

1 Cover Sheet and Transmittal Letter

A. Cover Sheet

(1) BAA Number	IARPA-BAA-16-12
(2) Technical Area	N/A
(3) Lead Organization Submitting Proposal	Star Lab Corporation 1221 Connecticut Ave NW Washington D.C. 20036
(4) Type of Business, Selected Among the Following Categories: "Large Business", "Small Disadvantaged Business", "Other Small Business", "HBCU", "MI", "Other Educational", or "Other Nonprofit"	Other Small Business
(5) Contractor's Reference Number (if any)	N/A
(6) Other Team Members (if applicable) and Type of Business for Each	Raytheon BBN Technologies, Corp. 1300 North 17th Street, #400 Arlington, VA 22209 Large Business
(7) Proposal Title	Galahad - A Secure User Computing Environment for the Cloud
(8) Technical Point of Contact to Include: Title, First Name, Last Name, Street Address, City, State, Zip Code, Telephone, Fax (if available), Electronic Mail (if available)	Mr. Adam Fraser 11467 Huebner Road, Suite 150 San Antonio, TX 78230 Phone: 210-542-0777 Email: adam@starlab.io
(9) Administrative Point of Contact to Include: Title, First Name, Last Name, Street Address, City, State, Zip Code, Telephone, Fax (if available), Electronic Mail (if available)	Mr. Don Sanders 125 North Side Square, Suite 300 Huntsville, AL 35801 Phone: 256-270-4301 Email: don@starlab.io
(10) Volume 1 no more than the specified page limit	Yes
(11) Restrictions on Intellectual property rights details provided in APPENDIX G format?	Yes
(12) OCI Waiver Determination, Notification or Certification [see Section 3.A.1] Included?	Yes
(12a) If No, is written certification included?	N/A
(13) Are one or more U.S. Academic Institutions part of your team?	No
(13a) If Yes, are you including an Academic Institution Acknowledgement Statement with	N/A

your proposal for each U.S. Academic Organization that is part of your team (Appendix A)?	
(14) Total Funds Requested from IARPA and the Amount of Cost Share (if any)	\$2,649,609 No cost share
(15) Date Proposal as Submitted.	180 days from 23 June 2017

B. Official Transmittal Letter



Star Lab Corp.
1221 Connecticut Ave NW
Suite 4A
Washington, DC 20036

20 June 2017

Mr. Kerry Long
Office of the Director of National Intelligence
Intelligence Advanced Research Projects Activity
Washington, DC 20511

Subject: Star Lab's Proposal in entitled "Galahad – A Secure User Computing Environment for the Cloud"
Reference: (1) IARPA Broad Area Announcement IARPA-BAA-16-12 entitled "Virtuous User Environment (VirtUE) Phase 1 dated October 18, 2016

Dear Mr. Long:

Star Lab Corporation (henceforth, Star Lab) is pleased to submit this **fully compliant proposal** in support of IARPA-BAA-16-12 entitled "Virtuous User Environment (VirtUE) Phase 1". Star Lab understands the requirements for this program and is fully qualified in terms of collective experience to recognize, understand, and service the requirements of this program. We propose this effort on a **Cost Plus Fixed Fee (CPFF)** basis. The labor rates for all of our engineering and project management labor categories are included in the attached Volume 2, Cost Proposal. This cost proposal is valid **for one hundred and eighty (180) days**.

For pricing purposes, we have assumed a 1 August 2017 start date. We have proposed a period of performance of 18 months.

Star Lab's **fiscal year**, used for the purposes of financial reporting and policy setting, corresponds with the typical calendar year of January 1 to December 31. Our Accounting Month begins on the 1st day of the month and ends on the last day of the month. Our pay period is monthly based on the 15th day of the month.

Our standard work year is based on a **1920 labor hours** per year. Our **financial system** is not required to be CAS compliant, and our provisional rates were last submitted to DCAA in October 2015 for review. Our accounting system has been reviewed by DCAA and the opinion states **"Star Lab's accounting system is adequate for accumulating and billing costs under Government contracts."** A copy of the letter can be sent to you if you so require.

Star Lab is a small non-traditional defense contractor organized in Delaware, with primary offices in Washington, DC, Huntsville, AL, and San Antonio, TX. The company's basic organizational information includes:


DUNS Number	079360917
CAGE Code	743R1
Tax Id#	46-5366379
NAICS Codes	541511, 541512, 541712

Star Lab represents that we are not a **"foreign corporation"**, or representative of any foreign interest (RFI), and that if there is any change in this status, we will notify the government immediately. **We do not intend to use foreign nationals in support of this program.**

The respective technical and administrative points of contact for this effort are Adam Fraser (adam@starlab.io) and Don Sanders (don@starlab.io), respectfully. Adam can be reached via telephone at 210.542.0777 and Don can be reached at 256.527.4511. I am the only individual authorized to obligate the company on all contractual matters.

We look forward to working closely with the government this program as well as and other programs and initiatives. If you should have any questions or concerns, please contact the respective contacts.

Kind Regards,

A handwritten signature in cursive script, appearing to read "Irby Thompson", written in dark ink on a light background.

Irby Thompson
President, Star Lab Corp.

Enclosures:

- (1) Volume 1, Technical Proposal
- (2) Volume 2, Cost Proposal
- (3) Cost Spreadsheet

Contents

1	Cover Sheet and Transmittal Letter	1
	A. Cover Sheet	1
	B. Official Transmittal Letter	3
2	Summary of Proposal.....	1
	A. Technical Overview of Proposed Research and Plan	1
	B. Summary of Products, Transferable Technology, and Deliverables	5
	C. Schedule and Milestones	5
	D. Related Research	6
	E. Project Contributors.....	8
	F. Technical Resource Summary	8
3	Detailed Proposal Information.....	11
	A. Statement of Work (SOW).....	11
	B. Detailed Description of Research.....	14
	B.1. R&D of Galahad's Virtue Construct (Task 1).....	14
	B.2. R&D of Galahad's Canvas Interface (Task 2)	18
	B.3. R&D of Galahad's Lifecycle Management Construct (Task 3)	21
	B.4. R&D of Galahad's Sensing and Logging Capabilities (Task 4)	23
	B.5. Performance Considerations (Requirement 10)	25
	B.6. Program Execution and Risk Management.....	26
	C. State-of-the-Art.....	27
	D. Data Sources	29
	E. Deliverables.....	29
	F. Cost, Schedule, Milestones	29
	G. Offeror's Previous Accomplishments	31
	H. Facilities	33
	I. Detailed Management Plan	33
	J. Resource Share	35
	K. Other Supporters.....	35
4	Attachments.....	36
	Attachment 1: Academic Institution Acknowledgement Letter.....	36
	Attachment 2: Restrictions on Intellectual Property	37
	Attachment 3: OCI Waiver/Determination/Notification or Certification.....	38
	Attachment 4: Bibliography.....	39
	Attachment 5: Relevant Papers	42

Attachment 6: Consultant Commitment Letters	Error! Bookmark not defined.
Attachment 7: Summary Charts	43

Figures

Figure 1: Galahad Virtue Construct.....	2
Figure 2: Galahad CONOPS	3
Figure 3: Concept of Operations for Cloud Snooping	6
Figure 4: Organizational Chart	8
Figure 5: Galahad’s Canvas Architecture.....	19
Figure 6: Research and Development Schedule.....	30

Tables

Table 1: Requirements Mapping to Galahad Capabilities	3
Table 2. Technical and Reporting Deliverables.....	5
Table 3. Work Breakdown Structure, Schedule, and Milestones	5
Table 4. Level of Effort (hours) for Star Lab.....	9
Table 5. Level of Effort (Hours) for BBN.....	9
Table 6. Travel by Affiliation	10
Table 7. Statement of Work for the Development of Galahad	11
Table 8. Galahad Sensor Capabilities	24
Table 9: Risks and Mitigation Strategies	27
Table 10. Technical and Reporting Deliverables	29
Table 11: Cost Summary by Task and Subtask (No Fee).....	29
Table 12. Milestones and Dates for Phase I.....	30
Table 13. Profiles of Key Personnel	35
Table 14. Commercial Items	37

2 Summary of Proposal

A. Technical Overview of Proposed Research and Plan

Star Lab, with its partner Raytheon BBN Technologies (BBN), proposes to conduct an applied research and development project to create and transition Galahad, a revolutionary user computer environment (UCE) for the Amazon Cloud that is designed to be highly interactive while mitigating legacy and cloud-specific threats. Galahad meets all 11 requirements called out in IARPA-BAA-16-12, addressing the broad areas of threat mitigation, sensing and logging, lifecycle management and control, user interaction, and cost and performance.

Unique to the VirtUE program is the requirement that a defensible interactive UCE be built on top of a ***completely untrusted computing foundation***, namely Amazon Web Services (AWS). Secure systems are typically designed from the bottom-up to attest hardware and software using trust-establishing components such as trusted platform modules (TPMs), Intel's Trusted eXecution Technology (TXT), and Intel's Safer Mode eXtensions (SMX). These components are not accessible in AWS, nor does Amazon make available verifiable trust measurements, making it impossible for users to detect several attacks, e.g., hypervisor rootkits, that enable an adversary to affect a target, e.g., read / modify memory, disengage sensors, and filter logs. Worse still, no meaningful progress has been made to improve the security of interactive UCEs currently offered by cloud service providers (CSPs). As a result, adversaries can leverage the arsenal of existing capabilities used to exploit traditional desktops and servers to target users, applications, and virtual desktops within public clouds.

Star Lab's Galahad takes a holistic approach to creating a secure, interactive UCE. Galahad leverages role-based isolation, attack surface minimization practices, operating system (OS) and application hardening techniques, real-time sensing, and maneuver / deception approaches to reduce the risk associated with cloud deployments. Galahad makes no attempt to establish trust, nor does it require specialized, more costly services provided by AWS, e.g., dedicated servers. Instead, Galahad impedes the ability of adversaries to operate within the AWS by making it more difficult to co-locate (either through the use of insiders, compromised hypervisors, witting or unwitting peers, or remote access) with targets, while also requiring adversaries consume more resources. Such an increase in complexity and cost means Galahad also increases the accuracy, rate, and speed with which threats are detected.

Galahad facilitates the construction, deployment, management, and interfacing of role-based virtues. Star Lab's virtues, shown in Figure 1, consist primarily of two components: Valor, a nested hypervisor and Unity, a virtual machine running a small, hardened, de-privileged Linux operating system (OS). Valor includes protections, sensors, and virtual machine (VM) image authentication, but most importantly it facilitates the regular, recurring live migration of Unity VMs. This moving target defense approach deters hypervisor and peer VM threats within AWS,

an operating environment where trust can never be completely established, by limiting the time an adversary has to access targeted virtues while also increasing the number of access points required by the adversary to be successful.

Unity includes additional protections to address external and insider threats while also incorporating a compatibility layer for supporting legacy Windows applications and adjustable sensors. Limiting Unity to Linux (versus using Linux and Windows) allows Star Lab to more greatly reduce the overall attack surface of the virtue, improve performance, and streamline development. Additionally, executing Windows applications on Linux increases the uncertainty, complexity, and cost to the adversary.

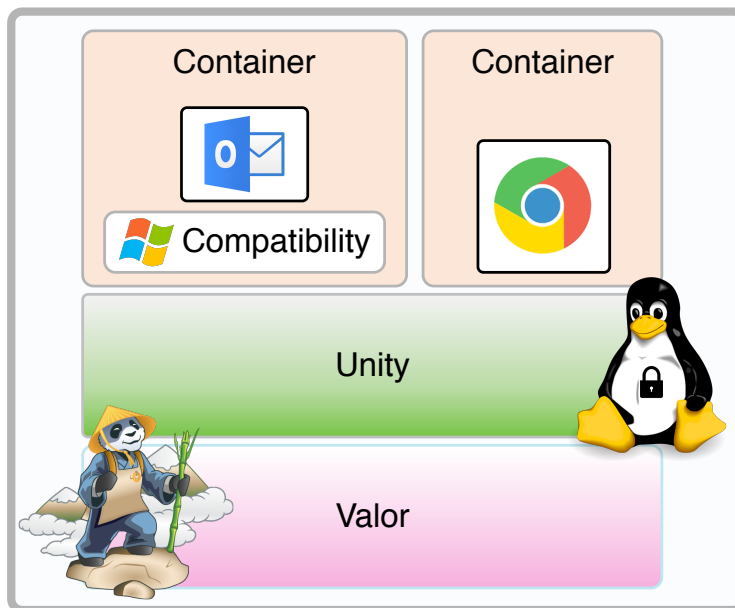


Figure 1: Galahad Virtue Construct

Sensing and logging take place throughout the Galahad software stack, without requiring modifications to the applications under protection. Within Unity, these introspection points consist of kernel instrumentation, activity monitors, and sensing points added to the compatibility layer. Additional sensing and logging capabilities are layered across Unity and Valor creating tripwires to detect adversary tampering. Overall, a virtue's protections and sensing capabilities are adjustable, updatable, and collaborative, forcing adversaries to navigate a well-covered attack surface to avoid detection. Furthermore, the use of role-based virtues to reduce behavior complexity boosts out-of-band analytics making Galahad a highly defensible construct.

As Figure 2 shows, users access virtues stored in AWS through Galahad's Canvas, a presentation interface accessible from a user's thin client. Canvas leverages Galahad's Virtue Control Application Programming Interface (API) to authenticate and launch virtues and Galahad's Virtue-Canvas API to enable users to perform normal work activities, invoke applications, initiate workloads, and perform virtue-to-virtue sharing. Behind the scenes, virtues are regularly migrated to other hosts within AWS using the Virtue Control API. Users experience minimal impact during migrations. Finally, during all of this activity, sensors are logging the activity of all virtues (system and user behaviors) and smartly storing the information for later retrieval through Galahad's Sensing and Logging Control API.

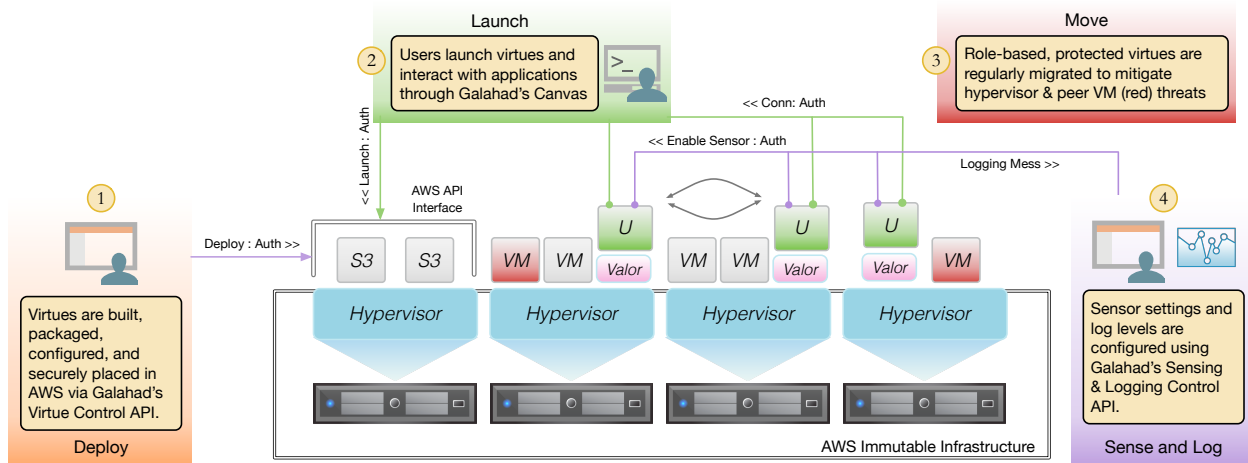


Figure 2: Galahad CONOPS

Delivering Galahad requires the following key innovative technologies:

- A small lightweight nested hypervisor that enables the live migration of virtues inside AWS without causing significant user interruptions;
- A customizable presentation interface that facilitates secure interactions with AWS-based virtues and enables necessary resource sharing;
- A virtue control layer to securely interface with AWS storage, manage the lifecycle of virtues, and inform decisions influencing performance and cost;
- Administrative tools that enable system situational awareness and the easy and rapid creation and movement of virtues; and
- Dynamic configurable sensors and logging mechanism that spans the entire virtue software stack while also balancing inspection granularity with storage and transmission constraints.

With the above innovations, and other practical engineering solutions, Galahad will meet or exceed all performance metrics and address each of the 11 requirements. Table 1 provides a more succinct mapping of requirements to Galahad capabilities.

Table 1: Requirements Mapping to Galahad Capabilities

Requirement	Galahad Capability	Proposal Reference
1 Interfacing with virtues within AWS	Galahad's Virtue Control API enables the lifecycle management of virtues, to include their secure deployment into AWS.	3.B.3.2
2 Immutable minimized virtues	Galahad's Virtue Assembler automates the creation of immutable, minimized virtues.	3.B.1.1
3 Role-based virtues	All applications required for a role are installed within a single Unity instance, which in turn executes within a single instance of Valor. Galahad maintains a 1:1 Unity to Valor relationship.	3.B.1

4 Defensible virtualized construct	Galahad's Unity and Valor are built with only required functionality, configured to reduce the attack surface, and regularly migrated, addressing external, internal, peer, and infrastructure threats.	3.B.1.3
5 Virtue logging and adjustability	Galahad's sensors span the entire software stack and Galahad's Log Digester filters logs based on configurations and tasking received through Galahad's Sensing and Logging Control API.	3.B.4
6 Dynamic virtue protections	Dynamic protections include live migration, memory sanitization, and a stackable security policy model. Protections are adjusted through the Virtue Control API.	3.B.1.2
7 Legacy application support	Unity includes a compatibility layer specifically designed to support legacy Windows applications.	3.B.1.4
8 Information exchange	Galahad's Virtue-Canvas API facilitates the secure exchange of presentation information and enables resource sharing.	3.B.2.2
9 Virtue presentation interface	Galahad's Canvas is a lightweight application or webpage that presents the components necessary to invoke and interact with applications running inside virtues.	3.B.2.1
10 Virtue performance	Virtues have minimal inherent "must-have" AWS requirements. Star Lab will use an AWS integrated test environment to better refine instance configurations that effect cost and performance.	3.B.5
11 Virtue management	Inspired by Docker, Galahad's Virtue Assembler enables the easy transportation and extension of existing virtues as well as the attestation of virtue creators.	3.B.3.1

Star Lab's approach to researching and developing Galahad will prioritize an iterative and adaptive agile development methodology. Such a methodology will include Test Driven Development, continuous integration, and regular communication with stakeholders. It will also ensure Galahad is delivered within the demanding 18-month period of performance. Star Lab will also leverage open source technologies where practical to bootstrap the creation of an initial prototype. Furthermore, Star Lab will extend its research and development infrastructure into AWS EC2 to enable researchers to regularly experiment, test, and evaluate Galahad and its various components against the program's performance metrics and against ground-truth operational conditions.

Finally, Galahad's design is based on a thorough understanding of virtualized environments, AWS, and secure system design. Star Lab and BBN personnel are both veteran research and development performers. Star Lab's flagship product Crucible is a security hypervisor for Intel & ARM platforms that addresses concerns unique to mission-critical computing, namely: secure boot, runtime hardware & software control, execution isolation, and technology protection. Star Lab is also a regular contributor to the Xen Project. BBN's Cyber Security department has been performing security-focused research and development for DoD and Intelligence Community customers for more than 15 years. BBN is currently a performer on the DARPA Transparent Computing program where they are developing technologies to collect security-relevant information across all layers of software abstraction to support proactive enforcement of desirable policies, near-real-time intrusion detection, and forensic analysis. Together, this team brings both the research and development expertise and practical engineering experience to realize Galahad.

B. Summary of Products, Transferable Technology, and Deliverables

Table 2 summarizes the reporting and technical deliverables, and the frequency of delivery.

Table 2. Technical and Reporting Deliverables

Item	Month
Slide presentations (kickoff, on-site visits, & Phase II Proposers Day)	1, 4, 11, 12
Phase 2 slide presentation	11
Test plans	9, 18
Source code, prototypes, and documentation	6, 12, 18
Conference paper	16
Monthly reports	Monthly
Cost reports	Monthly
Final report	Conclusion of Phase I,

C. Schedule and Milestones

Table 3 provides a work break down structure (WBS) of tasks for the effort proposed as well as the start and duration for each task and milestone schedule. A Gantt chart and a detailed milestone table are provided in Section 3.F. Milestones will be tied to improving security, functional, and performance measures

Table 3. Work Breakdown Structure, Schedule, and Milestones

Task #	Task Description	Start Month	Duration	Milestone Months / Description
1	Virtue Construct R&D	1	11 Months	
1.1	Design & Develop Threat Mitigation Enablers	1	7 Months	4, 5, 7, 14 increasing # of protections
1.2	Integrate Application Support Enablers	8	4 Months	4, 7, 10 increasing # of Linux apps then Windows apps
2	Canvas R&D	5	9 Months	
2.1	Design & Develop Canvas Interface	5	5 Months	6, 7, 11, 16 increase # of virtues connected to Canvas
2.2	Design & Develop Resource Sharing	12	4 Months	14 demonstrate resource sharing use cases
3	Virtue Lifecycle Management R&D	1	12 Months	
3.1	Design & Develop Virtue Building and Packing Tools	1	8 Months	2, 3, 6, 14 increase maturity of virtue packaging and deployment
3.2	Design & Develop Virtue Control	5	4 Months	10 demonstrate virtue launch and control w/Canvas
4	Sensor and Logging R&D	1	17 Months	
4.1	Design & Develop Sensor Suite	1	9 Months	3, 6, 10 increase # of logging capabilities
4.2	Design & Develop Sensor Control	10	8 Months	14 demonstrate dynamic control

5	System Testing and Experimentation	1	18 Months	8, 17 Focus on evaluation success
6	Technology Demonstration and Evaluation	8	4 Months	9, 18 Support evaluations
7	Reporting and Management	1	18 Months	Monthly

D. Related Research

Securing cloud computing user environments requires the formulation and verification of various threat models. Threats to CSPs range from traditional remote exploitation and DDoS attacks to the targeted exploitation of the virtualization and management technologies underpinning today's cloud services. The most publicized attacks in the latter area include hypervisor specific vulnerabilities that enable arbitrary code execution [1] and guest escape [2,3]. Receiving considerably less attention (based solely on the lack of published reports) are cloud attacks (reference Figure 3) wherein a malicious actor attempts to spy on a target system by (1) co-locating a VM and (2) covertly exploiting shared resources to collect sensitive information. Public cloud providers have taken steps to prevent many co-location activities [4,5]; however, recent research suggests not all the countermeasures have been effective [5].

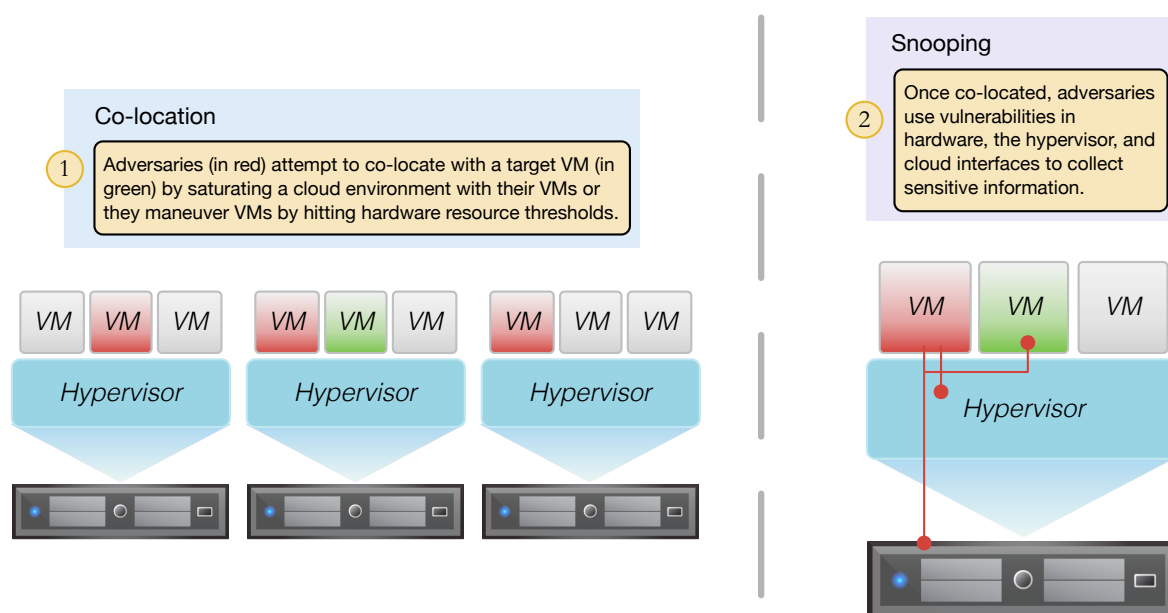


Figure 3: Concept of Operations for Cloud Snooping

In an effort to defeat these threats, as well as others, researchers have attempted to develop new constructs to introduce security into virtualized and cloud environments. Attack surface minimization, i.e., activities to reduce interfaces and resources available to an attacker to exploit, is one approach overlapping with the VirtUE program. One example of attack surface minimization are unikernels [6,7]. Unikernels are an emerging innovation for developing and

deploying secure light-weight application stacks. Today applications and services like web servers and databases are deployed to VMs to execute as user-space processes on top of full operating systems. While these applications and services are now being containerized, they still use monolithic operating systems where numerous *unused* libraries and services are available to adversaries. Unikernels allow for the restructuring of application and kernel code to create optimized singular runtime environments. As a result, the attack surface, i.e., the unnecessary functionality, is no longer available to adversaries. Also, by pairing unikernels with virtualization technologies, multiple unikernel applications can operate in isolation so that the potential compromise of one application unikernel does not affect the operation of other application unikernels. Star Lab has extensive experience with Unikernels and will investigate their use for special-purpose virtues.

Another approach to countering cloud-based threats is to detect compromised hypervisors. In traditional environments where access to hardware is possible, the best approach is to use a TPM. As previously mentioned, however, access to and multiplexing of the TPM is not provided by CSPs. This lack of chip-level access also makes it infeasible to use out-of-band execution environments, like System Management Mode (SMM), to repeatedly measure the integrity of hypervisors [8]. In contrast to these techniques, some researchers have attempted to address this problem by prototyping solutions CSPs could deploy, such as using a publicly visible central verifier [9]. Other solutions use environmental key generation to create keys for encrypting / decrypting VMs [10] or they use hash chains [11] to create a non-reputable log of hypervisor activity that can be analyzed for malicious activity. The latter approach still requires a root-of-trust.

A third approach to cloud security is to leverage VM introspection. Typically, VM introspection leverages hypervisor libraries, e.g., VM event and `altp2m` in Xen, to trap on memory access within a guest VM. Since, however, the hypervisor is normally inaccessible, researchers and industry have developed and deployed nested hypervisors. Nested hypervisors are simply hypervisors that execute inside of a virtual machine. Using nested hypervisors enables the use of introspection libraries, but it can facilitate other protections such as encrypting / decrypting memory pages [12].

Nested hypervisors are particularly relevant to one of Star Lab's proposed innovations, namely its implementation and use of a moving target defense approach. AWS EC2, by default, does not allow for the live migration of virtual machines, i.e., moving a running VM to a different physical machine without halting it. Thus, an additional layer, a nested hypervisor, is required to make live migration possible. Fortunately, researchers have already demonstrated this is possible using a nested hypervisor, XenBlanket [13]. More importantly, researchers have demonstrated it is possible to use XenBlanket **inside AWS**; they implemented live migration for the purpose of using Spot Instances to obtain cost savings for an always-on, Internet-accessible service [14].

Finally, several products exist that use virtualization to enable more secure end-point or desktop computing. Solutions like Bromium [15] and Qubes OS [17] leverage virtualization to establish containerized and isolated execution environments, but they are not designed to be deployed inside AWS. Still, many of these products have solved similar problems, and, Qubes OS in particular, will be leveraged to bootstrap the development of several of Galahad's capabilities such as Galahad's canvas and resource sharing.

E. Project Contributors

Figure 4 displays Star Lab's organizational chart. It includes all anticipated project participants, functional roles, and assigned tasks.

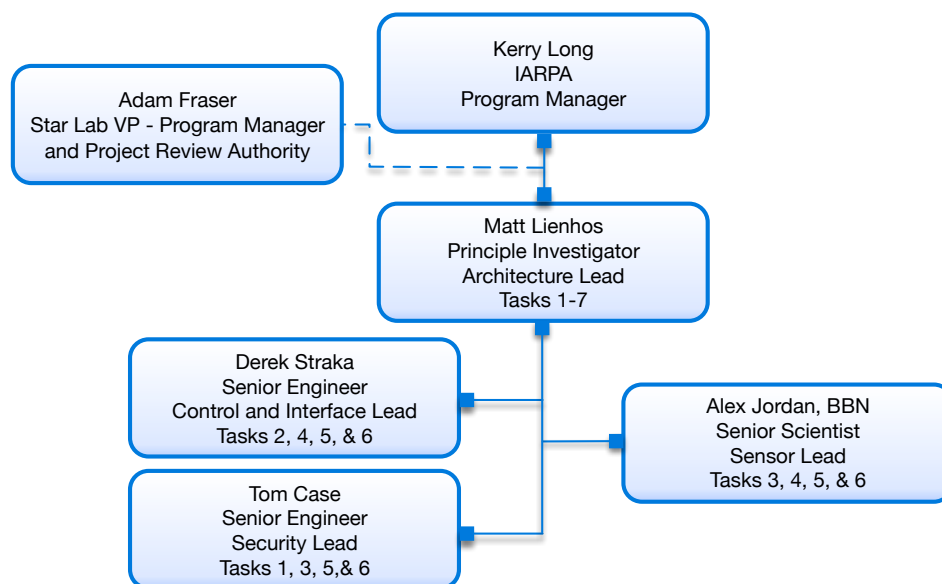


Figure 4: Organizational Chart

F. Technical Resource Summary

Table 4 and Table 5 summarize the total level of effort by labor category, technical discipline, and affiliation by task. Key personnel are identified by name. These individuals will be assisted by other individuals within the same labor category.

Table 4. Level of Effort (hours) for Star Lab

Task #	Technical Specialist 5-3	Technical Specialist 5-4	Technical Specialist 6-2	Total
1	2,120	660		2,780
1.1	1,360	540		1,900
1.2	760	120		880
2	860	800		1,660
2.1	220	480		700
2.2	640	320		960
3	320	1,420		1,740
3.1	160	600		760
3.2	160	820		980
5	200	720		920
6	200	300		500
7			432	432

Table 5. Level of Effort (Hours) for BBN

Task #	Staff Sci II Alex Jordan	Sr Scientist	Lead Scientist	Admin	Sr Fin Analyst	Mgr III Fin Analyst	Mgr III Prog Mgr	Prin Scientist	Total
3	370	24	6						400
3.1									
3.2	370	24	6						400
4	2,184	817	130						3131
4.1	1,062	440	66						1568
4.2	1,122	377	64						1563
5	120	40							160
6	160	88	16						423
7				36	144	9	216	18	432

To perform this project, Star Lab will leverage three open source commercial items: Wine, CentOS, and XenBlanket. Star Lab will also leverage source code it has created and upstreamed (released) to the Xen project. No proprietary software will be used to develop Galahad. Also, Star Lab will not require any materials or equipment (such as IT). A monthly expense of \$500 a month is anticipated to extend the research and development infrastructure into AWS EC2. Estimated travel for Star Lab and BBN are included in Table 6.

Table 6. Travel by Affiliation

Affiliation	Purpose	Number of Personnel	Estimated Cost (Unburdened)
Star Lab	Kickoff Workshop	3	\$3,716.00
Star Lab	Site Visit	1	\$1,187.00
Star Lab	Midterm Waypoint Exam	3	\$4,663.50
Star Lab	Phase 2 Proposer's Day Workshop	2	\$1,889.00
Star Lab	Optional Site Visit	1	\$962.00
Star Lab	Final Waypoint Exam	3	\$4,261.50
BBN	Kickoff Workshop	2	\$857.50
BBN	Team Visit	2	\$2038.61
BBN	Site Visit	2	\$2038.61
BBN	Team Visit	2	\$2038.61
BBN	Phase 2 Proposer's Day Workshop	2	\$857.50
BBN	Optional Site Visit – Star Lab	2	\$2038.61
BBN	Team Visit	2	\$2038.61
BBN	Optional Site Visit – Star Lab	2	\$2038.61
BBN	Optional Site Visit JHU	2	\$164.23
BBN	Midterm Waypoint Exam	2	\$1135.85
BBN	Optional Site Visit JHU	2	\$82.12
BBN	Final Waypoint Exam	2	\$1135.85

3 Detailed Proposal Information

A. Statement of Work (SOW)

Table 7 defines the technical tasks and sub-tasks to be performed during this project.

Table 7. Statement of Work for the Development of Galahad

Task 1: Virtue Construct Research and Development	
Description	The contractor shall research and develop a virtue, a more secure, capable sensor and user computing environment for the cloud.
Approach	The contractor will conduct research and development in two specific areas (outlined below).
Responsible Organization	Star Lab.
Exit Criteria	Secure virtues are delivered.
Deliverables	Source code, prototypes, and documentation
Task 1.1: Design and Develop Threat Mitigation Enablers	
Description	The contractor shall research and develop a methods and techniques for reducing the complexity of virtues and protecting virtues from external, insider, peer, and infrastructure threats.
Approach	The contractor shall leverage its extensive experience building secure, minimized Linux instances to create virtues comprised of several protective elements and applications.
Responsible Organization	Star Lab.
Exit Criteria	Virtues are protected from external, insider, peer, and infrastructure attacks.
Deliverables	Source code, prototypes, and documentation.
Task 1.2: Integrate Application Support Enablers	
Description	The contractor shall research and develop a mechanism for users to invoke and interact with Linux and legacy Windows applications
Approach	The contractor shall leverage Wine to support legacy Windows applications.
Responsible Organization	Star Lab.
Exit Criteria	Virtues are able to execute Linux and Windows legacy apps.
Deliverables	Source code, prototypes, and documentation.
Task 2: Canvas Research and Development	
Description	The contractor shall research and develop a mechanism for users to interface with virtues and invoke and interact with applications.
Approach	The contractor will conduct research and development in two specific areas (outlined below).
Responsible Organization	Star Lab.
Exit Criteria	A canvas supportive of multiple virtues is delivered.
Deliverables	Source code, prototypes, and documentation.
Task 2.1: Design and Develop Canvas Interface	
Description	The contractor shall research and develop a presentation interface for users to invoke and interact with applications located in virtues.
Approach	The contractor shall leverage Amazon's virtual private clouds and existing RDP desktop access protocols such as Spice to connect virtues to a web-based presentation interface.
Responsible Organization	Star Lab.
Exit Criteria	A canvas is delivered.
Deliverables	Source code, prototypes, and documentation.

Task 2.2: Design and Develop Resource Sharing	
Description	The contractor shall research and develop a mechanism for enabling virtues to share data hyperlinks, files, etc.
Approach	The contractor shall leverage a model developed by the creators of Qubes OS to provide a user-in-the-loop process for sharing resources.
Responsible Organization	Star Lab.
Exit Criteria	Virtues are able to share hyperlinks, files, etc.
Deliverables	Source code, prototypes, and documentation.
Task 3: Virtue Lifecycle Management Research and Development	
Description	The contractor shall research and develop a mechanism for building, deploying, storing, launching, and transporting virtues.
Approach	The contractor will conduct research and development in two specific areas (outlined below).
Responsible Organization	Star Lab and BBN.
Exit Criteria	A virtue lifecycle management API is delivered.
Deliverables	Source code, prototypes, and documentation.
Task 3.1: Design and Develop Virtue Building and Packaging Tools	
Description	The contractor shall research and develop a means to build, deploy, terminate, query, and migrate virtues.
Approach	The contractor shall use its experience with Docker, Kconfig, hypervisors and operating system configuration to research and develop tools to build and package virtues so they can be deployed to Amazon Web Services (AWS).
Responsible Organization	Star Lab.
Exit Criteria	Virtue building and packaging tools are delivered.
Deliverables	Source code, prototypes, and documentation.
Task 3.2: Design and Develop Virtue Control	
Description	The contractor shall research and develop a means to deploy, launch, terminate, query, and migrate virtues.
Approach	The contractor shall use its experience with the open source nested hypervisor XenBlanket and its experience with AWS to research and develop a control API that will control virtues stored within AWS.
Responsible Organization	Star Lab and BBN.
Exit Criteria	A virtue control API is delivered.
Deliverables	Source code, prototypes, and documentation.
Task 4: Sensor and Logging Research and Development	
Description	The contractor shall research and develop a sensing and logging capabilities that enables developers to produce more secure applications.
Approach	The contractor will conduct research and development in two specific areas (outlined below).
Responsible Organization	Star Lab and BBN.
Exit Criteria	Controllable sensing and logging capabilities are delivered.
Deliverables	Source code, prototypes, and documentation.
Task 4.1: Design and Develop a Sensor Suite	
Description	The contractor shall research and develop a means to collect, log, and store security-relevant data about the operation of virtues.
Approach	The contractor shall use its experience with application instrumentation and low-level system monitoring to research and develop hypervisor, operating system, and application sensing / logging technologies.
Responsible Organization	Star Lab and BBN.

Exit Criteria	Sensing and logging capabilities are delivered.
Deliverables	Source code, prototypes, and documentation.
Task 4.2: Design and Develop Sensor Control	
Description	The contractor shall research and develop a means to dynamically adjusting sensors and logging behaviors through a sensing and logging control mechanism.
Approach	The contractor shall use its experience with decision logic and actuation to research and develop a control API that will adjust the level and amount of sensing and logging.
Responsible Organization	Star Lab and BBN.
Exit Criteria	A sensing and logging control API is delivered.
Deliverables	Source code, prototypes, and documentation.
Task 5: System Testing and Experimentation	
Description	The contractor shall perform system integration and operational testing in preparation for the demonstrations and evaluations.
Approach	The contractor shall integrate Galahad's components, perform end-to-end functional testing, and debug and modify as needed.
Responsible Organization	Star Lab.
Exit Criteria	Delivery of Galahad for demonstrations and evaluations.
Deliverables	Test plans.
Task 6: Technology Demonstration and Evaluation	
Description	The contractor shall demonstrate their solution's ability to advance toward and meet security test criteria, functionality test criteria, and performance test criteria.
Approach	The contractor will travel to the Government-selected test and evaluation facilities at the Johns Hopkins University Applied Physics Lab (APL) outside Baltimore, MD to assess Galahad's performance.
Responsible Organization	Star Lab.
Exit Criteria	Demonstrations and evaluations conclude.
Deliverables	Test plans.
Task 7: Reporting and Management	
Description	The contractor shall ensure the effort meets the government requirements.
Approach	The contractor will use project management best practices in order to track project status, spending, and manage the security aspects of the effort. The contractor will conduct meetings at least on a biweekly interval to assess project status and timeliness.
Responsible Organization	Star Lab.
Exit Criteria	Contract End.
Deliverables	Monthly status and cost reports, any associated whitepapers, the kickoff presentation, and the final report.

B. Detailed Description of Research

The following sections outline the research Star Lab and its partner BBN will complete to deliver an interactive and secure UCE acceptable to government customers. Star Lab and BBN will concentrate its efforts along four strategic paths:

- **R&D of Galahad's virtue construct** – Virtues will be minimized and defensible, addressing four classes of threats: external, internal, peer, and infrastructure;
- **R&D of Galahad's canvas interface** – Through Galahad's canvas, users will invoke and interact with several role-based virtues simultaneously while also being able to cooperate with applications defined within the same virtue as well with applications residing in other virtues;
- **R&D of Galahad's lifecycle management construct** – Administrators will create, configure, transport, and extend virtues while security personnel will configure virtues, adjusting protections and controlling migration behaviors; and
- **R&D of Galahad's sensors and logging capabilities** – Sensing and logging will reach across a virtue and include technologies that enable the logging of a configurable set of attributes.

Following these paths, along with tasks related to integration, testing, and evaluation, will enable Star Lab to satisfy the 11 program requirements and meet functional, security, and performance goals.

B.1. R&D of Galahad's Virtue Construct (Task 1)

Star Lab's proposed virtue construct will allow for the role-based division of computational tasks (Requirement 3). Windows and Linux applications, utilities, scripts, etc. for a given role will be placed in a single virtue's Unity VM, a small, hardened, de-privileged Linux instance. Unity VMs, in turn, will execute inside Valor, a nested hypervisor that implements additional protections, sensors and logging, and a live migration capability. This construct achieves isolation between virtues and hence roles, while also enabling innovative threat mitigation approaches. The following subsections outline Star Lab's approach for designing and developing more defensible virtues--by reducing their complexity and introducing dynamic, adjustable protections-- and to support Windows and Linux applications.

B.1.1. Minimized Virtues (Requirement 2)

Star Lab will make its virtues more defensible by first reducing the amount of available and reachable resources exposed to / by Unity and Valor. Today's operating systems and hypervisors include hundreds of drivers, libraries, services, settings, and files all designed to help them be multi-user and multi-purpose. As a result, these foundational components are necessarily complex and large, but also less defensible. Galahad will flip this paradigm on its head; instead of including as much functionality as possible, Unity and Valor will be minimized,

i.e., base instances of Unity and Valor will include only the functionality needed to initialize within AWS. Any required functionality, realized with the addition of drivers, apps, services, etc., will be added during the virtue build process explained in greater detail in Section B.3.1.

In order to reduce the overall attack surface of the Valor, Star Lab will utilize Xen Kconfig. Star Lab built Xen Kconfig by porting Linux's Kernel Kconfig build configuration machinery (ref. paper in Attachment 5) and it has been upstreamed to the Xen build system. Using Xen Kconfig allows Valor to be built in a more modular fashion and it enables the disabling of unnecessary features that may enable potential attack vectors. For example, the kexec feature adds a special hypercall giving the control domain the ability to warm-boot an alternate hypervisor image or secondary crash kernel [17]. While this feature may be useful for debugging and other well controlled high-availability platforms, it represents a significant risk to deployments not requiring such support.

A minimal Unity will be based on the CentOS Linux distribution. This decision is motivated by the distribution's granular packages and existing package manager. Star Lab will also use a modular Linux Kernel in which every component is built as a loadable module. Furthermore, Star Lab will research techniques for defining Unity configurations which only include the modular kernel components required for a given role. This ensures that Unity VMs only have the OS kernel services which are explicitly required. For example, if a given Unity VM does not require networking support it will not contain any physical or virtual network drivers required to connect with those interfaces. The CentOS package manager, along with the aforementioned Unity configuration definition, will be used to limit the number of services, daemons, etc. running within a given Unity user-space. For additional fine-grained restrictions on user-space to kernel-space syscalls, Star Lab proposes to leverage libseccomp's syscall filtering [18] to limit available syscalls to only those required for each individual process running within the Unity VM. All of these techniques combine to reduce the attack surface of the Unity VM by turning an existing Linux distribution into something much closer to a Library OS.

Finally, for those few virtues that require a single or a small number of applications, Star Lab will investigate unikernels as a possible construct for creating an even more reduced Linux / Wine execution environment. Unikernels are an emerging innovation for developing and deploying secure lightweight software stacks that contain a minimal set of libraries, services, and code. Star Lab has considerable experience (ref. paper in Attachment 5) with the Rump Unikernel [19], specifically developing isolated network stack unikernels.

B.1.2. Virtue Protections (Requirement 6)

It is advantageous to restrict individual components within a single virtue to only the system and data resources which that instance absolutely requires. Star Lab create a stackable Linux Security Module (LSM) capable of co-operating alongside existing LSM based security technologies (such as SELinux). Deploying an LSM will contain the effects of an individual

application/component compromise and de-privilege the root user, such that traditional privilege escalation attacks will not necessarily lead to a compromise of sensitive data.

An LSM will also provide a useful point of introspection within the Linux kernel which can be used by sensors and for mandatory access control enforcement [20]. Points of introspection include: program execution, task interaction, inode/file, socket, key-management, and memory operations.

Valor will also include protection capabilities. Valor will provide control over the memory sanitization of guest VMs. This is important as it gives greater control over both the state of uninitialized memory as well as the ability to generically sanitize the memory after termination. Finally, Valor will be responsible for enforcing Unity base image immutability by verifying the signature of Unity instances during the deployment process.

B.1.3. Active Defenses (Requirement 4)

There are many unique threats specific to the cloud. From within a guest VM, it is impossible to assess the integrity or trustworthiness of the underlying CSP's virtualization stack. It is also challenging for a guest VM to defend itself from the hypervisor on which it runs and other co-resident guests. Recent research has concluded that hypervisor vulnerabilities and side channel attacks represent a serious threat to sensitive information stored or processed within the cloud [5, 21, 22]. While AWS does support dedicated hosts, this does nothing to address underlying issue of side channel attacks and hypervisor vulnerabilities and defeats the cost-saving benefits of the cloud paradigm.

Star Lab proposes to address these attack vectors in-part by introducing the capability to intelligently migrate Unity instances within AWS. At this time, AWS has some significant migration limitations and existing moving target defense strategies rely entirely on the cloud provider implementing support for live migration [23, 24]. Star Lab proposes alleviating these shortcomings by leveraging its nested hypervisor, Valor. Valor will provide live migration support while also increasing the overall flexibility of the proposed approach by not relying on a specific cloud provider. Star Lab will base Valor on XenBlanket. While XenBlanket represents a proven reference platform, the work has not been maintained and will require some updating to bring it up to the latest upstream Xen.

As of version 4.6, the Xen hypervisor has full support for live migration. Xen's live migration leverages dirty page tracking to decrease the amount of time that a VM must be paused between migration events; however, this process requires a secondary VM host to mirror the image for a period of time before the full state is migrated and control is transferred. To improve performance and decrease user interruptions, Star Lab will include with Galahad a migration engine that will interpret usage patterns, e.g., user inactivity and the backgrounding of applications, to decide when to migrate Unity instances. Another approach to limiting the frequency of migration events while reducing exposure is to employ co-residency sensing. By

leveraging the same side channels employed by the offense to both target and attack co-located guest VMs, it may be possible for virtues to detect the presence of one or more additional instances. Detecting the presence of VMs provides valuable context and allows the frequency of migration to be tuned based on the critical mass of a given VM host. Research has shown that it is possible to detect the presence of specific co-resident VM instances using shared cache timing, memory bus timing, prime and probe, and memory deduplication copy-on-write timing [5, 24, 25]. Given that these techniques have been employed without inter-VM cooperation it is assumed that such a partnership between friendly instances could improve the reliability of these techniques.

AWS, along with other cloud providers, support the concept of configurable geographic zones for high-availability applications. AWS currently supports 14 different zones with plans to add four more in the near future. Star Lab proposes to (where possible) leverage these zones strategically, to increase the diversity of the migration events. Migrating between geographic zones ensures that virtues migrate between different hosts and increase the resources required to mount an attack or syphon observable information from any given virtue.

B.1.4. Application Support (Requirement 7)

Galahad's virtues will also support the use of Windows and Linux applications, to include legacy applications. A variety of potential solutions exist to support Windows applications. The solution that offers the most support is to license and deploy full functioning Windows VMs. This approach, however, is counter to attack surface minimization efforts and the cost for licenses and cloud execution time may be excessive. An alternate solution is to run applications within ReactOS [26]. ReactOS is open source (reducing licensing costs), but it still has a large attack surface and the ReactOS development community is small, implying the solution is high risk. Another approach that reduces the attack surface is to split up an application into parts that execute across multiple VMs, for instance, an application's user interface may run on one VM while the application's core logic executes on another. This approach would necessitate rewriting the application or transforming its binary representation. It would also require some form of cross-VM remote procedure calls, thus impacting application performance. A final alternate solution is the Drawbridge prototype from Microsoft [27]. While this solution attempts to strengthen isolation and attack surface reduction, it is not openly available and its maturity level is unknown.

Rather than using the aforementioned approaches, Star Lab will support invoking and interacting with Windows applications by running the applications within Wine [28] inside Unity. Wine is a long running open-source project dedicated to creating a Windows application compatibility layer for Linux. It also has a long history of adapting to new software that runs under newer versions of Windows. Wine was selected for several reasons. First, using Wine supports Star Lab's strategy to reduce each virtue's attack surface as much as possible. A complete Windows

VM presents an attack surface that is larger than necessary, as it includes not only the applications needed for a given virtue, but also a set of windows applications and services that are not needed by those applications. A Linux VM running Wine can be made to present a minimal attack surface. Second, Wine can be run as a Linux user with minimal privileges and with access to the smallest set of system and network resources necessary for the given virtue. This adds another layer of security and minimizes the attack impact. Thirdly, since this particular method of running Microsoft applications is less common than the typical setup (simply running applications in Windows), it requires more sophisticated tools and planning, making it an attractive target for only the highly skilled, resourced, and motivated attackers.

This method is not without its challenges; some Windows applications are not yet supported. One application that fits into this category is Microsoft Office 2013. None of the existing communities currently supporting Windows emulation (ReactOS, Wine, and the commercial company Code Weavers) have achieved full support for Office 2013, despite market pressures [29-31]. Still, the Wine community is large and active. Star Lab anticipates the community will start supporting Office 2013 during the contract period; however, if that is not the case, Star Lab will support the most recent version of Office that is supported in Wine, Office 2010. Also, though Star Lab does not have the intention of becoming heavily involved in the Wine community, it can still support the Wine community by providing more up-to-date assessments of Office and other customer-determined legacy software on Wine, analyzing cases where such software fails and providing feedback, crashdumps, and testing proposed changes to Wine.

Supporting Linux applications is more straight forward. In addition to again using de-privileged users, Star Lab will deploy Linux applications as Docker containers. Thus, each application will have its own kernel namespace, its own TCP/IP stack, and in its own chrooted view of the filesystem. The use of containers will also support lifecycle management strategies discussed in Section B.3.

B.2. R&D of Galahad's Canvas Interface (Task 2)

Users will invoke and interact with their various virtues through Galahad's Canvas. A user's canvas will exist within AWS and will be accessible from any location with access to AWS and from any type of machine capable of properly interfacing with AWS, e.g., thin client, mobile device, etc. Additionally, Galahad's Canvas will also manage inter-virtue resource sharing.

B.2.1. Presentation Interface or User Canvas (Requirement 9)

Star Lab will leverage the ideas behind Qubes OS to design and develop Galahad's Canvas. Qubes OS allows multiple VM instances to display their graphical content in a unified presentation interface. In Qubes OS, each virtual machine contains a process running in the application VM as well as a process running in the presentation interface. The two processes are connected via a bidirectional shared memory communication channel that allows events such as window creation, user input, window resize, repaint messages, etc. to be exchanged. This design

allows for the isolation of applications and window content while simultaneously providing a unified desktop environment.

Since virtues will be geographically disparate, a secure network channel, not shared memory, will be used to connect a virtue to Galahad's Canvas. These secure network channels will run in the Amazon Virtual Private Cloud (VPC). Amazon's VPCs allow for the creation of an isolated network that can further secure virtues by forbidding inter-virtue communication and encrypting all Canvas-to-virtue communication. Figure 5 illustrates the basic architecture of Galahad's Canvas. Each GUI application runs without modification and uses the existing open source *X window* system. The only modifications required will be swapping out the typical backend with a frame buffer accessible over a remote desktop protocols (RDPs) such as Spice. Control information exchanged between a virtue and Canvas will utilize a set of RPC functions Star Lab will implement in Galahad's Virtue-Canvas API. This allows Galahad's Canvas to be a small, cross platform application that manages the display of virtue windows on the thin client.

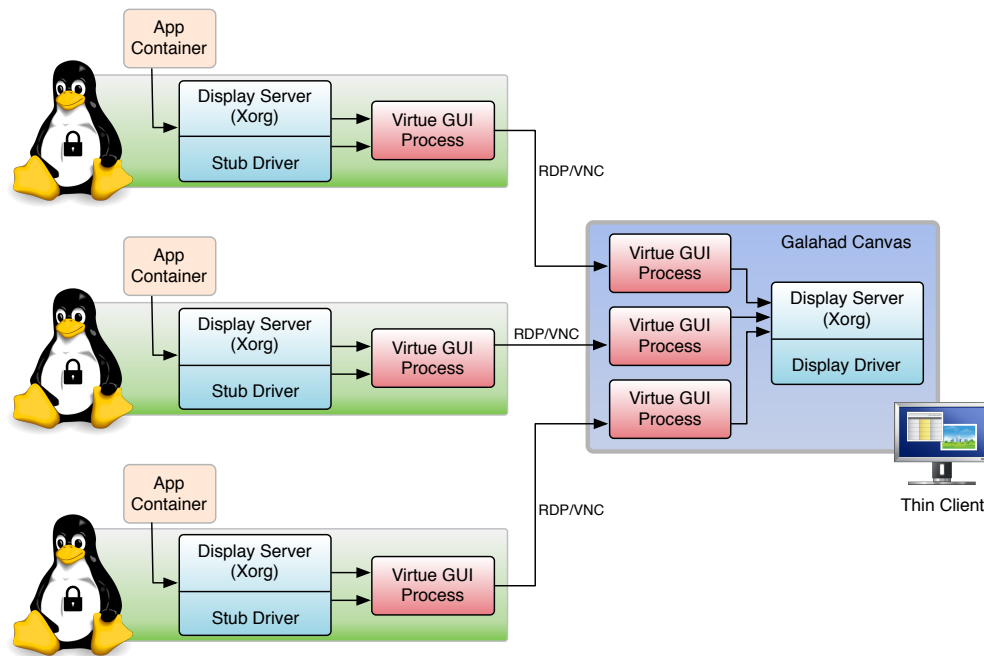


Figure 5: Galahad's Canvas Architecture

Galahad's Canvas will provide a window management layer for virtues by painting window borders and labeling the windows with names specific to each application and role. Finally, Galahad's Canvas will also allow users to stop and start virtues. For them, this is the equivalent of opening an application for a given role. Star Lab will borrow heavily from Qubes OS presentation layer and create a Start menu that allows users to first select a role, then an application within that role.

B.2.2. Resource Sharing (Requirement 8)

Galahad will include a secure and streamlined resource sharing protocol, through the Virtue-Canvas API. This protocol will enable the interactive sharing of data files and information between virtues and the sharing of software and system configurations. Sharing data across virtues, e.g., transferring URL received via email in one virtue to a browser in another, will be implemented to facilitate two interactive data-sharing behaviors: cross-virtue copy-and-paste of text data (clipboard sharing) and cross-virtue file transfer. Star Lab will build its resource sharing capabilities by adapting similar capabilities within Qubes OS and Cappsule2 [32].

Borrowing from Qubes OS's high-level workflows, coordinating inter-virtue clipboard sharing will be the responsibility of Galahad's Canvas. Like the sharing of application interactions and window manipulations, clipboard sharing exchanges will take place over the same secure private network that will enable RDP exchanges. In this manner, not only will virtue isolation be preserved through the Canvas, all virtue-to-virtue copy-and-paste behavior can be logged. Sensors for copy-and-paste behavior can be scaled up and down as necessary as well as based on the current threat scenario.

Each Unity instance will include a daemon that leverages the Virtue-Canvas API. When a user presses a special key combination, the Canvas will be provided access to the virtue's clipboard contents. The user will then select a destination virtue and invoke another special key combination. The latter key sequence triggers the transfer of the clipboard contents through Canvas to the destination virtue. Once Canvas passes the clipboard contents to the destination, it will clear its internal buffers to prevent other virtues from accidentally acquiring this data, and the user can paste within the destination VM as normal.

Cross-virtue file transfer within Galahad will also leverage the Virtue-Canvas API. Qubes OS leverages Xen's Dom0 to assist in cross-domain file sharing. Qubes OS's file manager runs on Dom0 and exposes each isolated VM's file structure to the user, who can use this manager to transfer files. Typical drag-and-drop behaviors are prohibited across isolated VMs in Qubes OS, and a user must instead use a special command to copy files between VMs (either through the file manager or the console).

As with enabling support for copy-and-paste, Qubes OS's mechanics for file transfer are easily adaptable to Galahad. The Virtue-Canvas API will include support for arbitrary file transfer to and from Canvas. Through this API, Canvas will enable the user to copy files between virtues. Canvas will request the file listing for each virtue and will present it to the user. When a user wishes to copy a file from one virtue to another, the user will request this operation through Canvas using a special key combination. When the user selects a destination virtue, the file's contents are streamed from the source virtue through Canvas (using the Virtue-Canvas API) and into the selected destination virtue. The target virtue handles writing the file's contents to its file system.

As with copy-and-paste, file transfers through Canvas can be logged. Furthermore, a policy mechanism can easily be implemented to prevent file transfer to or from specific virtues.

Additionally, Star Lab will build persistent setting support into Galahad's Canvas. The Canvas interface will contain a dialog to change settings that are common to all virtues, e.g. default printer or home directory location, settable in one location and shared with the individual virtues when they are started by the user. When the user changes one of these common settings, Canvas will attach the setting to the user's profile and will use the Canvas-Virtue API to share the setting with each of the users running virtues. When new virtues are started by Canvas, the complete set of common settings will also be shared over the Canvas-Virtue API.

B.3. R&D of Galahad's Lifecycle Management Construct (Task 3)

Critical to obtaining user acceptance of Galahad will be shielding users and administrators from interacting directly with AWS. To achieve this goal, Star Lab proposes the development of utilities, APIs, and services that can be integrated *under the hood* of Galahad, but which will serve to facilitate virtue creation, packaging, storage, launching, and termination.

B.3.1. Virtue Packaging (Requirement 11)

Star Lab proposes to research and develop a Virtue Assembler for Galahad. The Virtue Assembler will address Requirement 11 and will satisfy the two applicable functional measures: speed of packaging and accuracy of content reporting. Star Lab proposes to extend the Docker model of container creation to virtue creation so that virtues will include the same creation and portability advantages as Docker containers while maintaining the isolation that virtualization provides. Star Lab will leverage its experience creating packaging tools for its Crucible product to design and develop Galahad's Virtue Assembler.

As with Docker images, Galahad's Virtue Assembler will facilitate sharing and extendibility without having to follow cumbersome recipes or copying multiple disparate files. However, unlike Docker images, Virtue Assembler will provide convenient, assured methods to ascertain both the contents and configuration of the virtue package as well as the virtue package creator. This will enable the quick identification of unauthorized virtues and allow for the comparison of the contents of virtue packages with software vulnerability databases.

To create a role-based virtue package, the Virtue Assembler will use a configuration file called *VirtueFile*. Using this file, virtue administrators will specify a description of the virtue, i.e., specify a set of modifications that shall be made to a base virtue package to establish the role-based virtue. Example modifications may include: adding an application, adding non-executable files, executing commands that configure the virtue, and the application start-up order for the virtue. Additionally, the configuration file will allow the user to specify what resources, logging requirements, and protections the virtue requires. Finally, a *VirtueRole* construct will capture

role-specific attributes, for example, it will define any settings/constraints for role-to-role interactions.

Once the virtue is fully specified in the configuration file, Galahad's Virtue Assembler will build a new virtue with its output being a file bundle signed by the administrator's private X.509 key. The virtue file bundle will consist of a metadata section that contains the resources, logging requirements, protections, and the base system image of the virtue. The metadata file will also include a list of the software components present in the virtue at creation time, along with the X.509 certificate of the virtue creator and the signature attached to the virtue. Additionally, the file bundle will contain a disk image that leverages the standard QCow2 format that consists solely of the file system modifications that the virtue has on top of the base system image. By only storing the file system modifications, the virtue consists of a chain of modifications back to the initial base virtue. This minimizes the amount of storage required for any one virtue and support extending virtues over time. Once created, the virtue will be pushed to a virtue store where other administrators can retrieve the virtue for use or for extension. Since the virtue metadata contains a list of installed software components, all virtue settings, and the X.509 certificate of the virtue creator, the Virtue Assembler is able to extract audit information from the virtue. During the extraction process, the signed metadata is validated to ensure correctness.

Also, Virtue Assembler will include ways for administrators to define required software components individually or in groups. In the latter case, groups could be role-based such as *"document viewing"*, or *"router admin"*. The build tools will also support Docker containers, CentOS packages, and kernel drivers. This will provide the maximum flexibility to compose Unity with only the bare minimum components to achieve role-defined tasks.

B.3.2. Virtue Control API (Requirement 1)

Star Lab will build upon the existing AWS functionality to create the Virtue Control API to automate the storing, launching, and terminating of virtues. All virtues will be stored in a replicated, centralized registry similar to those used by existing container services such as Docker. The underlying repository data store will be encrypted and each virtue signed to ensure integrity at original storage and at run-time. Also, all communications with the data store and registry will be encrypted using TLS and unencrypted communications will be refused.

The storage extensions of the Virtue Control API will allow authorized administrators to upload images to the virtues repository. It will be built on the existing AWS S3 REST APIs to perform interactions with the cloud data store. When an administrator attempts to store a new image, the role of the user is validated along with the integrity of the new virtue. If both validations are successful, the user will be permitted to upload the virtue along with a user defined name and version. When a user attempts to search the virtue repository, their view is limited to the set of virtues associated with their roles.

Launching a virtue will utilize existing AWS API functionality, e.g., AWS *CreateInstance*, to start the instances and attach networking and data stores as required. A new virtue will be started with a preshared key (PSK) that will be used for remote access. The PSK will be unique to the running virtue and only the user's Canvas will have access to the key. Once the newly instantiated virtue is running, the user's Canvas will connect and start displaying the graphical content from applications(s) using the architecture described in Section B.2.1. When a user closes an application, the user's Canvas will first check to see if any other applications within the same role are running. If no other applications for that role are running, the user's Canvas will shut down or terminate the virtue via AWS *TerminateInstance*. Star Lab will also implement a 'Force Quit' capability as well, along with a service that terminates all virtues when a user terminates their Canvas.

Finally, the Virtue Control API will also provide mechanisms to gather status and monitoring information from executing virtues, e.g., resource utilization, network configuration information, firewall rules, etc. This will allow users to see what virtues are running, assess their performance and configuration, and potentially provide insight into their overall cloud usage over different periods of time. This latter capability will aid in educating users about their cloud utilization and potentially help control costs. Administrators will also have access to these controls so they can identify potentially rogue virtues.

B.4. R&D of Galahad's Sensing and Logging Capabilities (Task 4)

Star Lab will design Galahad to support dynamically adjustable sensors and numerous logging capabilities. Galahad's sensors will provide deep application and OS introspection. Also, Galahad will include a cross-layer sensor validation capability that will guard against adversary tampering. Each virtue will include a log digester responsible for minimizing, storing, processing, and aggregating messages from each sensor/logger. Finally, real-time control of sensors and loggers will be provided through Galahad's Sensing and Logging Control API.

B.4.1. Application and OS Sensing and Logging (Requirement 5)

Galahad leverages Wine to provide Windows application capability within its virtues. Wine uses a set of dynamically loadable libraries (DLLs) that are API-compliant with standard Windows DLLs to implement a Windows compatibility layer on top of the host operating system, in this case, Linux. This architecture introduces some inherent security advantages as mentioned in Section B.1.4. Wine implements the actual library functionality in a server process, *wineserver*, to which the DLLs forward service requests over a UNIX socket. Subsequently, *wineserver* emulates the appropriate task on the host OS and then forwards the results back to the Wine DLL, thus providing a service comparable to the Windows kernel.

Wine's *wineserver* provides an ideal instrumentation point. Most of the operations performed by Windows applications will require a call into the *wineserver*; filesystem interactions, process

creation calls, IPC operations with other processes, and administrative tasks all use this mechanism. The remaining operations, like network communications, are typically implemented directly in the Wine version of the Windows DLL. Thus, Star Lab will instrument the *wineserver* and the Wine DLL to log critical information. Actions in the *wineserver*, and actions taken by instrumented DLLs will be logged locally by Galahad and fed into a log digester.

Native Linux applications will be instrumented using standard application instrumentation techniques including SELinux audit logging, library interposition, and sensing points added to the Linux kernel interface. Additionally, the LSM mentioned in Section B.1.2 will also provide a useful point of introspection within the Linux kernel which can be used to identify malicious events and gain valuable context into overall system operation. Points of introspection include: program execution, task interaction, inode/file access, socket creation, key-management actions, and memory operations.

Table 8 provides some examples of what can be sensed at each of the various layers of the Galahad's virtues. Note that this list is not comprehensive, but shows the diversity of signals that can be observed by looking across the entire stack.

Table 8. Galahad Sensor Capabilities

Layer	Sensing Capabilities	
Wine	<ul style="list-style-type: none"> Windows-layer network activity Windows API calls 	<ul style="list-style-type: none"> Inter-process communications
Linux OS (Unity)	<ul style="list-style-type: none"> Process creation Storage access Network access 	<ul style="list-style-type: none"> Physical device access Libraries loaded by Wine process Attempted access to privileged resources
Valor	<ul style="list-style-type: none"> Network communications Virtual memory remapping 	<ul style="list-style-type: none"> Physical device access

In addition to providing more sensing, Valor and Unity will perform inter-layer validation: sensors at one layer will trigger an action that is visible to another layer to indicate some status. For example, a sensor inside Unity can make a periodic hypercall to indicate to Valor that the sensor is still running. Conversely, Valor can check the Unity hypercall's call stack to ensure that it is in fact the sensor making the hypercall. This provides a cross-layer signal that the sensing infrastructure is operating correctly.

A multi-scale aggregation and summarization process will minimize storage and transmission requirements for data gathered from these well-positioned sensors. Each layer of Galahad's sensing architecture will report events to a log digester located in Valor. The digester will collect logs, prioritize them, and apply aggregators and filters to identify important or interesting features. Aggregation and filtering policies will be customized to the applications being run inside the Virtue, and will be adjustable using the Sensing and Logging Control API. This is analogous to the SELinux policy distribution mechanism, where each application carries along the correct piece of policy to defend itself.

While several generically useful aggregation rules will be applied, e.g., average, frequency count, etc., aggregators will be customized to applications or written to identify features that are specific to a custom application. For example, a database client application may use a set of stored procedures to complete its tasks. While a generic database query logging sensor would perhaps identify which stored procedures are being run, a custom aggregator will extract and store only the query ID and the important arguments of procedures. This greatly reduces log storage requirements and focuses the logging on the security-critical features of the application's behavior. In Phase II, a great deal of interesting research may be done to create new application-specific aggregators, to identify useful information in available logs, and to distribute and validate these custom logging policies and filters.

B.4.2. Sensing and Logging Control (Requirement 5)

The inclusion of sensors also provides an opportunity to embed actuators, active response mechanisms that automatically take action to mitigate threats. Actuators will be embedded in each of the layers (application, OS, nested hypervisor) and will be tasked by the Sensing and Logging Control API. Examples of active response actions include closing connections, terminating processes, triggering migration, and revoking storage permissions. The Sensing and Logging Control API will also be used to provide “standing order” signature-based rules that specify an action to take when an activity is detected. This allows virtues to defend themselves when they see something that should be forbidden; for example, a policy may say that Notepad.exe should never make an outbound network connection. If it does, Galahad can take action to terminate the offending process.

Finally, the Sensing and Logging Control API will include extensions to retrieve logs. These extensions will use the REST model running over TLS and will implement best practices, such as certificate pinning, perfect forward secrecy, signed messages, and client certificates, to ensure the confidentiality and integrity of transmissions. The Sensing and Logging Control API will implement commands encoded in a structured language to make parsing safe and unambiguous.

B.5. Performance Considerations (Requirement 10)

Star Lab has proposed a number of specific techniques to balance good defensive tactics and overall system cost. Avoiding dedicated hardware instances and reducing the size of instances are examples of attempts to balance security, performance, and cost. AWS provides a number of instance types that have been tailored to specific cloud workloads. The only requirement which Star Lab's approach places on instance selection is HVM AMI support. The hardware acceleration provided by HVM support will be important for achieving the highest performance from Valor and is compatible with all current generation AWS instance types. Further, while Galahad will maintain a 1:1 AWS instance to virtue mapping, the specifics of instance selection will be dynamic, based on resource utilization metrics collected per user and per virtue.

Star Lab proposes to leverage its integrated AWS test harness (ref Section I, Detailed Management Plan, for more details) to aid in selecting baseline instance types. The selection process will take into account anticipated initial program metrics, various cloud workloads, and possibly host utilization. Ultimately, an optimized Galahad would include the capability to select instance types during launch and possibly during migration. Phase II analytics will be able to use data collected by Galahad to inform such decisions, and Galahad's Virtue Control API could be enhanced in Phase II to expose and facilitate the selection of instance types as well as other instance parameters. Realization of methods to enable the real-time creation of optimized instances would provide the best cos-based performance.

B.6. Program Execution and Risk Management

Upon contract award, Star Lab will immediately begin setting-up and configuring its AWS integrated development and test environment. This will relieve the team from low-level system maintenance tasks such as network configuration, hardware setup, etc. while also allowing the team to very quickly experiment and identify bugs and performance problems within an operationally relevant context. Initial research and development will focus on creating Galahad's virtue construct while enabling defenses and supporting Linux applications. Star Lab has already obtained and experimented with XenBlanket (the starting point for Valor) and, as has already been described, possess tools for building minimized operating systems and hypervisors. Initial development of Galahad's virtue construct will inherently lead to the development of building and packaging tools.

Concurrent with this development, BBN will research and develop sensors and logging mechanisms. Initially, BBN will use default Linux instances, however, once an initial virtue prototype exists, Star Lab will branch the code for both Unity and Valor so sensors can be integrated and tested.

Once Star Lab has successfully created its virtue prototypes, researchers will focus on interfacing them with AWS (Galahad's Virtue Control API) and users (Galahad's Canvas). By the midterm waypoint exam, Star Lab will be able to demonstrate the construction and launch of virtues inside AWS, as well as users invoking applications from Galahad's Canvas. These virtues will support Linux applications and a small subset of Windows applications, i.e., those already supported by Wine. Finally, the prototype used during the midterm waypoint exam will be able to demonstrate no less than 10 sensors and logging mechanism. The second half of the effort will focus on layering sensor and logging controls, enabling virtue resource sharing, enhancing application support, and strengthening defenses.

During the project, Star Lab will continually communicate its progress in monthly reports and other recurring meetings to include updates on any risks to the project. Star Lab will leverage a standard risk management process to identify, track, and manage key technical and non-technical project risks. Risks will be documented and tracked on the project's GitLab tickets. Risks will be

reviewed and discussed at the team’s daily stand ups. Risk mitigations will be identified as tasks and tracked as part of the GitLab issue tracker. High and Critical risks will be reported to IARPA monthly (included as part of our monthly progress). Table 9 present the highest priority technical risks and how they will be mitigated during the course of the project.

Table 9: Risks and Mitigation Strategies

Risk	Mitigation
User experience is negatively impacted by sensor collaboration or instrumentation within the compatibility layer	Eliminate traps into Valor, i.e., Unity to Valor context switching and reduce introspection points
Virtue start-up is too slow effecting adoptability	Continue to reduce size of Valor and Unity while also experimenting to identify best instance type
Galahad cannot support Office 2013 or other legacy applications	Use an earlier version supported by the compatibility layer or use a Windows VM instead of Unity
Unable to reliably detect co-residency in support of a migration strategy	Migrate on pre-determined regular intervals (based on role), still preferring offline migration when possible
Sensing and virtue C2 enable new virtue-only attacks	Collaborate T&E team to comprehensively understand new attack space; pursue preemptive, prevention techniques to reduce attack surface & mitigate risk; employ secure coding best practices
Virtue development and testing of Galahad causes Amazon to interrupt or deny service	Contact AWS and explain research goals or leverage T&E environment during development and testing

C. State-of-the-Art

Advancing the state-of-the art for secure, cloud-based user computing environments has traditionally been driven by cloud service providers (CSPs) like Microsoft, Amazon, and Google. While today’s CSPs have not offered new secure cloud computing paradigms, they have provided best practices and design patterns for infusing security in areas such as networking, data management, and software updating. In addition to best practices, many of the large CSPs offer anti-malware services [33], provided by third party vendors or using in-house capabilities. Lacking in all of these offerings is access to processor and hypervisor security capabilities such as the dom0, trusted platform module (TPM), Intel Trusted eXecution Technology (TXT), PCI passthrough, etc. As a result, customers cannot inspect VMs, measure or validate the trust of hypervisors, guarantee isolation, or put into practice other hypervisor hardening techniques.

To address the weak security offerings made available by cloud providers, industry has attempted to layer in security and control on top of the cloud. Ravello [34], for example, has developed a nested hypervisor called HVX, designed to run on already virtualized hardware and public cloud environments. In addition to executing traditionally unsupported VM formats, HVX also includes an overlay network to support the deployment of custom network topologies that allows security vendors to more easily control, monitor, and log VM-to-VM communications. Other cloud security solutions, like vArmour and GuardiCore, leverage software defined networking capabilities to segment, monitor, and deceptively respond to threats in data centers and cloud environments. Like the CSP-driven solutions, however, none of the above commercial solutions address all the threats considered as part of this project.

In contrast to protecting entire distributed cloud implementations, Menlo Security's Menlo Security Isolation Platform (MSIP) is an end-point security solution that uses cloud-based, short-lived VMs to execute potentially malicious content and then provide safe content to the end-user. MSIP does not require agent software be installed on end-point devices and computers; however, its protections are not available for all applications or use cases.

Moving away from cloud related security products in general, another isolation-driven solution is the aforementioned Qubes OS. Qubes OS prioritizes isolation and attack surface minimization for the desktop computing environment. Using the Xen type-1 Hypervisor and processor features such as Intel VT-d, Qubes OS creates Qubes, role-based, disposable computing environments in the form of virtual machines, that are isolated from one another. Users of Qubes OS interface with the Qubes through a windows manager or unified desktop. Qubes OS serves as a good model for this project, specifically its approach to a unified desktop, resource sharing, and its use of templates for defining new roles; however, the following limitations keep it from being directly applicable to an AWS UCE:

- Reliance upon Intel x86 security extensions and the TPM;
- Lack of remote viewing support; and
- Lack of cloud specific attack vector defenses.

The aforementioned Microsoft's Drawbridge is an example of application-focused, isolation construct. Drawbridge consists of two enablers: the picoprocess, and library OS. The picoprocess is a secure isolation container that protects a host from a potentially-malicious application by restricting the kernel API surface. Drawbridge's LibOS is a collection of user-mode and some kernel components of Windows that allows applications built for earlier versions of Windows to execute on new versions. Together, the picoprocess and LibOS form a Windows container capable of supporting legacy applications. Drawbridge has been used to further research in fault-tolerant computing [35]. Unfortunately, while Microsoft has published several papers, they have not released the code for Drawbridge.

Academia too has investigated the security of public clouds. Efforts to protect VMs and applications from potentially malicious hypervisors or peers within a CSP have focused on trusted security monitoring deployed in the form of a nested hypervisor [12]. The use of a nested hypervisor decouples the security features from the virtual machine manager allowing high performance integrity verification and the potential use of high assurance software languages. However, the capabilities that provide integrity verification and trust are not legitimately available to AWS users thereby weakening their security claims.

In contrast, Star Lab's Galahad seeks to strike the correct balance of security, performance, and functionality. Central to Star Lab's approach is an acknowledgement that it is impossible to trust any CSP's hypervisor. Not only can they not be attested at boot, but they cannot be trusted to provide the necessary isolation.

D. Data Sources

Star Lab does not anticipate the need for any existing data sources.

E. Deliverables

Table 10 summarizes the reporting and technical deliverables, and the frequency of delivery. Star Lab proposes to provide all deliverables with Government Purpose Rights.

Table 10. Technical and Reporting Deliverables

Item	Month
Slide presentations (kickoff, on-site visits, & Phase II Proposers Day)	1, 4, 11, 12
Phase 2 slide presentation	11
Test plans	9, 18
Source code, prototypes, and documentation	6, 12, 18
Conference paper	16
Monthly reports	Monthly
Cost reports	Monthly
Final report	Conclusion of Phase I,

F. Cost, Schedule, Milestones

Table 11 shows Star Lab's proposed cost for the proposed research, including estimates of cost by task.

Table 11: Cost Summary by Task and Subtask (No Fee)

Task #	Task Description	Cost
1	Virtue Construct R&D	\$405,814.76
1.1	Design & Develop Threat Mitigation Enablers	\$277,195.94
1.2	Integrate Application Support Enablers	\$128,618.82
2	Canvas R&D	\$246,583.34
2.1	Design & Develop Canvas Interface	\$104,894.50
2.2	Design & Develop Resource Sharing	\$141,688.84
3	Virtue Lifecycle Management R&D	\$356,373.80
3.1	Design & Develop Virtue Building and Packing Tools	\$114,116.65
3.2	Design & Develop Virtue Control	\$242,257.15
4	Sensor and Logging R&D	\$855,865.32
4.1	Design & Develop Sensor Suite	\$432,229.13
4.2	Design & Develop Sensor Control	\$423,636.20
5	System Testing and Experimentation	\$193,185.82
6	Technology Demonstration and Evaluation	\$163,767.79
7	Reporting and Management	\$209,242.88
Total		\$2,430,833.71

Figure 6 provides a Gantt chart schedule for all SOW tasks. Tasks highlighted in green will primarily be completed by Star Lab. BBN will be primarily responsible for tasks in red. Yellow tasks will be completed jointly between Star Lab and BBN.

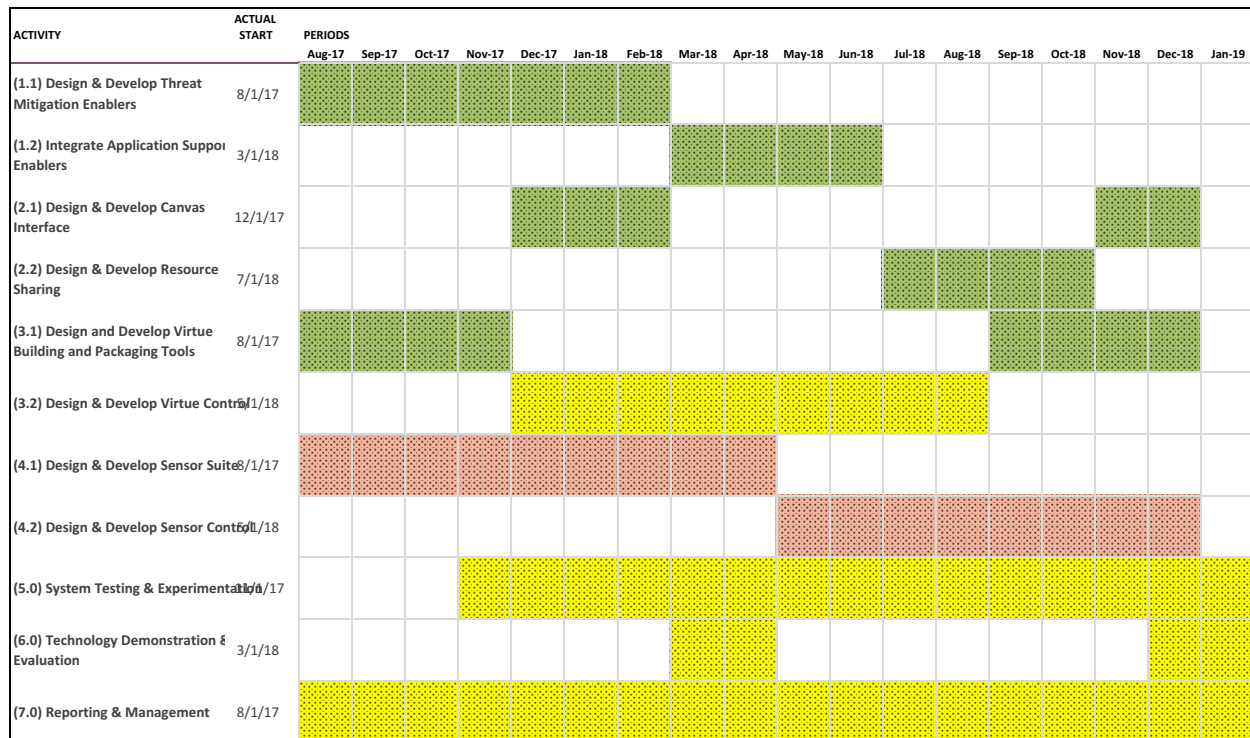


Figure 6: Research and Development Schedule

Using the Gantt chart above, Table 12 presents a list of milestones and dates for the proposed research.

Table 12. Milestones and Dates for Phase I

Milestone	Milestone Description	Month	Metrics
A	Kick-Off Meeting	1	All
B	Initial prototype of Valor launched using existing AWS interfaces	2	All
C	Initial prototype of Unity launched inside Valor using existing AWS interfaces	2	All
D	Launch initial Galahad Virtue prototype via initial AWS Control API	3	11.a, 11.b
E	Demonstrate < 5 Galahad logging options	3	5.a, 5.b
F	Demonstrate <10 Galahad Virtue protections / attack surface minimizations	4	2.a, 2.b, 2.c, 2.d, 2.e, 4.a, 6.a, 6.b, 6.c
G	Demonstrate < 10 supported applications (Linux only)	4	7.b
H	Demonstrate live migration of a single Galahad Virtue	5	4.a, 6.a, 6.b, 6.c
I	Demonstrate initial Galahad Virtue packaging without AWS deployment	6	11a, 11.b
J	Demonstrate < 10 Galahad logging options	6	5.a, 5.b
K	Link prototype virtue with initial canvas prototype	6	9.a, 9.b
L	Demonstrate live migration of a single Galahad Virtue with Canvas integration	7	4.a, 6.a, 6.b, 6.c
M	Demonstrate > 20 Galahad Virtue protections / attack surface minimizations	7	2.a, 2.b, 2.c, 2.d, 2.e, 4.a, 6.a, 6.b, 6.c
N	Demonstrate > 10 supported applications (Linux only)	7	7.b
O	Collect initial performance and resource utilization measurements	7	10.a, 10.b

Milestone	Milestone Description	Month	Metrics
P	Integration, testing, and experimentation of complete Galahad prototype for tech demonstration and evaluation	8	All
Q	Tech demonstration and evaluation	9	All
R	Demonstrate initial Galahad Virtue packaging with AWS deployment	10	11a, 11.b
S	Demonstrate > 20 Galahad logging options	10	5.a, 5.b
T	Demonstrate > 20 supported applications (Windows and Linux)	10	7.a, 7.b
U	Present Galahad at Phase II industry day	11	N/A
V	Demonstrate 2+ virtues with Canvas prototype satisfying interfacing requirements	14	9.a, 9.b
W	Demonstrate 3+ resource sharing use cases	14	11a
X	Demonstrate initial Galahad Virtue packaging satisfies metrics	14	11a, 11.b
Y	Demonstrate Galahad Virtue protections / attack surface minimizations satisfy metrics	14	2.a, 2.b, 2.c, 2.d, 2.e, 4.a, 6.a, 6.b, 6.c
Z	Demonstrate Galahad Virtue logging satisfies metrics	15	5.a, 5.b
AA	Demonstrate ≥ 6 Galahad virtues interfacing satisfy metrics	16	9.a, 9.b
AB	Demonstrate performance and resource utilization measurements satisfy metrics	16	10.a, 10.b
AC	Submit conference paper	16	N/A
AD	Integration, testing, and experimentation of complete Galahad prototype for tech demonstration and evaluation	17	All
AE	Tech demonstration and evaluation	18	All
AF	Deliver final materials	18	N/A

G. Offeror's Previous Accomplishments

Star Lab is an industry-leading security engineering company focused on the research and development of secure virtualization technologies for critical systems. As the related projects described below reflect, Star Lab has a strong pedigree and past performance protecting Department of Defense (DoD) systems against nation-state level offensive-cyber and reverse-engineering threats. Specifically, Star Lab has learned several techniques for protecting systems such as attack surface minimization, isolation, and moving target defense. These capabilities, as well as Star Lab's deep understanding of attacker T3Ps, will be brought to bear on VirtUE to research and develop Galahad.

Project Emerald - PM: Judah Nyden, Navy's PEO Integrated Warfare Systems (IWS)

Star Lab, working as a subcontractor for Draper Laboratory, is designing and developing a tamper-resistant Intel-based processor solution that provides protection against all classes of physical, electro-magnetic, logical, cyber, and side-channel reverse engineering attacks. This effort is in support of DoD program protection requirements to protect critical technology against unauthorized access, theft, and reverse engineering. The solution has been designed to be form/fit/function compatible with existing Intel-based processors. The solution includes a trusted supervisory controller within the processor package. During system operation, an enhanced Xen-based hypervisor developed by Star Lab enforces logical isolation of processing cores, cache, and main memory. The hypervisor also interfaces with the trusted supervisory controller

hardware in order to maintain the integrity of the processor configuration and respond to any tamper events.

DARPA's XD3/MAGICWAND - PM: Dr. Stewart Wagner, DARPA

Star Lab, working as a subcontractor for Invincea Labs, is designing MAGICWAND, an applied research and development project to demonstrate a groundbreaking approach to detecting low volume, distributed denial of service (LVDDoS) attacks. MAGICWAND detects adverse situations by drawing state information in real time from isolated network stacks and automatically inferred application state instrumentation. Star Lab is primarily responsible for researching and developing MAGICWAND's instrumented network stack by leveraging unikernels, an emerging innovation for developing and deploying secure light-weight application stacks that contain a minimal set of libraries, services, and code. The use of unikernel-based network stacks represents a significant innovation and provides a variety of benefits. Unikernels allow for the restructuring of application and kernel code to create optimized singular runtime environments in which the potential attack surface available to adversaries is greatly reduced by removal of unnecessary code. Additionally, the use of unikernels allows for Star Lab to implement several attack mitigation strategies. For example, Star Lab is researching diversified network stacks to make stack behavior less predictable and adjustable stacks to directly counter attacks.

MLSDroid - PM: LTJG Eric A. Carlson, NSA

Star Lab is currently developing MLSDroid, a Xen hypervisor-based solution that enables multiple security levels to execute on a mobile Android device. As with conventional mobile devices, the user interacts with a fully-featured Android domain. Unlike traditional mobile devices, however, MLSDroid enforces security policies outside of the Android domain and moves that enforcement into functional domains. Storage and VPN domains protect data at rest and in transit commiserate with the device's current security level. An authentication domain authenticates the user before allowing the device to change security levels, removing the ability for a compromised Android domain to steal user credentials. The secure domain switching process allows the user to switch security levels at will and ensures that sensitive data cannot be exfiltrated during a security level transition. MLSDroid uses ARM's System Memory Management Unit (SMMU) to isolate devices such as the Wi-Fi and baseband radios, storage devices, and the USB port to specific domains. Star Lab's MLSDroid increases productivity for users that need access to systems at varying security levels by allowing security level switching without rebooting the device.

Crucible® - PM: Irby Thompson, Star Lab Corp.

Star Lab is the developer and maintainer of Crucible® – a secure virtualization framework based on the open-source Xen hypervisor that is targeted for embedded & tactical computing environments. Crucible provides a trusted execution environment for operational mission

systems and addresses concerns unique to safety and security-critical computing, including: secure boot, technology protection, deterministic performance, high-assurance operations, cyber resilience, and compatibility with common mission system hardware and software.

Crucible is designed for use in hostile computing environments, and operates as trusted supervisory control software within the processor – configuring and controlling both hardware resources and software execution in order to ensure and maintain the integrity of system operations. Crucible leverages hardware-based roots-of-trust and hardware security modules to perform a secure boot process, and can optionally leverage hardware-provided security services at runtime. During system operation, the separation hypervisor enforces logical isolation such that software mission loads execute within private enclaves, even though they may be running on a single physical processor board. This addresses safety, security and confidentiality requirements, and improves overall mission assurance – especially given the reality of software faults and cyber-attacks. Finally, Crucible has strong technology protections and anti-reverse engineering features built directly into the hypervisor.

Transparent Computing (BBN) - PM: Dr. Angelos Keromytis, DARPA

BBN is developing technologies to record and preserve the provenance of all system elements/components (inputs, software modules, processes, etc.); dynamically track the interactions and causal dependencies among cyber system components; assemble these dependencies into end-to-end system behaviors; and reason over these behaviors, both forensically and in real-time.

H. Facilities

Star Lab, as a company, is small and young; however, Star Lab researchers have decades of experience performing research and development on complex systems. In addition to its research and development capability, Star Lab has also successfully completed technology transfer and systems integration activities wherein security technologies were placed deep within several weapon system platforms. Star Lab's small, but accomplished staff works in three primary locations: Washington, DC, Huntsville, AL, and San Antonio, TX. Each office—all three approximately 1800 square feet—is designed to promote mutual problem solving in a team environment. Also, each office includes access to an extensive development and testing lab including Intel, ARM, and PowerPC reference platforms with tactical operating systems such as VxWorks, Linux, Solaris and Windows. This laboratory includes military-class hardware for research and development purposes. Finally, Star Lab's Washington, DC office is approved for classified SECRET storage, processing, and discussion.

I. Detailed Management Plan

Star Lab's proposed leadership team consists of a highly technical and innovative principal investigator (PI) supported by an experienced program manager (PM). Matthew Leinhos will

serve as Star Lab's PI bringing 15 years of experience leading projects involving the design and development of security-connected operating system and virtualization technologies. Star Lab will prioritize the integration of an AWS based test harness which makes use of the CloudWatch API for maintaining performance metrics. This test harness will aid in reducing the overall risk of the proposed development effort and support greater interaction and technical exchanges with the test and evaluation team. Additionally, Star Lab will implement a Test-Driven Development (TDD) methodology, driven by the program metrics called out in the BAA. This will ensure that a direct mapping exists between requirements and verifiable components, with tests running directly within AWS rather than an internal simulated cloud environment.

Supporting Matt as the PM will be Adam Fraser. Adam is the Vice President of Research for Star Lab with the authority to make resource decisions for Star Lab. He has also served both as a PI and a PM on dozens of research projects across the DoD and intelligence agencies. Adam will ensure Matt has the resources required to successfully research, develop, and deliver Galahad. Adam will coordinate with SETA and agent personnel to ensure deliverables are received, budgets and spending are understood, and invoices are delivered in a timely manner. This will allow Matt to focus on technical development, project management, and risk mitigation.

For risk management, Star Lab uses the traditional identify, assess, plan, and implement risk management approach. Each month as Star Lab documents its monthly accomplishments, the research team will also identify technical and programmatic risks. The PI and PM will discuss any risks and identify several mitigation strategies. Next, the team will plan ways to mitigate any identified risks and implement them immediately. The results of this continual risk analysis will be recorded in monthly and quarterly reports.

Star Lab has teamed with Raytheon BBN, an experienced IARPA performer located just north of IARPA in Columbia, Maryland. As part of the teaming agreement, BBN will provide Star Lab both technical support to fulfill the tasks as outlined in the SOW and the necessary inputs to complete all documentation and reports as required by the government. Specifically, BBN will focus on Galahad's sensors and the Sensing and Logging Control API. Star Lab and BBN have already negotiated terms and conditions, non-disclosure agreements, and subcontracting agreements as required. The team is fully assembled and ready to start work upon contract award. At the foundation of the team is a strategic partnership established over the past year via the members' shared interest and technological expertise in cyber security and low-level security design. Coordinating the research tasks across the team will occur with weekly conference calls together with email and a centralized team documentation and code repository. This will ensure open and consistent lines of communication between Star Lab and BBN.

Table 13 is a list of personnel showing a concise summary of their qualifications, previous accomplishments and related experience. The table also shows the utilization of key personnel.

Table 13. Profiles of Key Personnel

Key Technical Staff (% time commitment)	Relevant Experience	Role / Task
Matt Leinhos Star Lab MS, Colorado State University (75%)	<ul style="list-style-type: none"> 10+ years in security and virtualization research and development, including vulnerability research, hypervisor development, and reverse engineering 5+ years in Windows kernel development, specializing in rootkit design, malware command & control, and data collection & logging Designed and implemented multi-threaded shared memory protocol within the Rump Unikernel 	Principle Investigator (Tasks 1- 7) Key Personnel
Derek Straka Star Lab MS, University of Minnesota (75%)	<ul style="list-style-type: none"> 10+ years in embedded systems and security software development 6+ years in Linux development of high performance video and network device drivers and associated applications Software lead developing a system to protect applications from malicious kernels 	Control and Interface Lead (Tasks 2, 4, 5, & 6) Key Personnel
Tom Case Star Lab BS, Georgia Institute of Technology (100%)	<ul style="list-style-type: none"> 10+ years in kernel and hypervisor development specializing in software protection and vulnerability research Lead developer for light-weight hypervisor-based protection system capable of encrypted code execution on Intel VTX platforms Performed red team activities using memory forensic tools to defeat anti-tamper/software protection capabilities 	Security Lead (Tasks 1, 3, 5, & 6)
Alex Jordan BBN MS, Thayer School of Engineering, Dartmouth College (40%)	<ul style="list-style-type: none"> 11+ years of cyber security and software development experience focused on operating system internals, reverse engineering, binary static analysis, and embedded development Experienced DARPA PI; lead development of application instrumentation tools and automated attack surface analyzers for firmware Designed and developed network- and host-based anomaly detection, multi-hypothesis tracking, and stochastic control systems 	Sensor Lead (Tasks 3, 4, 5, & 6) Key Personnel
Adam Fraser Star Lab MS, Air Force Institute of Technology (25%)	<ul style="list-style-type: none"> Experienced DoD and IC PI and PM with both large and small companies 15+ years in computer security industry and research award winner in the areas of anonymous communications, cyber deception, and cyber security Seven years as a computer crime investigator for AFOSI; Crimes Investigator of the Year (2004) 	Program Manager (Task 7) Key Personnel

J. Resource Share

Star Lab does not request any support from the Government, such as use of facilities or equipment or materials. Further, no cost sharing is proposed.

K. Other Supporters

Star Lab has not provided this proposal to another federal, state, or local agency or other party, nor is it receiving funding for a similar effort.

4 Attachments

Below are the applicable attachments to Star Lab's proposal.

Attachment 1: Academic Institution Acknowledgement Letter

Star Lab is not academic institution, nor is it partnering with an academic institution. No Academic Institution Acknowledgement Letter is necessary.

Attachment 2: Restrictions on Intellectual Property

Except as otherwise stated below, the Government will obtain unlimited rights to any technical data and/or computer software first produced by Star Lab and its subcontractor Raytheon BBN under any resultant contract for the VirtUE Program.

Any commercial hardware or software acquired by Raytheon BBN in support of the proposed effort will be delivered subject to the manufacturer's or vendor's standard commercial terms and conditions.

Regarding Open Source Software (OSS), Star Lab anticipates that in order to achieve program goals in a cost-effective manner, it will need to use and modify OSS during contract performance and deliver OSS to the Government. As asserted in Table 14, Star Lab expects to use, modify and deliver several open source applications; however, it is not possible at this time to pre-determine and assert which other OSS may be needed during performance. During the design and development tasks under any resultant contract, and in collaboration with the and Government COTR/PM, Star Lab will select and identify any additional OSS necessary to achieve program goals. Star Lab assumes that the Government will permit the identification and use of additional OSS without requiring consideration from the contractor.

Table 14. Commercial Items

Technical Data, Computer Software to be Furnished with Restrictions	Basis for Assertion	Asserted Rights Category	Name of Person Asserting Restrictions
WINE	Software developed or regularly used for nongovernmental purposes which (i) has been sold, leased or licensed to the public or (ii) has been offered for sale, lease or license to the public.	Commercial Open Source (https://www.winehq.org/licenses) License: GNU LGPL	https://www.winehq.org/
CentOS	Software developed or regularly used for nongovernmental purposes which (i) has been sold, leased or licensed to the public or (ii) has been offered for sale, lease or license to the public.	Commercial Open Source (http://mirror.centos.org/centos/5/os/i386/GPL) License: GNU GPL	https://www.centos.org
XenBlanket	Software developed or regularly used for nongovernmental purposes which (i) has been sold, leased or licensed to the public or (ii) has been offered for sale, lease or license to the public.	Commercial Open Source (https://code.google.com/archive/p/xen-blanket/) License: GNU GPL v2	https://code.google.com/archive/p/xen-blanket/

No patented technology will be incorporated into any deliverable.

Attachment 3: OCI Waiver/Determination/Notification or Certification

June 16, 2017

Office of the Director of National Intelligence
Intelligence Advanced Research Projects Activity (IARPA)
Virtuous User Environment (VirtUE)
ATTN: Kerry Long
Washington, DC 20511

Subject: OCI Certification

Reference: VirtUE, IARPA-BAA-16-12

Dear Mr. Long,

In accordance with IARPA Broad Agency Announcement IARPA-BAA-16-12, Section 3.A.1, Procurement Integrity, Standards of Conduct, Ethical Considerations, and Organizational Conflicts of Interest (OCI), and on behalf of Star Lab I certify that neither Star Lab nor Raytheon BBN have any potential conflict of interest, real or perceived, as it pertains to the VirtUE program.

If you have any questions, or need any additional information, please contact Adam Fraser at 210-542-0777 or adam@starlab.io.

Sincerely,

A handwritten signature in cursive script, appearing to read "Irby Thompson, Jr.", written in dark ink.

Irby Thompson, Jr.
President and CEO, Star Lab Corporation

Attachment 4: Bibliography

- [1] Boutuille, Jérémie. Xen Exploitation Part 2: XSA-148, From Guest to Host. July 27, 2016, <http://blog.quarkslab.com/xen-exploitation-part-2-xsa-148-from-guest-to-host.html>.
- [2] Ducklin, Paul. The VENOM “Virtual Machine Escape” Bug – What You Need To Know. May 14, 2015, <https://nakedsecurity.sophos.com/2015/05/14/the-venom-virtual-machine-escape-bug-what-you-need-to-know/>.
- [3] Ducklin, Paul. Xen Fixes Another “Virtual Machine Escape” Bug. July 30, 2015, <https://nakedsecurity.sophos.com/2015/07/30/xen-fixes-another-virtual-machine-escape-bug/>.
- [4] Ristenpart, Thomas, Eran Tromer, Hovav Shacham, and Stefan Savage. "Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds." In *Proceedings of the 16th ACM conference on Computer and communications security*, pp. 199-212. ACM, 2009.
- [5] Inci, Mehmet Sinan, Berk Gulmezoglu, Gorka Irazoqui, Thomas Eisenbarth, and Berk Sunar. *Seriously, get off my cloud! Cross-VM RSA Key Recovery in a Public Cloud*. Cryptology ePrint Archive, Report 2015/898, 2015. <http://eprint.iacr.org>, 2015.
- [6] Madhavapeddy, Anil, Richard Mortier, Charalampos Rotsos, David Scott, Balraj Singh, Thomas Gazagnaire, Steven Smith, Steven Hand, and Jon Crowcroft. "Unikernels: Library operating systems for the cloud." *ACM SIGPLAN Notices* 48, no. 4 (2013): 461-472.
- [7] Madhavapeddy, Anil, Thomas Leonard, Magnus Skjogstad, Thomas Gazagnaire, David Sheets, Dave Scott, Richard Mortier et al. "Jitsu: Just-in-time summoning of unikernels." In *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*, pp. 559-573. 2015.
- [8] Azab, Ahmed M., Peng Ning, Zhi Wang, Xuxian Jiang, Xiaolan Zhang, and Nathan C. Skalsky. "HyperSentry: enabling stealthy in-context measurement of hypervisor integrity." In *Proceedings of the 17th ACM conference on Computer and communications security*, pp. 38-49. ACM, 2010.
- [9] Schiffman, Joshua, Thomas Moyer, Hayawardh Vijayakumar, Trent Jaeger, and Patrick McDaniel. "Seeding clouds with trust anchors." In *Proceedings of the 2010 ACM workshop on Cloud computing security workshop*, pp. 43-46. ACM, 2010.
- [10] Santos, Nuno, Rodrigo Rodrigues, Krishna P. Gummadi, and Stefan Saroiu. "Policy-sealed data: A new abstraction for building trusted cloud services." In *Presented as part of the 21st USENIX Security Symposium (USENIX Security 12)*, pp. 175-188. 2012.
- [11] Haeberlen, Andreas, Paarijaat Aditya, Rodrigo Rodrigues, and Peter Druschel. "Accountable Virtual Machines." In *OSDI*, pp. 119-134. 2010.
- [12] Zhang, Fengzhe, Jin Chen, Haibo Chen, and Binyu Zang. "CloudVisor: retrofitting protection of virtual machines in multi-tenant cloud with nested virtualization." In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*, pp. 203-216. ACM, 2011.
- [13] Williams, Dan, Hani Jamjoom, and Hakim Weatherspoon. "The Xen-Blanket: virtualize once, run everywhere." In *Proceedings of the 7th ACM european conference on Computer Systems*, pp. 113-126. ACM, 2012.

- [14] He, Xin, Prashant Shenoy, Ramesh Sitaraman, and David Irwin. "Cutting the cost of hosting online services using cloud spot markets." In *Proceedings of the 24th International Symposium on High-Performance Parallel and Distributed Computing*, pp. 207-218. ACM, 2015.
- [15] Bromium. <https://www.bromium.com>.
- [16] Qubes OS. <https://www.qubes-os.org>.
- [17] Damm, Magnus; Horman, Simon. "The Xen Port of Kexec / Kdump A Short Introduction and Status Report." Xen Summit 2006. http://www-archive.xenproject.org/files/summit_3/kexec_kdump.pdf
- [18] Libseccomp. <https://github.com/seccomp/libseccomp>
- [19] Rump Kernels. <http://rumpkernel.org>.
- [20] Chris Wright, Crispin Cowan, Stephen Smalley, James Morris, and Greg Kroah-Hartman. Linux security module framework. In 2002 Ottawa Linux Symposium, June 2002.
- [21] Razavi, Kaveh, Ben Gras, Erik Bosman, Bart Preneel, Cristiano Giuffrida, and Herbert Bos. "Flip feng shui: Hammering a needle in the software stack." In *Proceedings of the 25th USENIX Security Symposium*. 2016.
- [22] Xen Security Advisory CVE-2014-7188 / XSA-108 version 4. <http://xenbits.xen.org/xsa/advisory-108.html>.
- [23] Moon, Soo-Jin, Vyas Sekar, and Michael K. Reiter. "Nomad: Mitigating arbitrary cloud side channels via provider-assisted migration." In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pp. 1595-1606. ACM, 2015.
- [24] Xiao, Jidong, Zhang Xu, Hai Huang, and Haining Wang. "Security implications of memory deduplication in a virtualized environment." In *2013 43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pp. 1-12. IEEE, 2013.
- [25] Wu, Zhenyu, Zhang Xu, and Haining Wang. "Whispers in the hyper-space: high-bandwidth and reliable covert channel attacks inside the cloud." *IEEE/ACM Transactions on Networking (TON)* 23, no. 2 (2015): 603-614.
- [26] ReactOS. <https://www.reactos.org>.
- [27] Porter, Donald E., Silas Boyd-Wickizer, Jon Howell, Reuben Olinsky, and Galen C. Hunt. "Rethinking the library OS from the top down." In *ACM SIGPLAN Notices*, vol. 46, no. 3, pp. 291-304. ACM, 2011.
- [28] WineHQ. <https://www.winehq.org>.
- [29] Code Weavers Forum Discussion. <https://www.codeweavers.com/compatibility/crossover/forum/microsoft-office-2013?msg=159073>
- [30] WineHQ Application Support Database. Word Support. <https://appdb.winehq.org/objectManager.php?sClass=application&iId=10>
- [31] ReacOS Discussion Board. <https://www.reactos.org/forum/viewtopic.php?p=114462&sid=8012b6b5a940816f89772f73627c4ca5>

[32] Cappsule. <https://cappsule.github.io/>

[33] "Microsoft Antimalware for Azure Cloud Services and Virtual Machines." 19 Sep 2016.
<https://azure.microsoft.com/en-us/documentation/articles/azure-security-antimalware/>

[34] "HVX: Virtual Infrastructure For the Cloud". Oracle Revello.
<https://www.ravellosystems.com/technology/hvx>

[35] Lorch, Jacob R., Andrew Baumann, Lisa Glendenning, Dutch Meyer, and Andrew Warfield.
"Tardigrade: leveraging lightweight virtual machines to easily and efficiently construct fault-tolerant services." In *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*, pp. 575-588. 2015.

Attachment 5: Relevant Papers

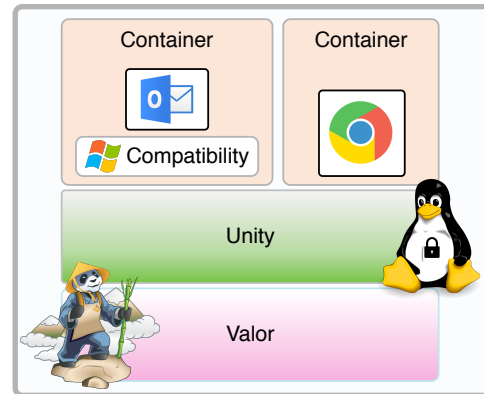
Attack Surface Reduction by Doug Goldstein, Star Lab Corporation. Prepared for Xen Summit 2016

Surrogating Software Stacks using Rump Unikernels by Dr. Mark Mason and Matt Leinhos, Star Lab Corporation

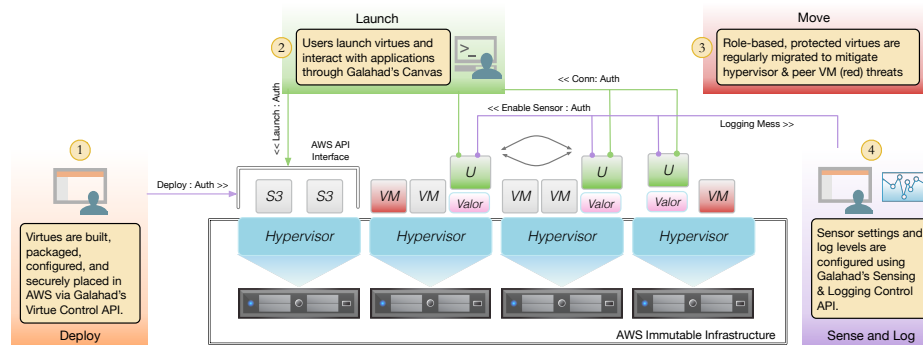
Attachment 7: Summary Charts

Overview - Galahad

- Galahad leverages role-based isolation, attack surface minimization, OS and app hardening, real-time sensing, and maneuver / deception to create a defensible UCE
- Galahad makes no attempt to establish trust, nor does it require specialized AWS services
- Galahad makes it more difficult for the adversary to co-locate with targets, while also requiring they consume more resources
- Galahad's virtue construct consists of a nested hypervisor (Valor) and a small, hardened, de-privileged Linux OS VM (Unity)
 - Valor facilitates the regular, recurring live migration of Unity VMs inside AWS
 - Unity provides sensing and support for Linux and Windows applications



Key Innovation



- A small lightweight nested hypervisor to enable live migration of virtues inside AWS
- A customizable presentation interface to facilitate secure interactions with AWS-based virtues and enables necessary resource sharing
- A virtue control layer to securely interface with AWS storage, manage virtues, and inform cost decisions
- Administrative tools to provide situational awareness and the easy management of virtues
- Dynamic configurable sensors and logging spanning the entire virtue software stack

Expected Impact

- Galahad users will leverage AWS for desktop-like computing, but with a significantly improved security posture
- Deliverable 1: Galahad Virtue
 - Limits time an adversary has to access targeted virtues in AWS while also reducing overall attack surface
 - Supportive of legacy applications
- Deliverable 2: Galahad Canvas
 - Accessible from a thin client, enables virtue-to-virtue sharing, and supportive of at least 6 roles
- Deliverable 3: Galahad Lifecycle API
 - Useful to moderate-skilled administrators and scalable to organizations of various sizes
- Deliverable 4: Galahad Sensors and Loggers
 - Captures hundreds of logging options across the entire virtue software stack
 - Facilitates improved analytics and sensor control (Phase II)
- R&D will advance several open source communities like Xen, XenBlanket, Linux, & Wine