JFAC/DAU/CSIAC Cyber Experiment (CYBEX)

Basic Center Operations (BCO) FA8075-17-D-0001

Tuesday, August 07, 2018

PREPARED FOR:

The Joint Federated Assurance Center (JFAC)

The Defense Acquisition University (DAU)

The Defense Technical Information Center (DTIC)

PREPARED BY:

CSIAC Cyber Security & Information Systems Information Analysis Center

266 Genesee Street
Utica, NY 13502

https://www.csiac.org
info@csiac.org

Cyber Security & Information Systems Information Analysis Center

Basic Center Operations (BCO)

FA8075-17-D-0001

CSIAC REPORT:

JFAC/DAU/CSIAC Cyber Experiment (CYBEX)

Sep 2018 (Quarter 4)

This Final Report is delivered by the CSIAC BCO to document the results of the JFAC/DAU/CSIAC Cyber Experiment during the fourth performance quarter of 2018. The Cyber Security and Information Systems Information Analysis Center (CSIAC) is operated by Quanterion Solutions Incorporated.

Approvals:

Michael W. Weir ______ Date 2/7/2019

Michael Weir, CSIAC Director

CSIAC is operated by:

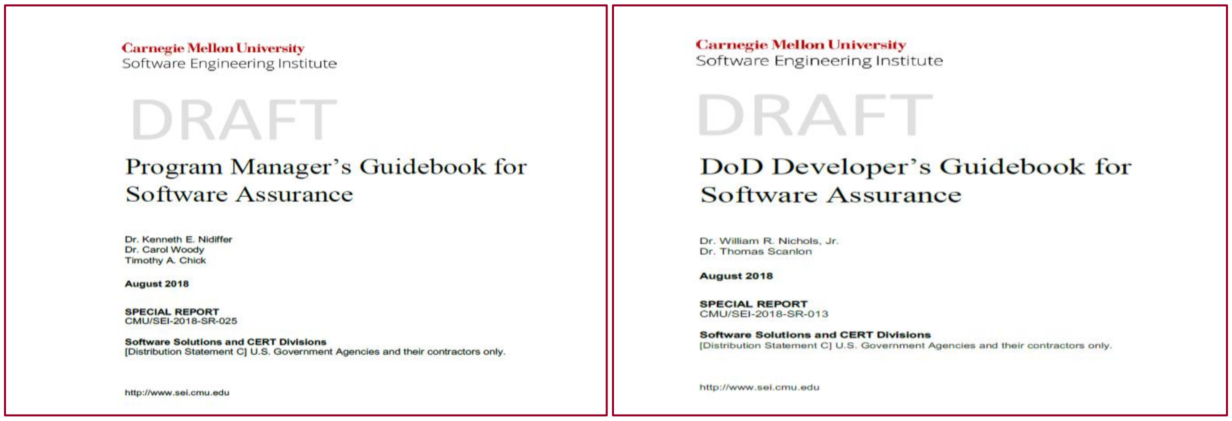
QUANTERION SOLUTIONS INCORPORATED

266 Genesee St.
Utica, NY 13502

TABLE OF CONTENTS

EXECUTIVE SUMMARY

On 7 Aug 2018, JFAC, DAU, SEI and CSIAC held a joint Cyber Experiment (CYBEX) to address SwA concerns and provide initial feedback on two draft SwA guidebooks, one Program Manager (PM) and the other, Developer focused, developed by the Software Engineering Institute (SEI) of Carnegie Mellon University (see Figure 1).



Carnegie Mellon University
Software Engineering Institute

DRAFT

Program Manager's Guidebook for Software Assurance

Dr. Kenneth E. Nidiffer
Dr. Carol Woody
Timothy A. Chick

August 2018

SPECIAL REPORT
CMU/SEI-2018-SR-025

Software Solutions and CERT Divisions
[Distribution Statement C] U.S. Government Agencies and their contractors only.

http://www.sei.cmu.edu

Carnegie Mellon University
Software Engineering Institute

DRAFT

DoD Developer's Guidebook for Software Assurance

Dr. William R. Nichols, Jr.
Dr. Thomas Scanlon

August 2018

SPECIAL REPORT
CMU/SEI-2018-SR-013

Software Solutions and CERT Divisions
[Distribution Statement C] U.S. Government Agencies and their contractors only.

http://www.sei.cmu.edu

Figure 1: PM and Developer Guidebooks for SwA

The one-day event allowed time for field experts to voice SwA related concerns in the overall acquisition process, both for now and in the future. The 30 members were then broken up into three diverse teams to provide feedback on the guidebooks to SEI prior to final editing and submission to the DoD for acceptance on 31 Aug 2018.

The below report details key concerns discussed which included addressing/bringing back foundational software/system engineering in order to address the root of fundamental SwA issues and adopting common language in the areas of functionality and risk in order to identify issues early and balance/trade off those issues through normalized PM and system developer practices to ensure a resilient capability. The report also has several concerns of getting ahead of technology problems both in development, and the technology itself, especially as Agile concepts rapidly become critical for DoD to achieve technological dominance over its adversaries.

BACKGROUND

In order to provide multiple perspectives to the experiment, many participants were invited across several career fields. On the day of the experiment, in addition to CSIAC facilitators, 28 acquisition professionals attended ranging from Government DoD SwA professionals to technical POCs in academia, the Defense Industrial Base (DIB), non-traditional DIB companies, as well as major Information Technology (IT) companies (Figure 2).

CSIAC collaborated with JFAC, DAU and SEI to develop a one-day Cyber Experiment (CYBEX) which was executed on 7 Aug 2018. The goal of this CYBEX was the following:

- Identify software assurance principles that must be considered throughout the acquisition lifecycle
- Provide feedback for both the Program Manager as well as the Developer Software Assurance Guidebooks developed by the Software Engineering Institute (SEI)

After careful consideration, coordination and planning, the goals of this CYBEX included the following:

- Given a notional system and a targeted acquisition phase (Engineering, Manufacturing, and Development (EMD), walk through using the PM and Developer guidebooks to provide recommended edits for them prior to final turn in to the DoD for acceptance on 31 Aug 2018
- Inputs from three separate teams were consolidated and provided to SEI guidebook authors in order to consider changes prior to submission to the DoD for acceptance on 31 Aug 2018

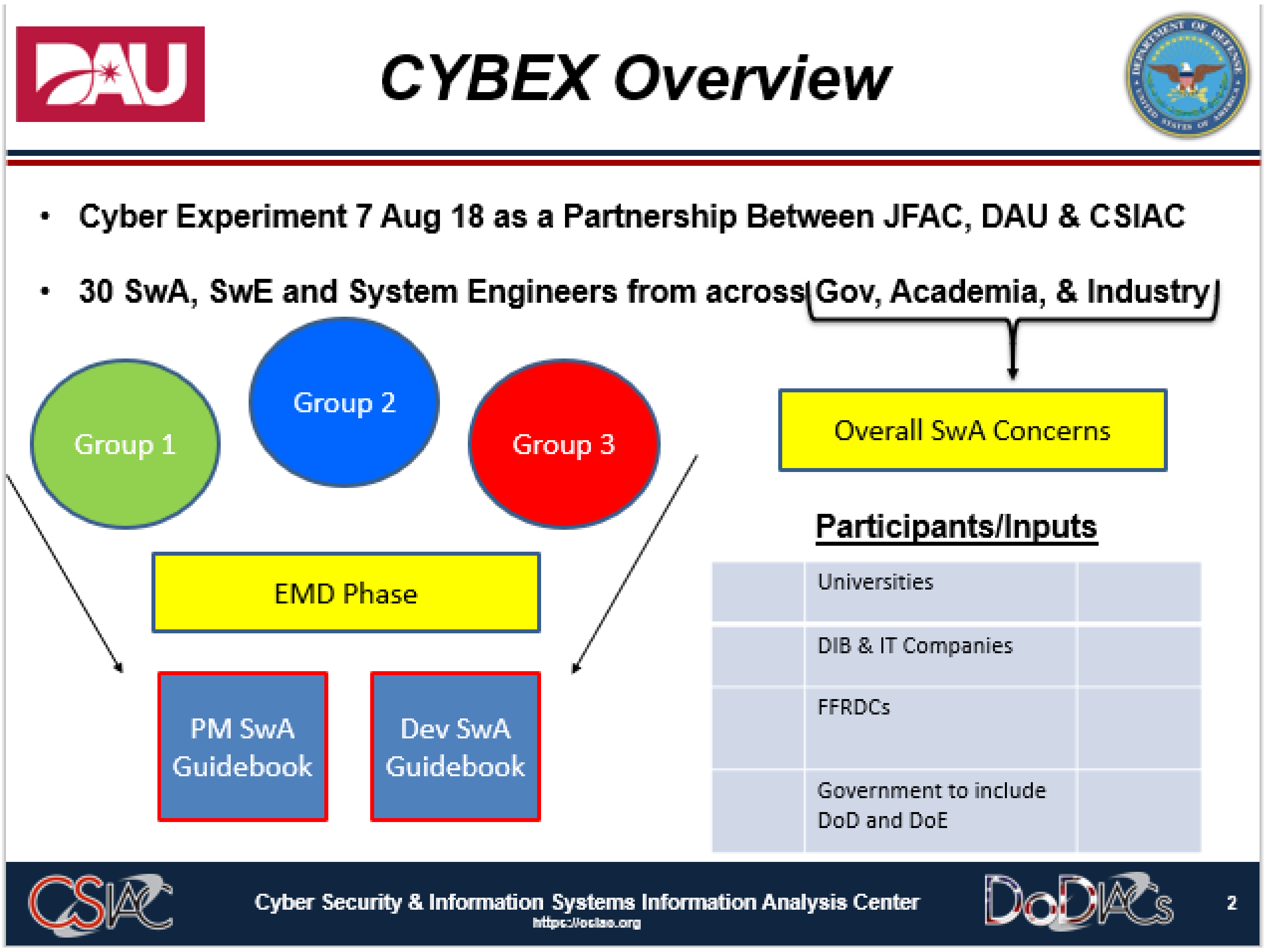
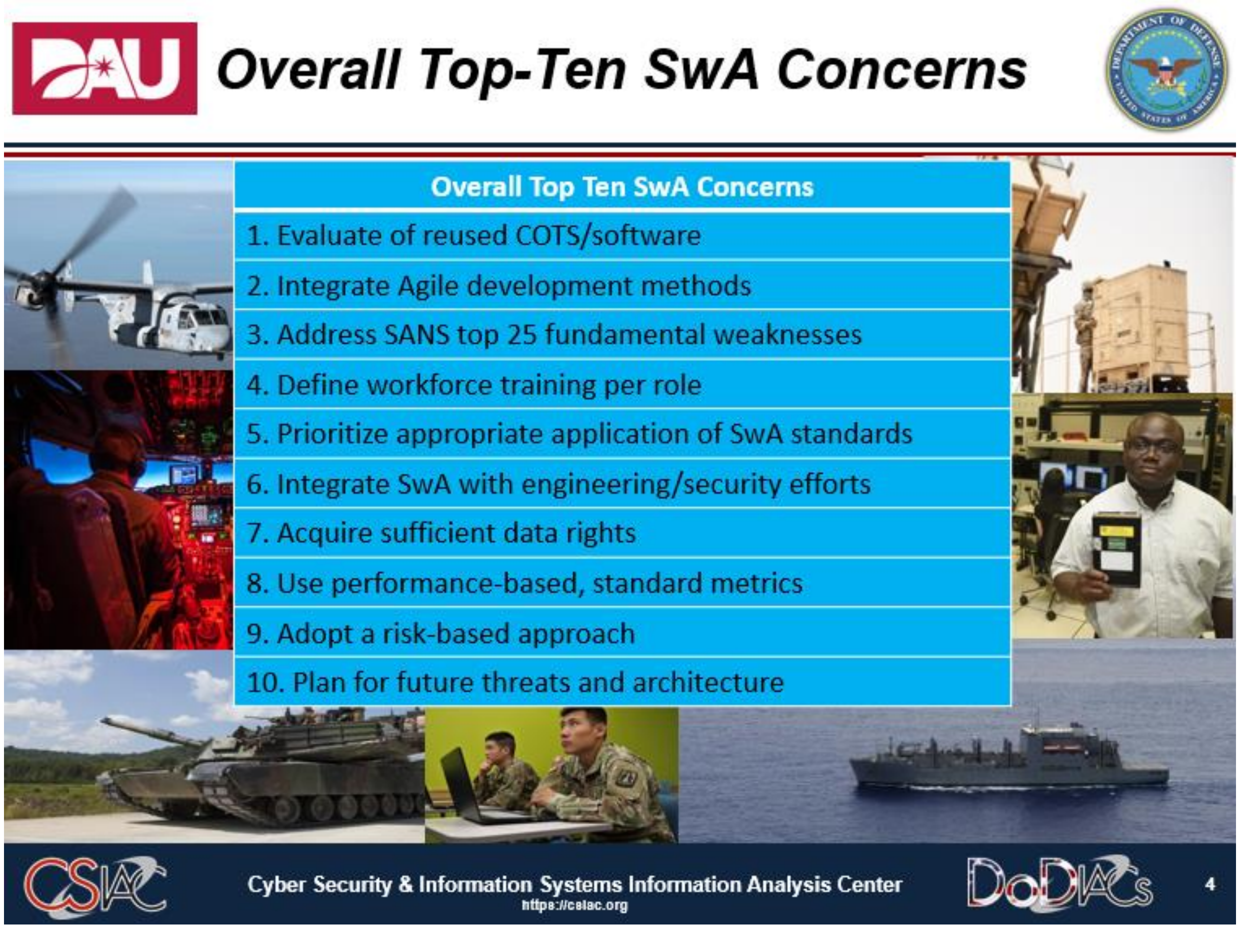

DAU
CYBEX Overview
• Cyber Experiment 7 Aug 18 as a Partnership Between JFAC, DAU & CSIAC
• 30 SwA, SwE and System Engineers from across Gov, Academia, & Industry
Group 1
Group 2
Group 3
Overall SwA Concerns
EMD Phase
PM SwA Guidebook
Dev SwA Guidebook
Participants/Inputs
Universities
DIB & IT Companies
FFRDCs
Government to include DoD and DoE
CSIAC
Cyber Security & Information Systems Information Analysis Center
https://csiac.org
DoDIACs
2

Figure 2: CYBEX Overview/Participants

KEY FINDINGS/CONCERNS

In addition to evaluating the guidebooks, the overall purpose of this CYBEX was to facilitate an environment for a group of acquisition SMEs to provide SwA related issues in the overall acquisition process that they have observed in the field. Below are the key findings that should be considered during future SwA related development:



DAU Overall Top-Ten SwA Concerns

Overall Top Ten SwA Concerns

1. Evaluate of reused COTS/software
2. Integrate Agile development methods
3. Address SANS top 25 fundamental weaknesses
4. Define workforce training per role
5. Prioritize appropriate application of SwA standards
6. Integrate SwA with engineering/security efforts
7. Acquire sufficient data rights
8. Use performance-based, standard metrics
9. Adopt a risk-based approach
10. Plan for future threats and architecture

Cyber Security & Information Systems Information Analysis Center https://csiac.org 4

Figure 3: Top Ten SwA Concerns

1. **Evaluate reused COTS/software** - Ensure reused software is considered for evaluation as well because the software may have existing, yet-to-be-discovered issues, especially when using in a new context. Unsecure SW reuse and insecure architectures are continuing in developing systems due to traditional functionality and up-front cost-savings priority. Often times, primes initially come back and say they are going to reuse previous software to produce software based on the new requirements. Usually, the software turns out to be unsuitable for the new product and causes planning problems down the line in extended time, cost, etc. One solution is to have a better software analysis early in the effort by key software engineers to determine feasibility of reuse during the initial planning stages. This will lead to higher correctness and less cost/schedule delays.
    a. **Suggested Software Reuse Best Practice References** – Recommend including references to the Data Item Description (DID) titled "Software Reuse Plan and Analysis" in order to

address software reuse guidance/best practices. (http://quicksearch.dla.mil/qsDocDetails.aspx?ident_number=281623)

2. **Integrate Agile development methods** - Need to address how to integrate with more rapid and incremental agile technical reviews vice traditional System Engineering Technical Review (SETR) phased Preliminary Design Review (PDR)/Critical Design Review (CDR) methodology, however, you can't skip critical parts. They must be combined in a smart, effective way.
    a. **Agile Coverage Concerns** – Some acquisition guidance has been relieved of the traditional acquisition cycle constraints to follow more agile concepts. However, it is still necessary to keep the effective structures aligned with traditional decision points and milestones in order to be successful. One example was to discuss what a CDR would look like from an Agile perspective. The use of Sprints and how they would come into play were mentioned and there is emerging discussion on how milestones such as CDRs could be addressed in Agile formats while still having an overall system focus and meeting the overall acquisition cycle phases of DODI 5000.02. CDRs and TRRs are often years in the making and can lose focus on intended goals. Agile processes would help with this issue, but must be integrated to meet current acquisition structure in order to stay on track. This issue could be addressed by including periodic software reviews as part of the System Engineering Process (SEP), as well as using periodic Integration Readiness Reviews (IRRs) or just incremental reviews every so many sprints to evaluate readiness/assurance of the supporting software. DODI 5000.02 seems to allow for these additional reviews.
    b. **Aligning Agile and Acquisition Language** – Again, upon the advent of the DoD turning to Agile-type practices, it would be advisable to help relate Agile language with traditional acquisition processes. This will allow for the most effective use of both concepts to quickly build resilient capabilities.
    c. **More About Language is Key** – SwA requirements must be talked about to System Engineers and PMs in the language in which they are disciplined to function in which is safety, reliability, survivability, etc. This will allow software/system engineers with SwA concerns to not only more effectively communicate the "why" to the PM and others, but also help the PM when it comes to making key trade-offs that inevitably occur when constrained by time and budgets.
3. **Address SANS top 25 fundamental weaknesses** - SANS top 25 SwA weaknesses show that universal requirements exist (i.e., need to prevent SQL injection or buffer overflow), but not if there is an unacceptable system engineering performance tradeoff - need to address this tradeoff/risk.
    a. The SANS top 25 has been the same for years! If you create a vulnerability through poor development practices you are likely to do it again as there is little feedback /accountability.
    b. Tune up the tools to help them. Suppose we put in tools in the compliers to help developers get that feedback.
    c. Use most secure process, even if it doesn't need to be secure as it will probably be reused on something more secure in the future. Fix it now to save time and risk.
    d. 99% of new software is based on something that exists today
    e. Learn how to read someone else's code
    f. Keep diversity of creativity while addressing the same terms

i. **Quarter Story**: Given an assignment to design something. Show them a quarter and ask what it is. If you get it right, you get 100, if you don't you fail. Possibly killing creativity of being a table leveler as opposed to the intuitive quarter. So, when developers turn software into something else, how can we keep it from being turned against us?

g. How do you do architecture design testing, microservices, etc.?
h. How do we incentivize fixing software flaws/vulnerabilities in the first place?
i. Automation
j. Line-by-line check failure: Limited ability to "write bad code"
k. Add signatures to code base (to add accountability)
l. Training/Education is key to fix root causes
m. Get back to roots - SwA is just effective software engineering

4. **Define workforce training per role** - Workforce training is required. Must train workforce to develop secure code in the first place.
   a. **Staffing** – Few qualified to really understand/do SwA; Venn diagram showing math/logic, CompSci, and security overlap – probably less than 1,000 available in a very large group of software professionals (military source, original from participant) that fall in the overlap of all three. Sub-category of clearances came up multiple times; in total number of cleared developers, need for more cleared developers in particular instances (ability to see higher-classification shortfalls in order to remedy). Workforce development as linchpin to making SwA 'better' brought up multiple times.
   b. **Teaming Requirements/Best Practices** – Suggested inclusion in different teaming strategies to include "Four Pillars for Effective, Engaged Teams (ex. https://cvdl.ben.edu/blog/4-pillars-effective-teams/) concepts in order to best engage team members across the spectrum to integrate system/software engineering strategies which will ultimately result in more resilient capabilities.
5. **Prioritize appropriate application of SwA standards** – This should include the Cyber Risk Assessment (CRA) – These similar risk-based approaches may be helpful in the development process in order to organize based on threat, as well as be able to draw priorities from it. PMs must use criticality analysis to focus in on what is important to include second, third order effects of training, SwA or other seemingly less important issues.
6. **Integrate SwA with engineering/security efforts -** SwA is not integrated with all security or germane engineering efforts. Need to show how to integrate with all program protection efforts, as well as all systems engineering efforts to achieve Return on Investment (ROI) in reliability, maintainability, safety, performance, etc. Great SwE equals great SwA! Synergistically applied engineering components/phases will produce more secure software/systems.
7. **Acquire sufficient data rights** - PMs didn't/can't get rights, with the goal of providing an opportunity for the PMO to perform independent evaluations. COTS evaluation methods issue is that the Government, or at least an independent third party, may not have access to the actual source code in order to evaluate for vulnerabilities. Recent agreements have occurred between the Government and industry where industry temporarily allows the Government/Customer access to the source code for vulnerability evaluations only. This has resulted in great cooperation between Government and Industry with benefits to both to include regular revisits, but issues still persist.

8. **Use performance-based, standard metrics -** i.e., Risk Management Framework (RMF), Critical Program Information (CPI) identification, Supply Chain Risk Management (SCRM) Criticality Analysis, and Cyber Survivability are weak, much less for traceability to other measures such as performance, reliability, maintainability, etc. A measurement for the tests themselves such as, “What percent has been tested?” or different tools/standards used could be helpful. Also, possibly measure the likelihood of exploitation by looking at the overall system load itself. Need to guide PMs on evaluating questions such as, “What is the quality of your metrics and tools?” “Are these metrics providing the information you need to effectively manage the risk?”
9. **Adopt a risk-based approach** - Need to prioritize SwA tasks and phasing - can't do everything - it would be cost and performance prohibitive.
    a. **Risk scoring suggestions** – In addition to the risk scoring ideas presented, to possibly develop a normalized scoring system that would help the PM, in general, monitor and be alerted to certain factors and engage when the relative risk starts to rise above other concerns.
    b. **Transfer of risk to sustainment side** – In discussions on how risk is measured and remediation planned as a capability transitions to the sustainment phase, these usually materialize in the form of a Plan of Action and Milestones (POA&M). However, including best practices as to how to best ensure these risks are mitigated would be helpful for this audience to include how to get it funded in the near future.
    c. **Resiliency Considerations** - The goal has to be to quickly build systems that we can use in a contested environment. We are currently building systems that operate well when not under stress, but fail in adversarial environments – why???
10. **Plan for future threats and architecture -** Bandwidth environment to effectively catch up with a rapidly changing problem set. Current Intel methods are not relevant or timely enough. MITRE and SANS open source SW weakness data is better than Intel reports - guidebooks need to address this limitation.
    a. **Considerations of the IT Development Infrastructure** – Consider not only the software products themselves, but the infrastructure that is used to securely design, build, test and possibly (given Agile considerations) to deploy/sustain. This is very important to consider because as the infrastructure ages and the requirements migrate, it is possible to incur a significant amount of “Technical Debt” where the cost of not upgrading your infrastructure results in more and more inefficiencies until developers/engineers are spending more time keeping the infrastructure going and dealing with inefficiencies than developing the capabilities themselves. This is also important to be visible when performing long-term planning of infrastructure and proving the above claim using carefully selected and collected metrics that will help keep ahead of this problem.

ADDITIONAL SWA CONCERNS/DETAILS

The participants had a number of initial concerns that were expressed both during the introductions, as well as throughout the experiment. A summary of the issues discussed is provided (see Table 1):

Table 1: Main SwA Concerns

Additional SwA Concerns
1. Requirements: Pre-, EMD- and Post-requirements hard to maintain consistency, coherence; not taking advantage of available standards in defining. Discussion about the current work to standardize security classification guides (SCG) as reasonable path ahead for that arena. Also, discussion about how to apply SCGs to COTS and custom code. Non-standard models and sim (and non-compatible approaches) keep from uniform progress. Trade off analysis limits visibility of SwA across larger requirements group (gets lost). Attacks/vulnerabilities could be used to inform requirements, if there were “room” in contracts to articulate. Generally agreed lack of scope for SwA in requirements (bumping up against desire for ‘agile’ approaches to contracting which limit requirements as too ‘restrictive’). Path could be from Program Protection Plan, to real requirements, to contract language.
2. Ensure Contract Guide is Consistent with SwA Guides: As both the PM and DEV guides are highly deconflicted, ensure that the in-draft contract wording guide is also in sync. By initial looks this appears to be the case. This also comes into play to help set requirements early, so contractors can prepare by hiring, partnering, etc. to meet the goals in a timely manner.
3. Variation in SwA ‘profiles’, even inside a single PMO; non-standard: Need to identify way ahead for more consistent execution of the SwA skills across personnel (Senior Systems Engineer (SSE), Systems Engineer (SE), PM, etc.). Related to workforce skills, but also about how to normalize the SwA skillset so that it can be promulgated across assigned personnel in different size/value projects. Possibly develop standard based on understanding each institutions’ strong points and leveraging accordingly. Software eval tools are requested in the same way. Currently, now coming to us near completion in order to get a snapshot assessment in order to obtain an ATO. Also, how to bring together engineering and RMF experts up front to accomplish an effective security solution?
4. Standards as Separate Topic Area: Standards as currently described/documented are not implementable; how to change? Automation is a large part of SwA activities. Need to articulate better how to use automation in SwA across all areas (testing at all levels, across all types of code) must be observable, measurable, repeatable and transparent (like size, weight and power (SWAP) standards). Contracting language is key to keeping intellectual property from leaving the Government.
5. SwA Normalization: SwA must be put into normalized quality, safety and performance terms in order to help the PM understand what is being traded in/off.
6. Speed of adoption of standards: Hard to keep up with software development activities as the field changes (methods of coding, methods of managing coding) with stable standards, especially if they are CAC enabled, etc. Needs to be easier for developers to come into Government spaces and cooperate or innovation will suffer. Need to plan for the future instead of fixing the past which means anticipating exponential scale increase in data and connectivity. Have to keep in mind that the final result will be used by soldiers; a good way to focus. Training SwA standards as well as

	Additional SwA Concerns
	SwA skills difficult – DAU to collaborate (260?). General agreement that SwA is not separate from systems engineering; engineer in assurance as best practice, helps overall. Software Quality not equivalent to SwA or security, but all mutually beneficial. “SwA Competency” being an attribute with all other processes to be successful.
7.	**Motivating Others to Care About SwA issues:** Stories of finding vulnerabilities, but lack of will to fix them. Also, SwA approaches have to be tailored to the program. However, there tends to be a lack of acceptance for such tailoring. One idea is once quality/performance standards are defined, relevance can be argued for each control based on a common understanding.
8.	**“Architecture is a Big Deal”:** In general, through discussions, the general idea was, “You can’t fix bad design.” Design pushes requirements thought further to the front/left. Work with PM to understand consequence of bad design/lack of requirements – impact of vulnerabilities clearly stated. Possibility of stopping compliance activities later (RMF), tailoring is not turnkey/automatic; environments used in early design ‘virtual’ and not representative of the end state. Need to look closer at SwA impact… If good system design is used up front, then SwA will be inherent or at least be easier to achieve.
9.	**“Guidance at the Speed of Relevance”:** Some discussion of providing 5/50/500 (based on the number of minutes available to view) method for getting more quickly through the basic understanding/steps, diving deeper when necessary. Agile development CAN’T mean tailoring out SwA or skipping requirements!!! (misunderstood by many).
10.	**Impact of AI, Machine Learning, Microservices:** Will impact what we do in SwA (goes back to keeping up with technology, standards and approaches to ensure proper SwA and other -ilities). If we are risk-averse across the PM and SE SSE field, we will continue with “frozen middle” where mid-level PM/SE/SSE will resist positive change to save career/status.
11.	**DIACAP Force fit into RMF:** One participant identifies that we are not really doing RMF, but the previous Department of Defense Information Assurance Certification Accreditation Process (DIACAP) by force-fitting it into RMF documents/artifacts.
12.	**Engineer in SwA with effective Software Engineering (SwE):** Not just “works as designed”, but must be joined at the hip with “free of vulnerabilities.” Discussion of the Fortify (maps cybersecurity issues to quality characteristics) weakness maps and other approaches could prevent potential problem areas early. Need to be able to effectively explain to the PM that if you take care of these technical issues, you will also take care of the related cybersecurity/SwA issues, possibly using Fortify as the evidence. Industry investment in past years has resulted in more resilient medical devices. They saw that the secure, compliant products would prevail and invested early. Discussion of commercial standards in different fields (including medical) have resulted in positive SwA/security results (referred to Motor Industry Software Reliability Association (MISRA)) (and ISO for Risk Management). “Why break something that you don’t have time to fix later on?” This is the idea of “skin in the game” motivation. For example, a

Additional SwA Concerns	
	developer was also the operator/maintainer which drove him to use agile-type testing that eliminated many issues and improved ease of operation before they became problems.
13.	**It is NOT the case that SwA and code quality can't be done:** It is the political/program will that is missing. Comments about the U.S. being the only entity across multiple countries that still has SQL injection problems (specific exemplar). Would like to find a quicker way to get beyond asking a question and when the answer is "I don't know," you can't do it. How do we balance performance/schedule with risk to fail fast, etc.? Why don't we manage by metrics? Metrics are often not used.
14.	**Program Protection Planning Course** at DAU very applicable to many of these issues.
15.	**Threat Intel Failures:** Threat intel doesn't get to developers; hard to teach how to eliminate vulnerabilities if developers/teachers don't know about them. However, if fundamental issues are addressed (buffer overflow, SQL injection, etc.) it would prevent many downstream vulnerabilities.
16.	**SwA Overlays:** Discussion covering ideas about SwA overlays as a more consistent way to enforce/codify (tactical, Modeling & Simulation (M&S), enterprise)
17.	**Agency-wide SwA Policy Development Lessons Learned:** One participant provided an overview of a multi-year effort in their agency to develop an effective SwA program to improve the integrity of operational software and minimize risk by identifying and mitigating software vulnerabilities before fielding. They've done this by taking 19 key controls from the Risk Management Framework (RMF) (pending final director signature) to manage, assess, and build, using the standard system engineering "V" process. There are many requests to leverage this process once approved.
18.	**How do you get the latest cybersecurity requirements rolled out to the contractors that maintain legacy systems?** - PMs to do soon within cost/schedule - What parts would be optional? What parts are Mandatory? (drives action) - Vulnerabilities that are found may be risk managed instead of fixed based on certain metrics
19.	**Vulnerabilities don't get to developers, further discussion on Vulnerability Equities Process:** Open source vulnerabilities seem to rarely result in patching friendly systems.
20.	**Integrate "ilities":** We should consider starting to lump together -ilities (reliability, maintainability, survivability, etc.) together to better present impact, make the -ilities tie much closer to the functional requirements (they now impact); consider thinking about "integrated security" that ties together SwA, SCRM, Cyber, Acquisition Team (AT), etc..
21.	**Definition of SwA (part of system engineering) and how it relates to the rest:** Partnering with the Red Teams and the SwA SMEs to do SwA at the Table Top phases to include RMF control best practices. Also, need dependencies in addition to code in order to do more effective testing. Need clear criteria to judge SwA.

Additional SwA Concerns
22. PM Guide Level of Detail: Retain focus on the fact that the PM doesn't need to know how to do all functions, but should understand them in order to bring them in/accomplish them at the appropriate times. This involves training PMs to integrate SwA in order to ask intelligent questions.
23. Firmware Responsibilities/Standards: Field Programable Gate Array (FPGA), Application-Specific Integrated Circuit (ASIC) guidance is an issue. There is little guidance and established SwA best practices on developing/integrating firmware into the overall acquisition process as it continues to increase in proliferation and potential supply chain/code vulnerabilities.
24. The Definition of "Remediate": Ensure the definition of "remediate" is clear to include that not all issues may be totally eliminated based on risk/cost analysis.
25. Consider leveraging other doctrine: Guidance on models that bring the What, When, How and Who Together – It was suggested that referencing guides such as the IEEE 330-198, which has been superseded by "ISO/IEC/IEEE 29148:2011 (E) – ISO/IEC/IEEE International Standard – Systems and Software Engineering – Life Cycle Processes – Requirements engineering" (https://standards.ieee.org/findstds/standard/29148-2011.html), be considered when for SwA issues.
26. Program Protection Plan (PPP) Reference Recommendation: Another reference that may be helpful is what NASA is currently using NSA-STD-8739.8 w/Change 1 (https://standards.nasa.gov/standard/nasa/nasa-std-87398), based on quality, safety, reliability that may be able to apply to the PPP.
27. Detecting the "Unknown Unknowns": As by definition this process is difficult, but perhaps the community could explore/reference some of the latest research to include Artificial Intelligence/Machine Learning (AI/ML) and related continuous monitoring methods such as behavioral analysis to help develop strategies ranging from the development infrastructure to the production environment to the operational software.

CONCLUSIONS

Overall this one-day event allowed for acquisition professionals from several backgrounds to voice and discuss SwA concerns in addition to providing field feedback for incoming PM and Developer Guidebooks. The anticipated hope is that this information can continue to be proliferated via this report in order to widen discussion and approach solutions to these hard problems.