Holistic Cybersecurity for the DAF: Transforming Cybersecurity & Cyber Defense

Quantum Security and Secured Software Development Framework (SSDF)

August 28, 2023

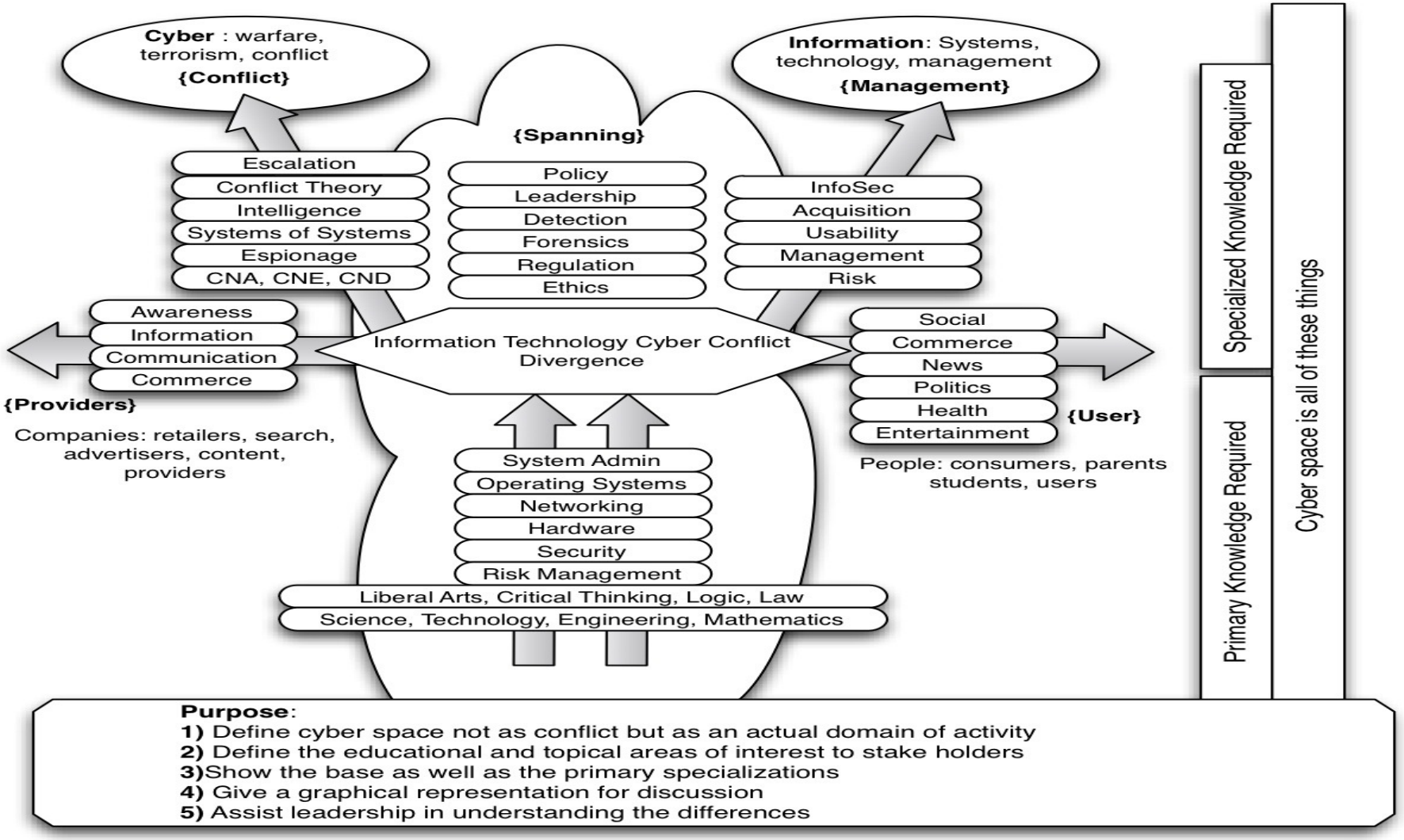
by

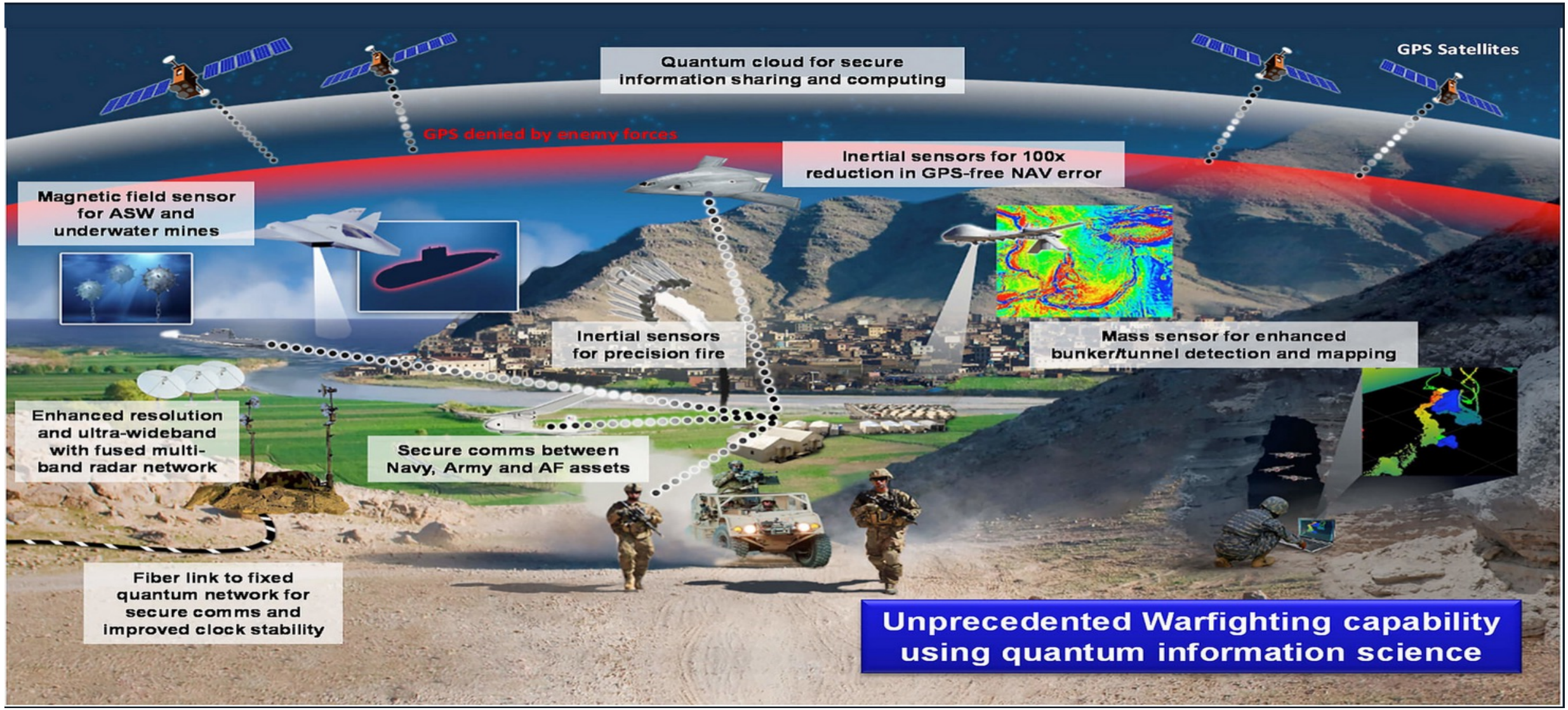
Dr. Merrick S. Watchorn, DMIST
Program Chair, Quantum Security Alliance

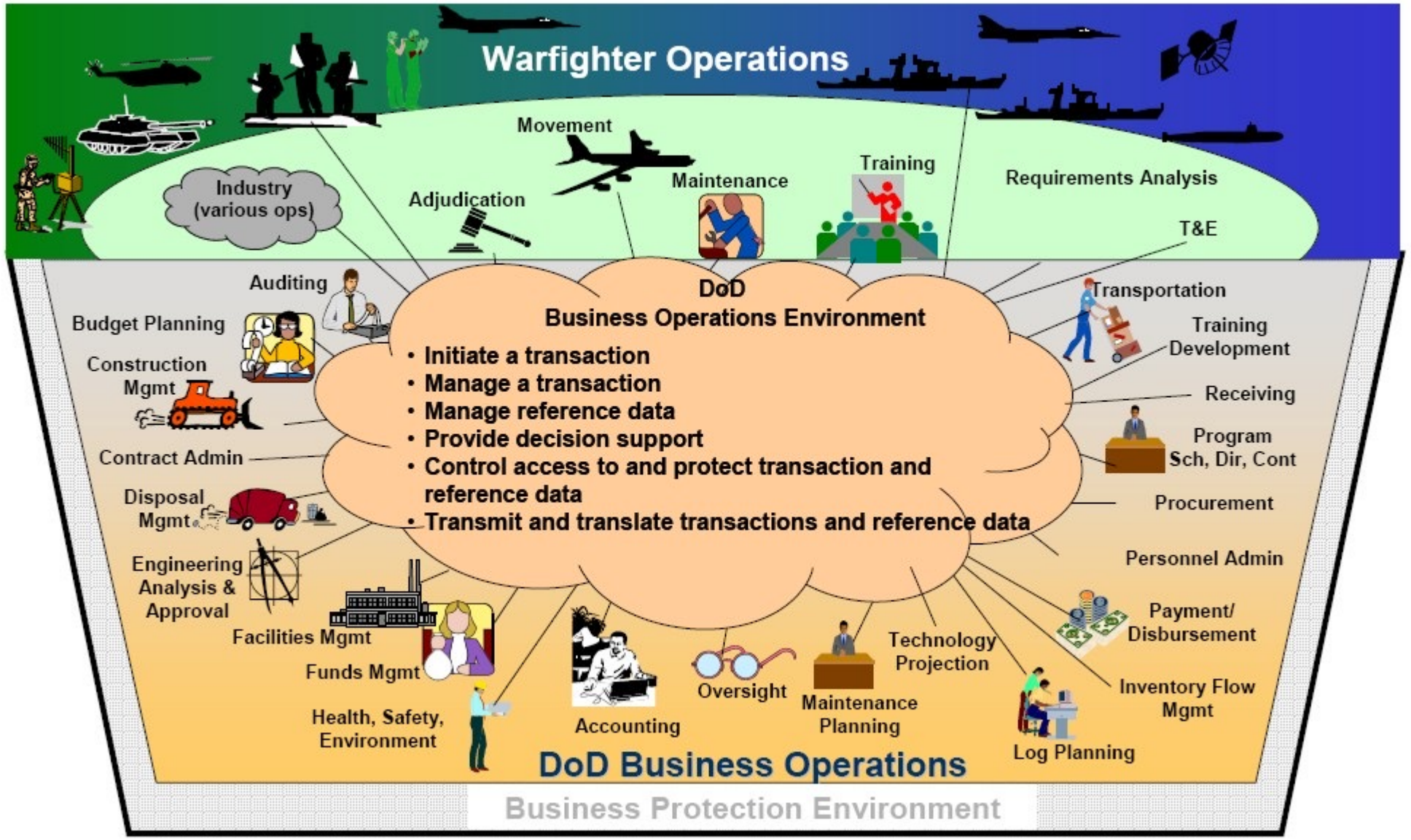
Executive Summary:

The US and its Allies have endured a sustained effort to reduce national security, resiliency, and confidence and undermine infrastructure found within Digital Warfare Strategies espoused by its enemies. This continuous strain has incurred a strategic financial, technical, and workforce debt not seen before in the cyber domain. Cloud computing enabled distributed computing at an economically affordable scale to commerce and the first integration of quantum and cloud with its success. When cyber adversaries have access to the power of quantum computing, our modern crypto systems based on public keys will break (NIST, 2021). The exploration of the known knowledge on Python strengths and weaknesses shall be explored using the NIST SP 800-218 also known as the Secured Software Development Framework (SSDF). Attendees, will learn how to evaluate the pros and cons for quantum adoption from a software perspective and begin the process of designing security policies to begin the mitigation process.

Cybersecurity Practitioner Perspective







Background:

A scripting or script language is a programming language for a special run-time environment that automates the execution of tasks; the tasks could alternatively be executed one-by-one by a human operator. Scripting languages are often interpreted, rather than compiled. Primitives are usually the elementary tasks or API calls, and the language allows them to be combined into more programs. Environments that can be automated through scripting include software applications, web pages within a web browser, usage of the shells of operating systems (OS), embedded systems, as well as numerous games. A scripting language can be viewed as a domain-specific language for an environment; in the case of scripting an application, it is also known as an extension language. Scripting languages are also sometimes referred to as very high-level programming languages, as they operate at a high level of abstraction, or as control languages, particularly for job control languages on mainframes. The term scripting language is also used loosely to refer to dynamic high-level general-purpose languages, such as Perl, PowerShell, Python, and Tcl with the term script often used for small programs (up to a few thousand lines of code) in such languages, or in domain-specific languages such as the text-processing languages sed and AWK. Some of these languages were originally developed for use within an environment, and later developed into portable domain-specific or general-purpose languages. Conversely, many general-purpose languages have dialects that are used as scripting languages.

Reported Security Vulnerabilities													
Keyword	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020
Perl	3,983.00	5,326.00	97.00	39.00	7.00	0.00	1.00	0.00	2.00	159.00	3.00	13.00	5.00
PHP													
PowerShell	0.00	40.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Python	0.00	353.00	42.00	1.00	0.00	0.00	0.00	0.00	0.00	4.00	1.00	0.00	0.00
Tcl	0.00	0.00	350.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	3,983.00	5,719.00	492.00	40.00	7.00	1.00	1.00	0.00	2.00	163.00	4.00	13.00	5.00

Temporal Analysis of Scripting Languages Vulnerabilities

Within the Common Vulnerabilities and Exposures / National Vulnerability Database (NVD) are three different versions on the Common Vulnerability Reporting Framework, 2.0, 3.0 and 3.1 each has a different focus and provides insight into how the temporal mathematics are used to score the risk associated with each vulnerability as report, and investigated. The Common Vulnerability Scoring System (CVSS) is a free and open industry standard for assessing the severity of computer system security vulnerabilities. CVSS attempts to assign severity scores to vulnerabilities, allowing responders to prioritize responses and resources according to threat. Scores are calculated based on a formula that depends on several metrics that approximate ease of exploit and the impact of exploit. Scores range from 0 to 10, with 10 being the most severe. While many utilize only the CVSS Base score for determining severity, temporal and environmental scores also exist, to factor in availability of mitigations and how widespread vulnerable systems are within an organization, respectively. The current version of CVSS (v3.1), which was released in June 2019.

When creating the next analysis section of this report, a formula was crafted to look at both the CVSS Version 2.0 and 3.1 to determine, if analysis could be performed on the distinct data sets in their natural states. The research discovered different terms were used between the versions, in 3.1 the use of Critical was used to describe a vulnerability, while the keyword High was used in the CVSS v2.0, additionally, the CVSS v3.0 was also discovered within the dataset and was accounted for within the formula. The following table represent the information discovered.

Keyword	Perl							
		CVSS v2.0 - Score Analysis				CVSS v3.1 - Score Analysis			
	Year	Low	Medium	High		Low	Medium	High	Critical
	1997								
	1998								
	1999								
	2000								
	2001								
	2002								
	2003								
	2004								
	2005								
	2006								
	2007								
	2008	135.00	1,714.00	2,023.00					1.00
	2009	178.00	2,572.00	2,534.00			1.00	1.00	
	2010	8.00	43.00	45.00					
	2011	4.00	29.00	4.00					
	2012	1.00	4.00	2.00					
	2013								
	2014		2.00						
	2015								
	2016	1.00	1.00				2.00		
	2017		2.00				2.00		3.00
	2018	1.00					1.00		
	2019	3.00	8.00	2.00		2.00	4.00	5.00	2.00
	2020		2.00	2.00			1.00	1.00	1.00

History of Python:

An interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented, and functional programming. Python is often described as a batteries included language due to its comprehensive standard library. Python was conceived in the late 1980s as a successor to the ABC language. Python 2.0, released in 2000, introduced features like list comprehensions and a garbage collection system with reference counting. Python 3.0, released in 2008, was a major revision of the language that is not completely backward-compatible, and much Python 2 code does not run unmodified on Python 3. The Python 2 language was officially discontinued in 2020 (first planned for 2015), and Python 2.7.18 is the last Python 2.7 release and therefore the last Python 2 release. No more security patches or other improvements will be released for it. With Python 2's end-of-life, only Python 3.5.x and later are supported. Python interpreters are available for many operating systems. A global community of programmers develops and maintains CPython, a free and open-source reference implementation. A non-profit organization, the Python Software Foundation, manages and directs resources for Python and CPython development.

Python – CVSS Scoring

	CVSS v2.0 - Score Analysis				CVSS v3.1 - Score Analysis			
Year	Low	Medium	High		Low	Medium	High	Critical
2002	-	2.00	1.00		-	-	-	-
2003	-	1.00	-		-	-	-	-
2004	-	2.00	1.00		-	-	-	-
2005	1.00	2.00	8.00		-	-	-	-
2006	1.00	1.00	3.00		-	-	-	-
2007	-	4.00	3.00		-	-	-	-
2008	1.00	10.00	12.00		-	-	-	-
2009	-	16.00	6.00		-	-	-	-
2010	-	8.00	4.00		-	-	-	-
2011	-	9.00	6.00		-	-	-	-
2012	3.00	11.00	1.00		-	-	-	-
2013	2.00	13.00	1.00		-	-	-	-
2014	5.00	28.00	12.00		-	-	-	-
2015	1.00	8.00	8.00		-	-	-	-
2016	-	9.00	5.00		-	6.00	7.00	2.00
2017	3.00	17.00	16.00		2.00	12.00	11.00	11.00
2018	4.00	21.00	10.00		1.00	11.00	16.00	7.00
2019	3.00	43.00	29.00		-	14.00	37.00	24.00
2020	4.00	24.00	14.00		1.00	12.00	22.00	7.00

CWE coverage for Python

Critical Vulnerability's for Python – Analysis

Keyword: Python

Vulnerabilities Found: 401

Average Base Impact Score: 7.685 out of 10

Average Exploitability Score: 2.934

Average Impact Score: 4.661

Critical Vulnerabilities Found: 52

Average Base Impact Score: 9.745 out of 10

Average Exploitability Score: 3.9

Average Impact Score: 5.845

Confidentiality: (52 / 52) = 100%

Integrity: (52 / 52) = 100%

Availability: (52 / 52) = 100%

Legal Risk Score: (52 / 401) = 12.967%

CWE	Title	Query name
CWE-20	py/count-untrusted-data-external-api	Frequency counts for external APIs that are used with untrusted data
CWE-20	py/untrusted-data-to-external-api	Untrusted data passed to external API
CWE-20	py/incomplete-hostname-regexp	Incomplete regular expression for hostnames
CWE-20	py/incomplete-url-substring-sanitization	Incomplete URL substring sanitization
CWE-20	py/overly-large-range	Overly permissive regular expression range
CWE-20	py/bad-tag-filter	Bad HTML filtering regexp
CWE-22	py/path-injection	Uncontrolled data used in path expression
CWE-22	py/tarslip	Arbitrary file write during tarfile extraction
CWE-22	py/zipslip	Arbitrary file access during archive extraction ("Zip Slip")
CWE-22	py/tarslip-extended	Arbitrary file write during tarfile extraction
CWE-22	py/unsafe-unpacking	Arbitrary file write during a tarball extraction from a user controlled source
CWE-23	py/path-injection	Uncontrolled data used in path expression
CWE-36	py/path-injection	Uncontrolled data used in path expression
CWE-73	py/path-injection	Uncontrolled data used in path expression
CWE-73	py/shell-command-constructed-from-input	Unsafe shell command constructed from library input
CWE-74	py/path-injection	Uncontrolled data used in path expression
CWE-74	py/command-line-injection	Uncontrolled command line
CWE-74	py/shell-command-constructed-from-input	Unsafe shell command constructed from library input
CWE-74	py/jinja2/autoescape-false	Jinja2 templating with autoescape=False
CWE-74	py/reflective-xss	Reflected server-side cross-site scripting

Cyber Security Framework (CSF) Identify

Develop an organizational understanding to manage cybersecurity risk to systems, assets, data, and capabilities. The activities in the Identify Function are foundational for effective use of the Framework. Understanding the business context, the resources that support critical functions, and the related cybersecurity risks enables and develop an organizational understanding to manage cybersecurity risk to systems, assets, data, and capabilities. The activities in the Identify Function are foundational for effective use of the Framework. Understanding the business context, the resources that support critical functions, and the related cybersecurity risks enables an organization to focus and prioritize its efforts, consistent with its risk management strategy and business needs. Examples of outcome Categories within this Function include Asset Management; Business Environment; Governance; Risk Assessment; Risk Management Strategy; and Supply Chain Risk Management.

Supply Chain Risk Management (ID.SC) - The organization's priorities, constraints, risk tolerances, and assumptions are established and used to support risk decisions associated with managing supply chain risk. The organization has established and implemented the processes to identify, assess and manage supply chain risks.

ID.SC-1 Cyber supply chain risk management processes are identified, established, assessed, managed, and agreed to by organizational stakeholders.

Tier Partial Count: 3.00 NIST SP 800-53 Rev. 4: SA-9; SA-12; PM-9

Tier Risk Informed Count: 43.00 NIST SP 800-53 Rev. 4: AT-3; CA-3; CM-8; CP-2; IR-4; IR-7; PE-3; PE-16; PL-2; PL-8; PS-7; RA-2; RA-3; RA-5; SA-3; SA-4; SA-8; SA-9; SA-10; SA-12; SA-13; SA-14; SA-15; SA-18; SA-19; SA-20; SC-8; SC-13; SC-26; SC-27; SC-29; SC-30; SC-34; SC-38; SI-3; SI-7; SI-14; PM-1; PM-7; PM-8; PM-9; PM-11

Tier Repeatable Count: 187.00 NIST SP 800-53 Rev. 4: AC-2; AC-3; AC-4; AC-5; AC-6; AC-7; AC-8; AC-9; AC-14; AC-16; AC-17; AC-18; AC-19; AC-20; AC-21; AT-1; AT-2; AT-3; AT-4; AU-1; AU-2; AU-3; AU-4; AU-5; AU-6; AU-7; AU-8; AU-9; AU-10; AU-11; AU-12; AU-13; AU-14; AU-16; CA-2; CA-3; CA-5; CA-6; CA-7; CM-2; CM-3; CM-4; CM-5; CM-6; CM-7; CM-8; CM-9; CM-11; CP-2; CP-4; CP-6; CP-7; CP-8; CP-9; CP-10; CP-12; IA-2; IA-3; IA-4; IA-5; IA-7; IA-8; IR-1; IR-2; IR-3; IR-4; IR-5; IR-6; IR-7; IR-8; MA-2; MA-3; MA-4; MA-5; MA-6; MP-1; MP-2; MP-3; MP-4; MP-5; MP-6; MP-7; MP-8; PE-2; PE-3; PE-4; PE-5; PE-6; PE-16; PE-18; PE-19; PL-2; PL-4; PL-7; PL-8; PS-1; PS-2; PS-3; PS-4; PS-5; PS-6; PS-7; PS-8; RA-1; RA-2; RA-3; RA-5; SA-3; SA-4; SA-5; SA-8; SA-9; SA-10; SA-11; SA-12; SA-13; SA-14; SA-15; SA-16; SA-17; SA-18; SA-19; SA-20; SA-21; SC-2; SC-3; SC-4; SC-5; SC-6; SC-7; SC-8; SC-10; SC-12; SC-13; SC-17; SC-22; SC-24; SC-25; SC-26; SC-27; SC-28; SC-29; SC-30; SC-31; SC-34; SC-35; SC-37; SC-38; SC-44; SI-2; SI-3; SI-4; SI-7; SI-11; SI-12; SI-14; PM-1; PM-5; PM-6; PM-7; PM-8; PM-9; PM-11; PM-14; AP-1; AP-2; AR-1; AR-2; AR-3; AR-4; AR-5; AR-6; AR-7; AR-8; DI-1; DI-2; DM-1; DM-2; DM-3; IP-1; SE-1; SE-2; TR-1; TR-2; UL-1; UL-2

Tier Adaptive Count: 210.00 NIST SP 800-53 Rev. 4: AC-1; AC-2; AC-3; AC-4; AC-5; AC-6; AC-7; AC-8; AC-9; AC-10; AC-14; AC-16; AC-17; AC-18; AC-19; AC-20; AC-21; AC-22; AT-1; AT-2; AT-3; AT-4; AU-1; AU-2; AU-3; AU-4; AU-5; AU-6; AU-7; AU-8; AU-9; AU-10; AU-11; AU-12; AU-13; AU-14; AU-16; CA-2; CA-3; CA-5; CA-6; CA-7; CA-9; CM-2; CM-3; CM-4; CM-5; CM-6; CM-7; CM-8; CM-9; CM-10; CM-11; CP-2; CP-3; CP-4; CP-6; CP-7; CP-8; CP-9; CP-10; CP-12; IA-2; IA-3; IA-4; IA-5; IA-7; IA-8; IR-1; IR-2; IR-3; IR-4; IR-5; IR-6; IR-7; IR-8; MA-2; MA-3; MA-4; MA-5; MA-6; MP-1; MP-2; MP-3; MP-4; MP-5; MP-6; MP-7; MP-8; PE-2; PE-3; PE-4; PE-5; PE-6; PE-14; PE-16; PE-17; PE-18; PE-19; PL-2; PL-4; PL-7; PL-8; PS-1; PS-2; PS-3; PS-4; PS-5; PS-6; PS-7; PS-8; RA-1; RA-2; RA-3; RA-5; SA-3; SA-4; SA-5; SA-8; SA-9; SA-10; SA-11; SA-12; SA-13; SA-14; SA-15; SA-16; SA-17; SA-18; SA-19; SA-20; SA-21; SC-2; SC-3; SC-4; SC-5; SC-6; SC-7; SC-8; SC-10; SC-11; SC-12; SC-13; SC-15; SC-16; SC-17; SC-18; SC-19; SC-20; SC-21; SC-22; SC-23; SC-24; SC-25; SC-26; SC-27; SC-28; SC-29; SC-30; SC-31; SC-34; SC-35; SC-37; SC-38; SC-39; SC-43; SC-44; SI-2; SI-3; SI-4; SI-7; SI-10; SI-11; SI-12; SI-14; PM-1; PM-4; PM-5; PM-6; PM-7; PM-8; PM-9; PM-10; PM-11; PM-14; AP-1; AP-2; AR-1; AR-2; AR-3; AR-4; AR-5; AR-6; AR-7; AR-8; DI-1; DI-2; DM-1; DM-2; DM-3; IP-1; IP-2; IP-3; SE-1; SE-2; TR-1; TR-2; UL-1; UL-2

Secure Software Development Framework (SSDF) 1.0 NIST SP 800-218 - Ontology

Prepare the Organization (PO) - Organizations should ensure that their people, processes, and technology are prepared to perform secure software development at the organization level. Many organizations will find some PO practices to also be applicable to subsets of their software development, like individual development groups or projects.

PO.1: Define Security Requirements for Software Development - Ensure that security requirements for software development are known at all times so that they can be taken into account throughout the SDLC and duplication of effort can be minimized because the requirements information can be collected once and shared. This includes requirements from internal sources (e.g., the organization's policies, business objectives, and risk management strategy) and external sources (e.g., applicable laws and regulations).

PO.1.1: Identify and document all security requirements for the organization's software development infrastructures and processes, and maintain the requirements over time. NIST SP 800-53 (Rev. 4): SA-1 System and Services Acquisition Policy and Procedures, SA-8 Security Engineering Principles, SA-15 Development Process, Standards, and Tools

Example(s):

1. Define policies for securing software development infrastructures and their components, including development endpoints, throughout the SDLC and maintaining that security.

2. Define policies for securing software development processes throughout the SDLC and maintaining that security, including for open-source and other third-party software components utilized by software being developed.

3. Review and update security requirements at least annually, or sooner if there are new requirements from internal or external sources, or a major security incident targeting software development infrastructure has occurred.

4. Educate affected individuals on impending changes to requirements.

Secure Software Development Framework (SSDF) 1.0 (NIST SP 800-218) – Python Issues

PO.2.3 - Obtain upper management or authorizing official commitment to secure development and convey that commitment to all with development-related roles and responsibilities.

NIST SP 800-53 Rev 4: SA-9, SA-12, PM-9

PM-9 Risk Management Strategy					
NIST SP 800-53 Rev 3		NIST SP 800-53 Rev 4		NIST SP 800-53 Rev 5	
DoD DISA STIG's (CCI Listing):	Related Control(s):	DoD DISA STIG's (CCI Listing):	Related Control(s):	DoD DISA STIG's (CCI Listing):	Related Control(s):
CCI-000227 - Develop a comprehensive strategy to manage security risk to organizational operations and assets, individuals, other organizations, and the Nation associated with the operation and use of information systems. CCI-000228 - Implement the risk management strategy consistently across the organization.	RA-3	CCI-000227 - Develop a comprehensive strategy to manage security risk to organizational operations and assets, individuals, other organizations, and the Nation associated with the operation and use of information systems. CCI-000228 - Implement the risk management strategy consistently across the organization. CCI-002994 - Review and update the risk management strategy in accordance with organization-defined frequency or as required, to address organizational changes. CCI-002995 - Defines the frequency with which to review and update the risk management strategy to address organizational changes.	RA-3	CCI-000227 - Develop a comprehensive strategy to manage security risk to organizational operations and assets, individuals, other organizations, and the Nation associated with the operation and use of information systems. CCI-000228 - Implement the risk management strategy consistently across the organization. CCI-002994 - Review and update the risk management strategy in accordance with organization-defined frequency or as required, to address organizational changes. CCI-002995 - Defines the frequency with which to review and update the risk management strategy to address organizational changes. CCI-004345 - Develop a comprehensive strategy to manage privacy risk to individuals resulting from the authorized processing of personally identifiable information.	AC-1; AU-1; AT-1; CA-1; CA-2; CA-5; CA-6; CA-7; CM-1; CP-1; IA-1; IR-1; MA-1; MP-1; PE-1; PL-1; PL-2; PM-2; PM-8; PM-18; PM-28; PM-30; PS-1; PT-1; PT-2; PT-3; RA-1; RA-3; RA-9; SA-1; SA-4; SC-1; SC-38; SI-1; SI-12; SR-1; SR-2

	Security Control Analysis (Confidentiality, Integrity, Availability and Legal): AC-1 Policy and Procedures
Related Control(s):	NIST SP 800-53 Rev. 3: PM-9 Risk Management Strategy NIST SP 800-53 Rev. 4: PM-9 Risk Management Strategy AR-4 Privacy Monitoring and Auditing AR-7 Privacy-Enhanced System Design and Development NIST SP 800-53 Rev. 5: IA-1 Policy and Procedures PM-9 Risk Management Strategy PM-24 Data Integrity Board PS-8 Personnel Sanctions SI-12 Information Management and Retention

Security Control Analysis (Confidentiality, Integrity, Availability and Legal): AC-1 Policy and Procedures

LAWS AND EXECUTIVE ORDERS

Cyber-Related Sanctions Regulations (CSR 578)

Health Insurance Portability and Accountability Act: §164.308(a)(3); §164.308(a)(4); §164.312(a)(1)

Critical Infrastructure Protection (CIP)

ID.GV-1 - Organizational information security policy is established. Control(s): -1 controls from all families

ID.GV-3 - Legal and regulatory requirements regarding cybersecurity, including privacy and civil liberties obligations, are understood and managed. Control(s): -1 controls from all families (except PM-1).

PR.AC-1 - Identities and credentials are issued, managed, verified, revoked, and audited for authorized devices, users and processes. Control(s): AC-1, AC-2, IA-1, IA-2, IA-3, IA-4, IA-5, IA-6, IA-7, IA-8, IA-9, IA-10, IA-11.

PR.AC-3 - Remote access is managed. Control(s): AC-1, AC-17, AC-19, AC-20, SC-15

PR.AC-4 - Access permissions and authorizations are managed, incorporating the principles of least privilege and separation of duties. Control(s): AC-1, AC-2, AC-3, AC-5, AC-6, AC-14, AC-16, AC-24.

PR.AC-6 - Identities are proofed and bound to credentials and asserted in interactions when appropriate. Control(s): AC-1, AC-2, AC-3, AC-16, AC-19, AC-24, IA-1, IA-2, IA-4, IA-5, IA-8, PE-2, PS-3.

Security Control Analysis (Confidentiality, Integrity, Availability and Legal):

AC-1 Policy and Procedures

REGULATIONS, DIRECTIVES, PLANS, AND POLICIES

Circular No. A-130 - Managing Information as a Strategic Resource

Main Body § 5(f)(1)(d) - Limit the creation, collection, use, processing, storage, maintenance, dissemination, and disclosure of PII.

Agencies shall limit the creation, collection, use, processing, storage, maintenance, dissemination, and disclosure of PII to that which is legally authorized, relevant, and reasonably deemed necessary for the proper performance of agency functions.

Appendix I § 3(c) - Require entities with which PII is shared to maintain the PII in an information system with a categorization level.

Agencies that share PII shall require, as appropriate, other agencies and entities with which they share PII to maintain the PII in an information system with a NIST FIPS Publication 199 confidentiality impact level, as determined by the agency sharing the PII.

Appendix I § 3(d) - Impose conditions on the creation, collection, use, processing, storage, maintenance, dissemination, disclosure, and disposal of shared PII through agreements.

Agencies that share PII with other agencies or entities shall impose, where appropriate, conditions (including the selection and implementation of particular security and privacy controls) that govern the creation, collection, use, processing, storage, maintenance, dissemination, disclosure, and disposal of the PII through written agreements, including contracts, data use agreements, information exchange agreements, and memoranda of understanding.

Appendix I § 4(j)(2)(b) - Ensure implementation of privacy controls for contractor information systems.

Agencies shall ensure that privacy controls of information systems and services used or operated by contractors or other entities on behalf of the agency are effectively implemented and comply with NIST standards and guidelines and agency requirements.