concept
OV-4	Organizational Relationships Chart - Organizational context, role or other relationships among organizations
OV-6c	Event-Trace Description - One of three models used to describe activity (operational activity). It traces actions in a scenario or sequence of events
Performer Subnetwork	Each performer will have a dedicated subnetwork (10.0.3.0 VIBRANCE, 10.0.4.0 PEASOUP and 10.0.5.0 MINESTRONE) of virtual machine hosts running the TEXAS client and their performer technology. The subnetwork will also include any external dependencies needed by the performer's technology or for the execution of individual tests, e.g., a SQL database, DNS server, Web server, or IRC Chat Server. Each host workstation in this subnetwork running a performer's technology will be configured as either an analysis or execution host for that specific performer technology, with memory and computing resources appropriate to that role.
Performer Technologies	Technologies advanced by the STONESOUP performers to test specific areas of technology.
Provisioner	Cloud Platform-as-a-Service (PAAS) services enable flexible lease of computing resources on demand with very little lead-time, including workstations and servers with specified memory and processing power. This flexibility is managed by a Provisioner who adds new workstations, servers, or entire subnetworks to the test infrastructure as needed, configuring each new platform from a predefined virtual machine template for the role it will play in the overall architecture.

Term	Definition
Rehearsal of Concept (ROC) Drill	A ROC drill is an exercise where the test and performer teams of STONESOUP work together to discuss and synchronize their roles in the Test and Evaluation event. The ROC drill is intended as an opportunity for the team to step through test execution and understand the sequencing as well as to discuss shared challenges and opportunities ahead. The end result is to clarify roles, expectations and elicit feedback on implementation plans and resources required for the testing.
Scoring Database Queries	The Scoring/Analysis Component is an automated system that scores test results and computes statistics across the database of completed tests. The Analysis function continuously updates statistics that show the distribution of different characteristics of the completed tests. These statistics are used by the Test Orchestrator to prioritize test case execution. Although each test case is scored by a default algorithm immediately on test completion, the Scoring function is used to rescore tests based on alternate algorithms or in the presence of filters defined by the Test Administrator based on review of anomalous results - all generated data and snapshot (such as environment information, overrides, filters) are stored in the repository.
Stage 1	Stage 1 of testing is where the test case's source code is compiled without performer technology and the I/O pairs are also executed without performer technology to ensure the validity of the exploit. Stage 1 is designed to provide a baseline for comparison to the performer technologies.
Stage 2	Stage 2 of testing is where the test cases are analyzed and modified with the respective performer hardening technologies. Stage 2 runs completely separate from Stage 1 however comparison to the Stage 1 results is required to determine the relative performance and impact of the performer's technology.
Subnetwork	A smaller segment of network carved out of the network allocation.
SV-1	Systems Interface Description - Identification of systems, system items, and their interconnections
SV-6	Systems Resource Flow Matrix - Details of system resource flow elements being exchanged between systems and the attributes of that exchange
Test & Evaluation Execution and Analysis System (TEXAS)	TEXAS is the term used for the entire STONESOUP test system including the cloud computing infrastructure, testing system components (e.g. Test Harness, Test Host Template, etc.) and the Test Case(s) Corpus. Used together they test the performer technologies.
Test Administrator	The Test Administrator is a user/system component member of the T&E team responsible for conducting the testing process according to the test plan, tuning and exercising engineering oversight of the test infrastructure as needed to successfully complete the testing. In practice the Test Administrator will choose one or more test cases to run against a performer and those test cases will be added to the appropriate job queues. System interface needs to be primarily pause and resume with automatic scoring and selection of next set of test cases to be run. Other control may be the size of the queue and number of available virtual machines to speed up or slow down adaption to test case results which the orchestrator attempts to maintain a flat distribution of completed tests across performers, weaknesses and other criteria. The Test Administrator keeps track of Jobs and specifically the jobs in analysis those in execution by tracking how big the queues are against goals and use of external resources. The provision function is adjusted manually via configuration scripts and a graphical user interface. The Test Administrator must configure resources (how many instances, how many CPUs, how long, whether it is a re-run), must assess costs and ensure resources stay within those costs, and must initiate the Test Host so it can start running Test Cases.

Term	Definition
Test Creator	User/system component that creates Test Cases and provides them to the T&E Framework. TEXAS uses a process developed during Phases 1 and 2 and carried on into Phase 3. For Phase 3, all methods were evaluated for effectiveness in collecting data and noninterference with the performer technologies.
Test Framework	Software component residing on the Test Host with Performer technology to run Test Cases and to publish results. The Test Framework must be available for and compatible with a variety of Linux distributions.
Test Harness	The software that interacts with other test functions to install and configure a test case, invoke the performer's technology, collect the test results, and send the results data to storage. The Test Harness function is implemented by a software application installed in each performer test host. The Test Harness is responsible for requesting tasks from a Test Broker when it is idle.
Test Host	Physical or virtual system that contains the Linux distribution, modified with the requisite test framework components and the performer technology. To accommodate the three performers, Test Hosts are modified to meet their specific operating system requirements, technology and supporting artifacts.
Test Host Template	A snapshot of a machine instance that has both a Performer Technology and a Test Framework installed that can be quickly provisioned.
Test Manager	The Test Manager is a software component that allows the Test Administrator to orchestrate the running of Test Cases through a Dashboard that displays testing progress and real-time monitoring of the running test cases. The Test Manager should be able to run one test, run all tests, or run a set of test (where the set is chosen by search criteria). The Test Administrator should also be able to view metadata and Results in the Dashboard.
Test Orchestrator	The Test Orchestrator is an automated rules engine that extracts Tasks from Jobs provided by the Test Manager according to a standard format (Analyze, Execute, Score). The Orchestrator attempts to maintain a flat distribution of completed tests across performers, weaknesses and other criteria.
Tracing	Tracing enables the Scoring Database Queries to collect accurate data during scoring and determination of where and when faults occur. Tracing instrumentation will be inserted with the Fault injection process during compiler translation.