**DESCRIPTION OF WORK**

|  |  |
| --- | --- |
| **Award/Mod Effective** | **Version Date** |
| {IF awardHistory.length < 1} |  |
| {``} | {``} |
| {END-IF} |  |
| {FOR award IN awardHistory} |  |
| {$award.contractAwardType.toLowerCase().replace(`\_`,` `)} | {$award.effectiveDate} |
| {END-FOR award} |  |

{ EXEC ci = contractInformation }

| Contract Number: | {ci.contractNumber ? ci.contractNumber : `TBD`} |
| --- | --- |
| Task Order (TO) Number: | TBD |
| Contractor Name | TBD |
| Tracking Number: | TBD |
| Follow-on to Previous Contract and TO Number: | {IF !ci.currentContractExists}  N/A  {END-IF}  {IF ci.currentContractExists}  Contract Number: {ci.contractNumber}  Task Order Number: {ci.previousTaskOrderNumber}  {END-IF} |

**1.**  TO Title. {toTitle}

**2.** Scope. {scope}

The information provided in this Description of Work (DoW) is to allow JWCC cloud service providers (CSPs) to propose/quote and the Government to analyze proposals/quotes in a consistent manner. All requirements specified in this DoW are presented as a minimum capability for analysis only and will neither limit the actual Cloud Service Offering(s) (CSO) ordered from the Contractor’s JWCC Catalog during TO performance nor exceed the dollar threshold established within the TO. Note: No CSOs shall be proposed/quoted using reserved instances/pre-paid/committed use.

The Government may require surge capabilities during the base or any option period (OP), and surge requirements will be within the scope of the task order for the defined task areas of this DoW. Surge capabilities over the life of the TO will not exceed {surgeRequirementCapabilities ? surgeRequirementCapacity : `0`}% of the contractor’s total proposed price for the base and all OPs, excluding any six-month extension of services pursuant to Federal Acquisition Regulation (FAR) 52.217-8.

**3.** Background/Current Environment.

{currentEnvironment.currentEnvironmentExists ? `A current environment to rehost exists.` : `A current environment to rehost does not exist.`}

{currentEnvironment.hasSystemDocumentation ? ` System diagrams, data architecture diagrams, charts, or other information exists for the current environment and will be provided.` : ` System diagrams, data architecture diagrams, charts, or other information do not exist`}

{currentEnvironment.hasMigrationDocumentation ? `A migration assessment, analysis, or process has occurred to identify the cloud services and tools needed and will be provided.` : `A migration assessment, analysis, or process has occurred to identify the cloud services and tools needed and will be provided.`}

{IF currentEnvironment.currentEnvironmentExists}

Number of instances (isolated environment, enclave, or collection of components): { sumTotalInstances(currentEnvironment.envInstances)}

{EXEC instance = 1}

{FOR env IN currentEnvironment.envInstances}

**3.{INS instance}** Instance {INS instance}

* Number of instances with the following configurations: {INS $env.numberOfInstances}
* Location of current environment: {INS formatEnum($env.instanceLocation)}

{IF $env.instanceLocation === `ON\_PREMISE`}

{IF $env.instanceLocation === `ON\_PREMISE` && $env.classificationLevel.classification === `U`}

* + Data classification level: Unclassified
  + Type of information being hosted:

{FOR infoType IN $env.classifiedInformationTypes}

* + - {$infoType}

{END-FOR infoType}

{END-IF}

{IF $env.instanceLocation === `ON\_PREMISE` && $env.classificationLevel.classification === `S`}

* + Data classification level: Secret

{END-IF}

{IF $env.instanceLocation === `ON\_PREMISE` && $env.classificationLevel.classification === `TS`}

* + Data classification level: Top Secret (TS)

{END-IF}

{END-IF}

{IF $env.instanceLocation === `CLOUD`}

Regions(s): {IF $env.deployedRegions.length < 1}N/A {END-IF}

{FOR region IN $env.deployedRegions}

* + {$region.name}

{END-FOR region}

{IF $env.instanceLocation === `CLOUD` && $env.classificationLevel.classification === `U`}

Data classification level: Unclassified

Impact level: {formatImpactLevel($env.classificationLevel.impactLevel)}

{END-IF}

{IF $env.instanceLocation === `CLOUD` && $env.classificationLevel.classification === `S`}

Data classification level: Secret

{END-IF}

{IF $env.instanceLocation === `CLOUD` && $env.classificationLevel.classification === `TS`}

Data classification level: Top Secret (TS)

{END-IF}

{END-IF}

* Current usage: {$env.currentUsageDescription ? formatEnum($env.currentUsageDescription) : `N/A`}

{IF $env.currentUsageDescription === `IRREGULAR\_USAGE` && $env.isTrafficSpikeEventBased }

* + Event based: { $env.trafficSpikeEventDescription}

{END-IF}

{IF $env.currentUsageDescription === `IRREGULAR\_USAGE` && $env.isTrafficSpikePeriodBased}

* + High usage periods: {$env.trafficSpikePeriodDescription}

{END-IF}

* Location(s) and approximate number of current users: {IF !$env.usersPerRegion} N/A {END-IF}

{IF $env.usersPerRegion}

{FOR regionUsers IN formatRegionUsers($env.usersPerRegion)}

* + {$regionUsers} users

{END-FOR regionUsers}

{END-IF}

* Approximate number of vCPUs/size of compute: {$env.numberOfVcpus ? $env.numberOfVcpus : `N/A`}

{IF $env.numberOfVcups && $env.processorSpeed}

* + Processor speed: {INS $env.processorSpeed}

{END-IF}

{IF $env.numberOfVcups && $env.operatingSystem}

* + Operating system: { INS $env.operatingSystem}

{END-IF}

* Licensing: {$env.licensing ? $env.licensing : `N/A`}
* Memory ({$env.memoryUnit ? $env.memoryUnit : `GB`}): {$env.memoryAmount ? $env.memoryAmount : `N/A`}
* Current Payment Arrangement: {$env.pricingModel ? `` : `N/A`}

{IF $env.pricingModel === `PREPAID`}

* + Reserved/pre-paid/up-front

{END-IF}

{IF $env.pricingModel === `PAY\_AS\_YOU\_GO`}

* + Pay-as-you-go

{END-IF}

* Reservation period/pre-paid period/up-front period/expiration date: {$env.pricingModel === `PREPAID` ? formatExpirationDate($env.pricingModelExpiration): `N/A`}
* Storage type and current size ({$env.storageUnit ? $env.storageUnit : `GB`}): {IF !$env.storageType} N/A {END-IF}

{IF $env.storageType === `BLOCK`}

* + Block Storage (Fixed-sized raw storage capacity): {$env.storageAmount} {$env.storageUnit}

{END-IF}

{IF $env.storageType === `OBJECT`}

* + Object Storage (Store and serve unstructured user-generated content): {$env.storageAmount} {$env.storageUnit}

{END-IF}

{IF $env.storageType === `FILE`}

* + File Storage (Store and serve shared file systems): {$env.storageAmount} {$env.storageUnit}

{END-IF}

{IF $env.storageType === `ARCHIVE`}

* + Archive Storage (Store and serve for long-term data retention): {$env.storageAmount} {$env.storageUnit}

{END-IF}

* Performance tier: {IF !$env.performanceTier} N/A` {END-IF}

{IF $env.performanceTier === `GENERAL`}

* + General Purpose (Provides a balance of compute, memory & network)

{END-IF}

{IF $env.performanceTier === `COMPUTE`}

* + Compute Optimized (Supports compute-bound applications that benefit from high performance processors)

{END-IF}

{IF $env.performanceTier === `MEMORY`}

* + Memory Optimized (Designed to deliver fast performance for workloads that process large data sets in memory)

{END-IF}

{IF $env.performanceTier === `STORAGE`}

* + Storage Optimized (Designed for high, sequential read and write workloads to very large data sets on local storage)

{END-IF}

* Approximate data/internet egress per month ({$env.dataEgressMonthlyUnit ? $env.dataEgressMonthlyUnit : `GB`}): {$env.dataEgressMonthlyAmount ? $env.dataEgressMonthlyAmount + ` ` + $env.dataEgressMonthlyUnit : `N/A`}
* Additional information: {$env.additionalInformation ? $env.additionalInformation : `N/A`}

{EXEC instance++}

{END-FOR env}

{END-IF}

**4.** Performance Requirements.

{ EXEC liftAndShift = currentEnvironment.currentEnvironmentReplicatedOptimized === `YES\_REPLICATE` ? true : false}

{ EXEC optimize = currentEnvironment.currentEnvironmentReplicatedOptimized === `YES\_OPTIMIZE` ? true : false}

{ EXEC needsArchitecturalDesign = currentEnvironment.needsArchitecturalDesignServices}

{EXEC archDesignReq = currentEnvironment.architecturalDesignRequirement}

{EXEC dowArchDesignReq = architecturalDesignRequirement}

**4.1** Task 1 - Objective-based Requirement.

**4.1.1** {liftAndShift ? ` Require current functions to be replicated (lift & shift) using JWCC offerings.` : `Reserved`}

{IF liftAndShift}

{currentEnvironment.statementReplicatedOptimized}

{currentEnvironment.additionalGrowth ? `Additional growth is anticipated as` + ` ` + currentEnvironment.anticipatedYearlyAdditionalCapacity +`%.` : `Additional growth is not anticipated.`}

{currentEnvironment.hasPhasedApproach ? `A phased approach is required.` : ` A phased approach is not required.`}

{IF currentEnvironment.hasPhasedApproach}

* Schedule: {currentEnvironment.phasedApproachSchedule}

{END-IF}

{END-IF}

**4.1.2** {optimize ? ` Require current functions to be optimized (improved/modernized) using JWCC offerings.` : `Reserved`}

{IF optimize}

{currentEnvironment.statementReplicatedOptimized}

{currentEnvironment.additionalGrowth ? `Additional growth is anticipated as` + ` ` + currentEnvironment.anticipatedYearlyAdditionalCapacity +`%.` : `Additional growth is not anticipated.`}

{currentEnvironment.hasPhasedApproach ? `A phased approach is required.` : ` A phased approach is not required.`}

{IF currentEnvironment.hasPhasedApproach}

* Schedule: {currentEnvironment.phasedApproachSchedule}

{END-IF}

{END-IF}

**4.1.3** {needsArchitecturalDesign || dowArchDesignReq ? `Require an architectural design solution to address a known problem/use-case.` : `Reserved`}

{IF needsArchitecturalDesign}

**4.1.3.1**  {archDesignReq.source === `CURRENT\_ENVIRONMENT` ?`Known problem/use-case (current environment)` : `Reserved`}

{archDesignReq.statement}

{EXEC designForApps = archDesignReq.applicationsNeedingDesign}

{designForApps ? `Applicable applications are ` + designForApps : ` There are no specific application requirements.`}

{IF archDesignReq.dataClassificationLevels.length > 0}

Classification level(s):

{EXEC dataLevels = filterDataLevels(archDesignReq.dataClassificationLevels)}

{END-IF}

{IF dataLevels.unclassified.length === 1}

{FOR level IN dataLevels.unclassified}

{IF $level.impactLevel === `IL2`}

Unclassified - IL2 (Accommodates DoD information that has been approved for public release (Low Confidentiality & Moderate Integrity))

{END-IF}

{IF $level.impactLevel === `IL4`}

Unclassified - IL4 (Accommodates DoD Controlled Unclassified Information (CUI))

{END-IF}

{IF $level.impactLevel === `IL5`}

Unclassified - IL5 (Accommodates DoD CUI & National Security Systems)

{END-IF}

{END-FOR level}

{END-IF}

{IF dataLevels.unclassified.length > 1 }

Unclassified

{FOR level IN dataLevels.unclassified}

{IF $level.impactLevel === `IL2`}

* + IL2 (Accommodates DoD information that has been approved for public release (Low Confidentiality & Moderate Integrity))

{END-IF}

{IF $level.impactLevel === `IL4`}

* + IL4 (Accommodates DoD Controlled Unclassified Information (CUI))

{END-IF}

{IF $level.impactLevel === `IL5`}

* + IL5 (Accommodates DoD CUI & National Security Systems)

{END-IF}

{END-FOR level}

{END-IF}

{IF dataLevels.secret}

Secret - IL6 (Accommodates DoD Classified Information up to SECRET)

{END-IF}

{IF dataLevels.ts}

TS

{END-IF}

{archDesignReq.externalFactors ? ` Other factors to consider regarding deployment are ` + archDesignReq.externalFactors : ` There are no external factors to consider regarding deployment.`}

**4.1.3.2**  {dowArchDesignReq ? `Known problem/use-case (other than current environment)` : `Reserved`}

{dowArchDesignReq.statement}

{EXEC designForApps = dowArchDesignReq.applicationsNeedingDesign}

{designForApps ? `Applicable applications are ` + designForApps : ` There are no specific application requirements.`}

{IF dowArchDesignReq.dataClassificationLevels.length > 0}

Classification level(s):

{EXEC dataLevels = filterDataLevels(dowArchDesignReq.dataClassificationLevels)}

{END-IF}

{IF dataLevels.unclassified.length === 1}

{FOR level IN dataLevels.unclassified}

{IF $level.impactLevel === `IL2`}

Unclassified - IL2 (Accommodates DoD information that has been approved for public release (Low Confidentiality & Moderate Integrity))

{END-IF}

{IF $level.impactLevel === `IL4`}

Unclassified - IL4 (Accommodates DoD Controlled Unclassified Information (CUI))

{END-IF}

{IF $level.impactLevel === `IL5`}

Unclassified - IL5 (Accommodates DoD CUI & National Security Systems)

{END-IF}

{END-FOR level}

{END-IF}

{IF dataLevels.unclassified.length > 1 }

Unclassified

{FOR level IN dataLevels.unclassified}

{IF $level.impactLevel === `IL2`}

* + IL2 (Accommodates DoD information that has been approved for public release (Low Confidentiality & Moderate Integrity))

{END-IF}

{IF $level.impactLevel === `IL4`}

* + IL4 (Accommodates DoD Controlled Unclassified Information (CUI))

{END-IF}

{IF $level.impactLevel === `IL5`}

* + IL5 (Accommodates DoD CUI & National Security Systems)

{END-IF}

{END-FOR level}

{END-IF}

{IF dataLevels.secret}

Secret - IL6 (Accommodates DoD Classified Information up to SECRET)

{END-IF}

{IF dataLevels.ts}

TS

{END-IF}

{dowArchDesignReq.externalFactors ? ` Other factors to consider regarding deployment are ` + dowArchDesignReq.externalFactors : ` There are no external factors to consider regarding deployment.`}

{END-IF}

**4.2** Task 2 - XaaS (Anything as a Service).

{EXEC egressPerMonth = calcAvgDataEgress(selectedClassificationLevels)}

{EXEC il2Level = sortedSelectedClassificationLevels.il2}

{EXEC il4Level = sortedSelectedClassificationLevels.il4}

{EXEC il5Level = sortedSelectedClassificationLevels.il5}

{EXEC il6Level = sortedSelectedClassificationLevels.il6}

{EXEC tsLevel = sortedSelectedClassificationLevels.ts}

{EXEC compute = xaasServices.computeInstances}

{EXEC database = xaasServices.databaseInstances}

{EXEC storage = xaasServices.storageInstances}

{EXEC general = xaasServices.generalInstances}

{EXEC selectedInstances = xaasServices.selectedInstances}

{EXEC selectedServices = Object.keys(selectedInstances)}

{EXEC il2EdgeCount = 0}

{EXEC il4EdgeCount = 0}

{EXEC il5EdgeCount = 0}

{EXEC il6EdgeCount = 0}

{EXEC tsEdgeCount = 0}

Approximate data/internet egress per month ({egressPerMonth.dataEgressMonthlyUnit ? egressPerMonth.dataEgressMonthlyUnit : `N/A`}) (across entire duration of this TO): {egressPerMonth ? egressPerMonth.dataEgressAverage + ` ` + egressPerMonth.dataEgressMonthlyUnit : `N/A`}

**4.2.1** {il2Level ? `Unclassified - IL2` : `Reserved`}

{IF il2Level}

Location(s) and approximate number of current users:

{IF il2Level.usersPerRegion.length > 0}

{FOR regionUsers IN formatRegionUsers(il2Level.usersPerRegion)}

* {$regionUsers} users

{END-FOR regionUsers}

{END-IF}

{IF il2Level.usersIncrease || il2Level.dataIncrease}

Anticipated future needs:

{IF !il2Level.usersIncrease}

* Users: Static

{END-IF}

{IF il2Level.usersIncrease}

{IF il2Level.userGrowthEstimateType === `SINGLE`}

* Users: Estimated lifecycle growth: {il2Level.userGrowthEstimatePercentage[0]}%

{END-IF}

{IF il2Level.userGrowthEstimateType === `MULTIPLE`}

* Users: Estimated growth per period:

{FOR estimatedGrowth IN formatGrowthEstimates(il2Level.userGrowthEstimatePercentage) }

* + {$estimatedGrowth}

{END-FOR estimatedGrowth}

{END-IF}

{END-IF}

{IF !il2Level.dataIncrease}

* Data: Static

{END-IF}

{IF il2Level.dataIncrease}

{IF il2Level.dataGrowthEstimateType === `SINGLE`}

* Data: Estimated lifecycle growth: {il2Level.dataGrowthEstimatePercentage[0]}%

{END-IF}

{IF il2Level.dataGrowthEstimateType === `MULTIPLE`}

* Data: Estimated growth per period:

{FOR estimatedGrowth IN formatGrowthEstimates(il2Level.dataGrowthEstimatePercentage) }

* + {$estimatedGrowth}

{END-FOR estimatedGrowth}

{END-IF}

{END-IF}

{END-IF}

**4.2.1.1** {compute.il2.length > 0 ? `Subtask 1 Compute (Unclassified - IL2)` : `Reserved`}

{IF compute.il2.length > 0}

{EXEC instancesCount = getInstancesCount(compute.il2)}

Number of instances: {instancesCount.totalInstances}

* Dev/Test: {instancesCount.dev}
* Pre-Production: {instancesCount.preProd}
* Production/Staging: {instancesCount.prod}
* COOP/Disaster Recovery: {instancesCount.coop}

{EXEC il2ComputeCount = 1}

{FOR compute IN compute.il2}

**4.2.1.1.{=il2ComputeCount}** Instance {=il2ComputeCount}

* Number of instances with similar configurations: {$compute.numberOfInstances ? $compute.numberOfInstances : `N/A`}
* Operating environment: {$compute.operatingEnvironment ? formatEnum($compute.operatingEnvironment) : `N/A`}
* OS and licensing: {$compute.operatingSystemLicensing ? formatEnum($compute.operatingSystemLicensing) : `N/A`}
* Approximate number of vCPUs/size of compute: {$compute.numberOfVcpus ? $compute.numberOfVcpus : `N/A`}
  + Processor speed: {$compute.processorSpeed ? $compute.processorSpeed : `N/A`}
  + Operating system: {$compute.operatingSystem ? $compute.operatingSystem : `N/A`}
* Memory ({$compute.memoryType ? $compute.memoryType : `GB`}): { $compute.memoryAmount ? $compute.memoryAmount + ` ` + $compute.memoryUnit : `N/A`}
* Performance tier: {IF !$compute.performanceTier }: N/A{END-IF}

{IF $compute.performanceTier === `GENERAL`}

* + General Purpose (Provides a balance of compute, memory & network)

{END-IF}

{IF $compute.performanceTier === `COMPUTE`}

* + Compute Optimized (Supports compute-bound applications that benefit from high performance processors)

{END-IF}

{IF $compute.performanceTier === `MEMORY`}

* + Memory Optimized (Designed to deliver fast performance for workloads that process large data sets in memory)

{END-IF}

{IF $compute.performanceTier === `STORAGE`}

* + Storage Optimized (Designed for high, sequential read and write workloads to very large data sets on local storage)

{END-IF}

* Storage type and size ({$compute.storageUnit ? $compute.storageUnit : `GB`}): {IF !$compute.storageType} N/A {END-IF}

{IF $compute.storageType === `BLOCK`}

* + Block Storage (Fixed-sized raw storage capacity): {$compute.storageAmount} {$compute.storageUnit}

{END-IF}

{IF $compute.storageType === `OBJECT`}

* + Object Storage (Store and serve unstructured user-generated content): {$compute.storageAmount} {$compute.storageUnit}

{END-IF}

{IF $compute.storageType === `FILE`}

* + File Storage (Store and serve shared file systems): {$compute.storageAmount} {$compute.storageUnit}

{END-IF}

{IF $compute.storageType === `ARCHIVE`}

* + Archive Storage (Store and serve for long-term data retention): {$compute.storageAmount} {$compute.storageUnit}

{END-IF}

* Statement of objectives for anticipated need/usage:

{$compute.anticipatedNeedOrUsage}

{EXEC il2ComputeCount++}

{END-FOR compute}

{END-IF}

{IF compute.il2.length < 1}

{END-IF}

{EXEC il2DevTools = selectedServiceExists(selectedServices, selectedInstances, `DEVELOPER\_TOOLS`, `il2`) && il2Level}

**4.2.1.2** {il2DevTools ? `Subtask 2 Developer Tools and Services (Unclassified - IL2)` : `Reserved`}

{IF il2DevTools}

{EXEC il2DevToolsCount = 1}

{FOR il2DevTool IN selectedInstances[`DEVELOPER\_TOOLS`].il2}

**4.2.1.2.{=il2DevToolsCount}** {$il2DevTool.name}

{$il2DevTool.usageDescription}

{EXEC il2DevToolsCount++}

{END-FOR il2DevTool}

{END-IF}

{EXEC il2AppTools = selectedServiceExists(selectedServices, selectedInstances, `APPLICATIONS`, `il2`) && il2Level }

**4.2.1.3** {il2AppTools ? `Subtask 3 Applications (Unclassified - IL2)` : `Reserved`}

{IF il2AppTools}

{EXEC il2AppToolsCount = 1}

{FOR il2AppTool IN selectedInstances[`APPLICATIONS`].il2}

**4.2.1.3.{=il2AppToolsCount}** {$il2AppTool.name}

{$il2AppTool.usageDescription}

{EXEC il2AppToolsCount++}

{END-FOR il2AppTool}

{END-IF}

{EXEC il2MlTools = selectedServiceExists(selectedServices, selectedInstances, `MACHINE\_LEARNING`, `il2`) && il2Level}

**4.2.1.4** {il2MlTools ? `Subtask 4 Advanced Technology and Algorithmic techniques (Machine Learning) and Data Analytics (Unclassified - IL2)` : `Reserved`}

{IF il2MlTools}

{EXEC il2MlToolsCount = 1}

{FOR il2MlTool IN selectedInstances[`MACHINE\_LEARNING`].il2}

**4.2.1.4.{=il2MlToolsCount}** {$il2MlTool.name}

{$il2MlTool.usageDescription}

{EXEC il2MlToolsCount++}

{END-FOR il2MlTool}

{END-IF}

{EXEC il2NetTools = selectedServiceExists(selectedServices, selectedInstances, `NETWORK`, `il2`) && il2Level }

**4.2.1.5** {il2NetTools ? `Subtask 5 Networking (Unclassified - IL2)` : `Reserved`}

{IF il2NetTools}

{EXEC il2NetToolsCount = 1}

{FOR il2NetTool IN selectedInstances[`NETWORK`].il2}

**4.2.1.5.{=il2NetToolsCount}** {$il2NetTool.name}

{$il2NetTool.usageDescription}

{EXEC il2NetToolsCount++}

{END-FOR il2NetTool}

{END-IF}

{EXEC il2SecTools = selectedServiceExists(selectedServices, selectedInstances, `SECURITY`, `il2`) && il2Level }

**4.2.1.6** {il2SecTools ? `Subtask 6 Security (Unclassified - IL2)` : `Reserved`}

{IF il2SecTools}

{EXEC il2SecToolsCount = 1}

{FOR il2SecTool IN selectedInstances[`SECURITY`].il2}

**4.2.1.6.{=il2SecToolsCount}** {$il2SecTool.name}

{$il2SecTool.usageDescription}

{EXEC il2SecToolsCount++}

{END-FOR il2SecTool}

{END-IF}

**4.2.1.7** {database.il2.length > 0 ? `Subtask 7 Database (Unclassified - IL2)` : `Reserved`}

{IF database.il2.length > 0}

{EXEC instancesCount = getInstancesCount(database.il2)}

Number of instances: {instancesCount.totalInstances}

{EXEC il2DatabaseCount = 1}

{FOR database IN database.il2}

**4.2.1.7.{=il2DatabaseCount}** Instance {=il2DatabaseCount}

* Number of instances with similar configurations: {$database.numberOfInstances ? $database.numberOfInstances : `N/A`}
* Approximate number of vCPUs/size of compute: {$database.numberOfVcpus ? $database.numberOfVcpus : `N/A`}
  + Processor speed: {$database.processorSpeed ? $database.processorSpeed : `N/A`}
  + Operating system: {$database.operatingSystem ? $database.operatingSystem : `N/A`}
* OS and licensing: {$database.operatingSystemLicensing ? formatEnum($database.operatingSystemLicensing) : `N/A`}
* DB and licensing: {$database.databaseLicensing ? formatEnum($database.databaseLicensing) : `N/A`}
* Memory ({$database.memoryType ? $database.memoryType : `GB`}): { $database.memoryAmount ? $database.memoryAmount + ` ` + $database.memoryUnit : `N/A`}
* Network performance: {$database.networkPerformance ? formatEnum($database.networkPerformance) : `N/A`}
* Storage type and size ({$database.storageUnit ? $database.storageUnit : `GB`}): {IF !$database.storageType} N/A {END-IF}

{IF $database.storageType === `BLOCK`}

* + Block Storage (Fixed-sized raw storage capacity): {$database.storageAmount} {$database.storageUnit}

{END-IF}

{IF $database.storageType === `OBJECT`}

* + Object Storage (Store and serve unstructured user-generated content): {$database.storageAmount} {$database.storageUnit}

{END-IF}

{IF $database.storageType === `FILE`}

* + File Storage (Store and serve shared file systems): {$database.storageAmount} {$database.storageUnit}

{END-IF}

{IF $database.storageType === `ARCHIVE`}

* + Archive Storage (Store and serve for long-term data retention): {$database.storageAmount} {$database.storageUnit}

{END-IF}

* Database Type: {$database.databaseType ? formatEnum($database.databaseType) : `N/A`}
* Statement of objectives for anticipated need/usage, to include the purpose and usage of the expected tools/capabilities:

{$database.anticipatedNeedOrUsage}

{EXEC il2DatabaseCount++}

{END-FOR database}

{END-IF}

{IF database.il2.length < 1}

{END-IF}

**4.2.1.8** {storage.il2.length > 0 ? `Subtask 8 Storage (requirements separate from Compute/Database needs) (Unclassified - IL2)` : `Reserved`}

{IF storage.il2.length > 0}

{EXEC il2StorageCount = 1}

{FOR storage IN storage.il2}

**4.2.1.8.{=il2StorageCount}** Instance {=il2StorageCount}

* Number of instances with similar configurations: {$storage.numberOfInstances ? $storage.numberOfInstances : `N/A`}
* Storage type and size ({$storage.storageUnit ? $storage.storageUnit : `GB`}): {IF !$storage.storageType} N/A {END-IF}

{IF $storage.storageType === `BLOCK`}

* + Block Storage (Fixed-sized raw storage capacity): {$storage.storageAmount} {$storage.storageUnit}

{END-IF}

{IF $storage.storageType === `OBJECT`}

* + Object Storage (Store and serve unstructured user-generated content): {$storage.storageAmount} {$storage.storageUnit}

{END-IF}

{IF $storage.storageType === `FILE`}

* + File Storage (Store and serve shared file systems): {$storage.storageAmount} {$storage.storageUnit}

{END-IF}

{IF $storage.storageType === `ARCHIVE`}

* + Archive Storage (Store and serve for long-term data retention): {$storage.storageAmount} {$storage.storageUnit}

{END-IF}

* Statement of objectives for anticipated need/usage, to include the purpose and usage of the expected tools/capabilities:

{$storage.anticipatedNeedOrUsage}

{EXEC il2StorageCount++}

{END-FOR storage}

{END-IF}

{EXEC il2TeTools = selectedServiceExists(selectedServices, selectedInstances, `EDGE\_COMPUTING`, `il2`) && il2Level }

**4.2.1.9** {il2TeTools ? `Subtask 9 Edge Computing and Tactical Edge (TE) (Unclassified - IL2)` : `Reserved`}

{IF il2TeTools}

{EXEC il2TeToolsCount = 1}

{FOR il2TeTool IN selectedInstances[`EDGE\_COMPUTING`].il2}

**4.2.1.9.{=il2TeToolsCount}** {$il2TeTool.name} {IF $il2TeTool.name === `Mobility`}({$il2TeTool.typeOfMobility ? formatEnum($il2TeTool.typeOfMobility) : `N/A`}) {END-IF}

{IF $il2TeTool.name === `Delivery`}

* {$il2TeTool.typeOfDelivery ? formatEnum($il2TeTool.typeOfDelivery) : `N/A`}

{END-IF}

{IF $il2TeTool.otherServiceOffering}

Other

* {$il2TeTool.otherServiceOffering ? $il2TeTool.otherServiceOffering: `N/A`}

{END-IF}

{$il2TeTool.usageDescription}

{EXEC il2TeToolsCount++}

{END-FOR il2TeTool}

{END-IF}

{EXEC il2IotTools = selectedServiceExists(selectedServices, selectedInstances, `IOT`, `il2`) && il2Level}

**4.2.1.10** {il2IotTools ? `Subtask 10 Internet of Things (IoT) (Unclassified - IL2)` : `Reserved`}

{IF il2IotTools}

{EXEC il2IotToolsCount = 1}

{FOR il2IotTool IN selectedInstances[`IOT`].il2}

**4.2.1.10.{=il2IotToolsCount}** {$il2IotTool.name}

{$il2IotTool.usageDescription}

{EXEC il2IotToolsCount++}

{END-FOR il2IotTool}

{END-IF}

**4.2.1.11** {general.il2.length > 0 ? `General Infrastructure as a Service (IaaS), PaaS, and SaaS to include third party marketplace (not covered in Subtasks 4.2.1.1-4.2.1.10) (Unclassified - IL2)` : `Reserved`}

{IF general.il2.length > 0}

{EXEC il2GeneralCount = 1}

{FOR general IN general.il2}

**4.2.1.11.{=il2GeneralCount}** Instance {=il2GeneralCount}

* Statement of objectives for anticipated need/usage, to include the purpose and usage of the expected tools/capabilities:

{$general.anticipatedNeedOrUsage}

{EXEC il2GeneralCount++}

{END-FOR general}

{END-IF}

{END-IF}

**4.2.2** {il4Level ? `Unclassified - IL4` : `Reserved`}

{IF il4Level}

Location(s) and approximate number of current users:

{IF il4Level.usersPerRegion.length > 0}

{FOR regionUsers IN formatRegionUsers(il4Level.usersPerRegion)}

* {$regionUsers} users

{END-FOR regionUsers}

{END-IF}

{IF il4Level.usersIncrease || il4Level.dataIncrease}

Anticipated future needs:

{IF !il4Level.usersIncrease}

* Users: Static

{END-IF}

{IF il4Level.usersIncrease}

{IF il4Level.userGrowthEstimateType === `SINGLE`}

* Users: Estimated lifecycle growth: {il4Level.userGrowthEstimatePercentage[0]}%

{END-IF}

{IF il4Level.userGrowthEstimateType === `MULTIPLE`}

* Users: Estimated growth per period:

{FOR estimatedGrowth IN formatGrowthEstimates(il4Level.userGrowthEstimatePercentage) }

* + {$estimatedGrowth}

{END-FOR estimatedGrowth}

{END-IF}

{END-IF}

{IF !il4Level.dataIncrease}

* Data: Static

{END-IF}

{IF il4Level.dataIncrease}

{IF il4Level.dataGrowthEstimateType === `SINGLE`}

* Data: Estimated lifecycle growth: {il4Level.dataGrowthEstimatePercentage[0]}%

{END-IF}

{IF il4Level.dataGrowthEstimateType === `MULTIPLE`}

* Data: Estimated growth per period:

{FOR estimatedGrowth IN formatGrowthEstimates(il4Level.dataGrowthEstimatePercentage) }

* + {$estimatedGrowth}

{END-FOR estimatedGrowth}

{END-IF}

{END-IF}

{END-IF}

**4.2.2.1** {compute.il4.length > 0 ? `Subtask 1 Compute (Unclassified - IL4)` : `Reserved`}

{IF compute.il4.length > 0}

{EXEC instancesCount = getInstancesCount(compute.il4)}

Number of instances: {instancesCount.totalInstances}

* Dev/Test: {instancesCount.dev}
* Pre-Production: {instancesCount.preProd}
* Production/Staging: {instancesCount.prod}
* COOP/Disaster Recovery: {instancesCount.coop}

{EXEC il4ComputeCount = 1}

{FOR compute IN compute.il4}

**4.2.2.1.{=il4ComputeCount}** Instance {=il4ComputeCount}

* Number of instances with similar configurations: {$compute.numberOfInstances ? $compute.numberOfInstances : `N/A`}
* Operating environment: {$compute.operatingEnvironment ? formatEnum($compute.operatingEnvironment) : `N/A`}
* OS and licensing: {$compute.operatingSystemLicensing ? formatEnum($compute.operatingSystemLicensing) : `N/A`}
* Approximate number of vCPUs/size of compute: {$compute.numberOfVcpus ? $compute.numberOfVcpus : `N/A`}
  + Processor speed: {$compute.processorSpeed ? $compute.processorSpeed : `N/A`}
  + Operating system: {$compute.operatingSystem ? $compute.operatingSystem : `N/A`}
* Memory ({$compute.memoryType ? $compute.memoryType : `GB`}): { $compute.memoryAmount ? $compute.memoryAmount + ` ` + $compute.memoryUnit : `N/A`}
* Performance tier: {IF !$compute.performanceTier }: N/A{END-IF}

{IF $compute.performanceTier === `GENERAL`}

* + General Purpose (Provides a balance of compute, memory & network)

{END-IF}

{IF $compute.performanceTier === `COMPUTE`}

* + Compute Optimized (Supports compute-bound applications that benefit from high performance processors)

{END-IF}

{IF $compute.performanceTier === `MEMORY`}

* + Memory Optimized (Designed to deliver fast performance for workloads that process large data sets in memory)

{END-IF}

{IF $compute.performanceTier === `STORAGE`}

* + Storage Optimized (Designed for high, sequential read and write workloads to very large data sets on local storage)

{END-IF}

* Storage type and size ({$compute.storageUnit ? $compute.storageUnit : `GB`}): {IF !$compute.storageType} N/A {END-IF}

{IF $compute.storageType === `BLOCK`}

* + Block Storage (Fixed-sized raw storage capacity): {$compute.storageAmount} {$compute.storageUnit}

{END-IF}

{IF $compute.storageType === `OBJECT`}

* + Object Storage (Store and serve unstructured user-generated content): {$compute.storageAmount} {$compute.storageUnit}

{END-IF}

{IF $compute.storageType === `FILE`}

* + File Storage (Store and serve shared file systems): {$compute.storageAmount} {$compute.storageUnit}

{END-IF}

{IF $compute.storageType === `ARCHIVE`}

* + Archive Storage (Store and serve for long-term data retention): {$compute.storageAmount} {$compute.storageUnit}

{END-IF}

* Statement of objectives for anticipated need/usage:

{$compute.anticipatedNeedOrUsage}

{EXEC il4ComputeCount++}

{END-FOR compute}

{END-IF}

{EXEC il4DevTools = selectedServiceExists(selectedServices, selectedInstances, `DEVELOPER\_TOOLS`, `il4`) && il4Level}

**4.2.2.2** {il4Level && il4DevTools ? `Subtask 2 Developer Tools and Services (Unclassified - IL4)` : `Reserved`}

{IF il4DevTools}

{EXEC il4DevToolsCount = 1}

{FOR il4DevTool IN selectedInstances[`DEVELOPER\_TOOLS`].il4}

**4.2.2.2.{=il4DevToolsCount}** {$il4DevTool.name}

{$il4DevTool.usageDescription}

{EXEC il4DevToolsCount++}

{END-FOR il4DevTool}

{END-IF}

{EXEC il4AppTools = selectedServiceExists(selectedServices, selectedInstances, `APPLICATIONS`, `il4`) && il4Level }

**4.2.2.3** {il4AppTools ? `Subtask 3 Applications (Unclassified - IL4)` : `Reserved`}

{IF il4AppTools}

{EXEC il4AppToolsCount = 1}

{FOR il4AppTool IN selectedInstances[`APPLICATIONS`].il4}

**4.2.2.3.{=il4AppToolsCount}** {$il4AppTool.name}

{$il4AppTool.usageDescription}

{EXEC il4AppToolsCount++}

{END-FOR il4AppTool}

{END-IF}

{EXEC il4MlTools = selectedServiceExists(selectedServices, selectedInstances, `MACHINE\_LEARNING`, `il4`) && il4Level}

**4.2.2.4** {il4MlTools ? `Subtask 4 Advanced Technology and Algorithmic techniques (Machine Learning) and Data Analytics (Unclassified - IL4)` : `Reserved`}

{IF il4MlTools}

{EXEC il4MlToolsCount = 1}

{FOR il4MlTool IN selectedInstances[`MACHINE\_LEARNING`].il4}

**4.2.2.4.{=il4MlToolsCount}** {$il4MlTool.name}

{$il4MlTool.usageDescription}

{EXEC il4MlToolsCount++}

{END-FOR il4MlTool}

{END-IF}

{EXEC il4NetTools = selectedServiceExists(selectedServices, selectedInstances, `NETWORK`, `il4`) && il4Level}

**4.2.2.5** {il4NetTools ? `Subtask 5 Networking (Unclassified - IL4)` : `Reserved`}

{IF il4NetTools}

{EXEC il4NetToolsCount = 1}

{FOR il4NetTool IN selectedInstances[`NETWORK`].il4}

**4.2.2.5.{=il4NetToolsCount}** {$il4NetTool.name}

{$il4NetTool.usageDescription}

{EXEC il4NetToolsCount++}

{END-FOR il4NetTool}

{END-IF}

{EXEC il4SecTools = selectedServiceExists(selectedServices, selectedInstances, `SECURITY`, `il4`) && il4Level }

**4.2.2.6** {il4SecTools ? `Subtask 6 Security (Unclassified - IL4)` : `Reserved`}

{IF il4SecTools}

{EXEC il4SecToolsCount = 1}

{FOR il4SecTool IN selectedInstances[`SECURITY`].il4}

**4.2.2.6.{=il4SecToolsCount}** {$il4SecTool.name}

{$il4SecTool.usageDescription}

{EXEC il4SecToolsCount++}

{END-FOR il4SecTool}

{END-IF}

**4.2.2.7** {database.il4.length > 0 ? `Subtask 7 Database (Unclassified - IL4)` : `Reserved`}

{IF database.il4.length > 0}

{EXEC instancesCount = getInstancesCount(database.il4)}

Number of instances: {instancesCount.totalInstances}

{EXEC il4DatabaseCount = 1}

{FOR database IN database.il4}

**4.2.2.7.{=il4DatabaseCount}** Instance {=il4DatabaseCount}

* Number of instances with similar configurations: {$database.numberOfInstances ? $database.numberOfInstances : `N/A`}
* Approximate number of vCPUs/size of compute: {$database.numberOfVcpus ? $database.numberOfVcpus : `N/A`}
  + Processor speed: {$database.processorSpeed ? $database.processorSpeed : `N/A`}
  + Operating system: {$database.operatingSystem ? $database.operatingSystem : `N/A`}
* OS and licensing: {$database.operatingSystemLicensing ? formatEnum($database.operatingSystemLicensing) : `N/A`}
* DB and licensing: {$database.databaseLicensing ? formatEnum($database.databaseLicensing) : `N/A`}
* Memory ({$database.memoryType ? $database.memoryType : `GB`}): { $database.memoryAmount ? $database.memoryAmount + ` ` + $database.memoryUnit : `N/A`}
* Network performance: {$database.networkPerformance ? formatEnum($database.networkPerformance) : `N/A`}
* Storage type and size ({$database.storageUnit ? $database.storageUnit : `GB`}): {IF !$database.storageType} N/A {END-IF}

{IF $database.storageType === `BLOCK`}

* + Block Storage (Fixed-sized raw storage capacity): {$database.storageAmount} {$database.storageUnit}

{END-IF}

{IF $database.storageType === `OBJECT`}

* + Object Storage (Store and serve unstructured user-generated content): {$database.storageAmount} {$database.storageUnit}

{END-IF}

{IF $database.storageType === `FILE`}

* + File Storage (Store and serve shared file systems): {$database.storageAmount} {$database.storageUnit}

{END-IF}

{IF $database.storageType === `ARCHIVE`}

* + Archive Storage (Store and serve for long-term data retention): {$database.storageAmount} {$database.storageUnit}

{END-IF}

* Database Type: {$database.databaseType ? formatEnum($database.databaseType) : `N/A`}
* Statement of objectives for anticipated need/usage, to include the purpose and usage of the expected tools/capabilities:

{$database.anticipatedNeedOrUsage}

{EXEC il4DatabaseCount++}

{END-FOR database}

{IF database.il4.length < 1}

{END-IF}

{END-IF}

**4.2.2.8** {storage.il4.length > 0 ? `Subtask 8 Storage (requirements separate from Compute/Database needs) (Unclassified - IL4)` : `Reserved`}

{IF storage.il4.length > 0}

{EXEC il4StorageCount = 1}

{FOR storage IN storage.il4}

**4.2.2.8.{=il4StorageCount}** Instance {=il4StorageCount}

* Number of instances with similar configurations: {$storage.numberOfInstances ? $storage.numberOfInstances : `N/A`}
* Storage type and size ({$storage.storageUnit ? $storage.storageUnit : `GB`}): {IF !$storage.storageType} N/A {END-IF}

{IF $storage.storageType === `BLOCK`}

* + Block Storage (Fixed-sized raw storage capacity): {$storage.storageAmount} {$storage.storageUnit}

{END-IF}

{IF $storage.storageType === `OBJECT`}

* + Object Storage (Store and serve unstructured user-generated content): {$storage.storageAmount} {$storage.storageUnit}

{END-IF}

{IF $storage.storageType === `FILE`}

* + File Storage (Store and serve shared file systems): {$storage.storageAmount} {$storage.storageUnit}

{END-IF}

{IF $storage.storageType === `ARCHIVE`}

* + Archive Storage (Store and serve for long-term data retention): {$storage.storageAmount} {$storage.storageUnit}

{END-IF}

* Statement of objectives for anticipated need/usage, to include the purpose and usage of the expected tools/capabilities:

{$storage.anticipatedNeedOrUsage}

{EXEC il4StorageCount++}

{END-FOR storage}

{END-IF}

{EXEC il4TeTools = selectedServiceExists(selectedServices, selectedInstances, `EDGE\_COMPUTING`, `il4`) && il4Level}

**4.2.2.9** {il4TeTools ? `Subtask 9 Edge Computing and Tactical Edge (TE) (Unclassified - IL4)` : `Reserved`}

{IF il4TeTools}

{EXEC il4TeToolsCount = 1}

{FOR il4TeTool IN selectedInstances[`EDGE\_COMPUTING`].il4}

**4.2.2.9.{=il4TeToolsCount}** {$il4TeTool.name} {IF $il4TeTool.name === `Mobility`}({$il4TeTool.typeOfMobility ? formatEnum($il4TeTool.typeOfMobility) : `N/A`}) {END-IF}

{IF $il4TeTool.name === `Delivery`}

* {$il4TeTool.typeOfDelivery ? formatEnum($il4TeTool.typeOfDelivery) : `N/A`}

{END-IF}

{IF $il4TeTool.otherServiceOffering}

Other

* {$il4TeTool.otherServiceOffering ? $il4TeTool.otherServiceOffering: `N/A`}

{END-IF}

{$il4TeTool.usageDescription}

{EXEC il4TeToolsCount++}

{END-FOR il4TeTool}

{END-IF}

{EXEC il4IotTools = selectedServiceExists(selectedServices, selectedInstances, `IOT`, `il4`) && il4Level}

**4.2.2.10** {il4IotTools ? `Subtask 10 Internet of Things (IoT) (Unclassified - IL4)` : `Reserved`}

{IF il4IotTools}

{EXEC il4IotToolsCount = 1}

{FOR il4IotTool IN selectedInstances[`IOT`].il4}

**4.2.2.10.{=il4IotToolsCount}** {$il4IotTool.name}

{$il4IotTool.usageDescription}

{EXEC il4IotToolsCount++}

{END-FOR il4IotTool}

{END-IF}

**4.2.2.11** {general.il4.length > 0 ? `General Infrastructure as a Service (IaaS), PaaS, and SaaS to include third party marketplace (not covered in Subtasks 4.2.2.1-4.2.2.10) (Unclassified - IL4)` : `Reserved`}

{IF general.il4.length > 0}

{EXEC il4GeneralCount = 1}

{FOR general IN general.il4}

**4.2.2.11.{=il4GeneralCount}** Instance {=il4GeneralCount}

* Statement of objectives for anticipated need/usage, to include the purpose and usage of the expected tools/capabilities:

{$general.anticipatedNeedOrUsage}

{EXEC il4GeneralCount++}

{END-FOR general}

{END-IF}

{END-IF}

{IF general.il4.length < 1}

{END-IF}

**4.2.3** {il5Level ? `Unclassified - IL5` : `Reserved`}

{IF il5Level}

Location(s) and approximate number of current users:

{IF il5Level.usersPerRegion.length > 0}

{FOR regionUsers IN formatRegionUsers(il5Level.usersPerRegion)}

* {$regionUsers} users

{END-FOR regionUsers}

{END-IF}

{IF il5Level.usersIncrease || il5Level.dataIncrease}

Anticipated future needs:

{IF !il5Level.usersIncrease}

* Users: Static

{END-IF}

{IF il5Level.usersIncrease}

{IF il5Level.userGrowthEstimateType === `SINGLE`}

* Users: Estimated lifecycle growth: {il5Level.userGrowthEstimatePercentage[0]}%

{END-IF}

{IF il5Level.userGrowthEstimateType === `MULTIPLE`}

* Users: Estimated growth per period:

{FOR estimatedGrowth IN formatGrowthEstimates(il5Level.userGrowthEstimatePercentage) }

* + {$estimatedGrowth}

{END-FOR estimatedGrowth}

{END-IF}

{END-IF}

{IF !il5Level.dataIncrease}

* Data: Static

{END-IF}

{IF il5Level.dataIncrease}

{IF il5Level.dataGrowthEstimateType === `SINGLE`}

* Data: Estimated lifecycle growth: {il5Level.dataGrowthEstimatePercentage[0]}%

{END-IF}

{IF il5Level.dataGrowthEstimateType === `MULTIPLE`}

* Data: Estimated growth per period:

{FOR estimatedGrowth IN formatGrowthEstimates(il5Level.dataGrowthEstimatePercentage) }

* + {$estimatedGrowth}

{END-FOR estimatedGrowth}

{END-IF}

{END-IF}

{END-IF}

**4.2.3.1** {compute.il5.length > 0 ? `Subtask 1 Compute (Unclassified - IL5)` : `Reserved`}

{IF compute.il5.length > 0}

{EXEC instancesCount = getInstancesCount(compute.il5)}

Number of instances: {instancesCount.totalInstances}

* Dev/Test: {instancesCount.dev}
* Pre-Production: {instancesCount.preProd}
* Production/Staging: {instancesCount.prod}
* COOP/Disaster Recovery: {instancesCount.coop}

{EXEC il5ComputeCount = 1}

{FOR compute IN compute.il5}

**4.2.3.1.{=il5ComputeCount}** Instance {=il5ComputeCount}

* Number of instances with similar configurations: {$compute.numberOfInstances ? $compute.numberOfInstances : `N/A`}
* Operating environment: {$compute.operatingEnvironment ? formatEnum($compute.operatingEnvironment) : `N/A`}
* OS and licensing: {$compute.operatingSystemLicensing ? formatEnum($compute.operatingSystemLicensing) : `N/A`}
* Approximate number of vCPUs/size of compute: {$compute.numberOfVcpus ? $compute.numberOfVcpus : `N/A`}
  + Processor speed: {$compute.processorSpeed ? $compute.processorSpeed : `N/A`}
  + Operating system: {$compute.operatingSystem ? $compute.operatingSystem : `N/A`}
* Memory ({$compute.memoryType ? $compute.memoryType : `GB`}): { $compute.memoryAmount ? $compute.memoryAmount + ` ` + $compute.memoryUnit : `N/A`}
* Performance tier: {IF !$compute.performanceTier }: N/A{END-IF}

{IF $compute.performanceTier === `GENERAL`}

* + General Purpose (Provides a balance of compute, memory & network)

{END-IF}

{IF $compute.performanceTier === `COMPUTE`}

* + Compute Optimized (Supports compute-bound applications that benefit from high performance processors)

{END-IF}

{IF $compute.performanceTier === `MEMORY`}

* + Memory Optimized (Designed to deliver fast performance for workloads that process large data sets in memory)

{END-IF}

{IF $compute.performanceTier === `STORAGE`}

* + Storage Optimized (Designed for high, sequential read and write workloads to very large data sets on local storage)

{END-IF}

* Storage type and size ({$compute.storageUnit ? $compute.storageUnit : `GB`}): {IF !$compute.storageType} N/A {END-IF}

{IF $compute.storageType === `BLOCK`}

* + Block Storage (Fixed-sized raw storage capacity): {$compute.storageAmount} {$compute.storageUnit}

{END-IF}

{IF $compute.storageType === `OBJECT`}

* + Object Storage (Store and serve unstructured user-generated content): {$compute.storageAmount} {$compute.storageUnit}

{END-IF}

{IF $compute.storageType === `FILE`}

* + File Storage (Store and serve shared file systems): {$compute.storageAmount} {$compute.storageUnit}

{END-IF}

{IF $compute.storageType === `ARCHIVE`}

* + Archive Storage (Store and serve for long-term data retention): {$compute.storageAmount} {$compute.storageUnit}

{END-IF}

* Statement of objectives for anticipated need/usage:

{$compute.anticipatedNeedOrUsage}

{EXEC il5ComputeCount++}

{END-FOR compute}

{END-IF}

{EXEC il5DevTools = selectedServiceExists(selectedServices, selectedInstances, `DEVELOPER\_TOOLS`, `il5`) && il5Level}

**4.2.3.2** {il5Level && il5DevTools ? `Subtask 2 Developer Tools and Services (Unclassified - IL5)` : `Reserved`}

{IF il5DevTools}

{EXEC il5DevToolsCount = 1}

{FOR il5DevTool IN selectedInstances[`DEVELOPER\_TOOLS`].il5}

**4.2.3.2.{=il5DevToolsCount}** {$il5DevTool.name}

{$il5DevTool.usageDescription}

{EXEC il5DevToolsCount++}

{END-FOR il5DevTool}

{END-IF}

{EXEC il5AppTools = selectedServiceExists(selectedServices, selectedInstances, `APPLICATIONS`, `il5`) && il5Level }

**4.2.3.3** {il5AppTools ? `Subtask 3 Applications (Unclassified - IL5)` : `Reserved`}

{IF il5AppTools}

{EXEC il5AppToolsCount = 1}

{FOR il5AppTool IN selectedInstances[`APPLICATIONS`].il5}

**4.2.3.3.{=il5AppToolsCount}** {$il5AppTool.name}

{$il5AppTool.usageDescription}

{EXEC il5AppToolsCount++}

{END-FOR il5AppTool}

{END-IF}

{EXEC il5MlTools = selectedServiceExists(selectedServices, selectedInstances, `MACHINE\_LEARNING`, `il5`) && il5Level}

**4.2.3.4** {il5MlTools ? `Subtask 4 Advanced Technology and Algorithmic techniques (Machine Learning) and Data Analytics (Unclassified - IL5)` : `Reserved`}

{IF il5MlTools}

{EXEC il5MlToolsCount = 1}

{FOR il5MlTool IN selectedInstances[`MACHINE\_LEARNING`].il5}

**4.2.3.4.{=il5MlToolsCount}** {$il5MlTool.name}

{$il5MlTool.usageDescription}

{EXEC il5MlToolsCount++}

{END-FOR il5MlTool}

{END-IF}

{EXEC il5NetTools = selectedServiceExists(selectedServices, selectedInstances, `NETWORK`, `il5`) && il5Level}

**4.2.3.5** {il5NetTools ? `Subtask 5 Networking (Unclassified - IL5)` : `Reserved`}

{IF il5NetTools}

{EXEC il5NetToolsCount = 1}

{FOR il5NetTool IN selectedInstances[`NETWORK`].il5}

**4.2.3.5.{=il5NetToolsCount}** {$il5NetTool.name}

{$il5NetTool.usageDescription}

{EXEC il5NetToolsCount++}

{END-FOR il5NetTool}

{END-IF}

{EXEC il5SecTools = selectedServiceExists(selectedServices, selectedInstances, `SECURITY`, `il5`) && il5Level }

**4.2.3.6** {il5SecTools ? `Subtask 6 Security (Unclassified - IL5)` : `Reserved`}

{IF il5SecTools}

{EXEC il5SecToolsCount = 1}

{FOR il5SecTool IN selectedInstances[`SECURITY`].il5}

**4.2.3.6.{=il5SecToolsCount}** {$il5SecTool.name}

{$il5SecTool.usageDescription}

{EXEC il5SecToolsCount++}

{END-FOR il5SecTool}

{END-IF}

**4.2.3.7** {database.il5.length > 0 ? `Subtask 7 Database (Unclassified - IL5)` : `Reserved`}

{IF database.il5.length > 0}

{EXEC instancesCount = getInstancesCount(database.il5)}

Number of instances: {instancesCount.totalInstances}

{EXEC il5DatabaseCount = 1}

{FOR database IN database.il5}

**4.2.3.7.{=il5DatabaseCount}** Instance {=il5DatabaseCount}

* Number of instances with similar configurations: {$database.numberOfInstances ? $database.numberOfInstances : `N/A`}
* Approximate number of vCPUs/size of compute: {$database.numberOfVcpus ? $database.numberOfVcpus : `N/A`}
  + Processor speed: {$database.processorSpeed ? $database.processorSpeed : `N/A`}
  + Operating system: {$database.operatingSystem ? $database.operatingSystem : `N/A`}
* OS and licensing: {$database.operatingSystemLicensing ? formatEnum($database.operatingSystemLicensing) : `N/A`}
* DB and licensing: {$database.databaseLicensing ? formatEnum($database.databaseLicensing) : `N/A`}
* Memory ({$database.memoryType ? $database.memoryType : `GB`}): { $database.memoryAmount ? $database.memoryAmount + ` ` + $database.memoryUnit : `N/A`}
* Network performance: {$database.networkPerformance ? formatEnum($database.networkPerformance) : `N/A`}
* Storage type and size ({$database.storageUnit ? $database.storageUnit : `GB`}): {IF !$database.storageType} N/A {END-IF}

{IF $database.storageType === `BLOCK`}

* + Block Storage (Fixed-sized raw storage capacity): {$database.storageAmount} {$database.storageUnit}

{END-IF}

{IF $database.storageType === `OBJECT`}

* + Object Storage (Store and serve unstructured user-generated content): {$database.storageAmount} {$database.storageUnit}

{END-IF}

{IF $database.storageType === `FILE`}

* + File Storage (Store and serve shared file systems): {$database.storageAmount} {$database.storageUnit}

{END-IF}

{IF $database.storageType === `ARCHIVE`}

* + Archive Storage (Store and serve for long-term data retention): {$database.storageAmount} {$database.storageUnit}

{END-IF}

* Database Type: {$database.databaseType ? formatEnum($database.databaseType) : `N/A`}
* Statement of objectives for anticipated need/usage, to include the purpose and usage of the expected tools/capabilities:

{$database.anticipatedNeedOrUsage}

{EXEC il5DatabaseCount++}

{END-FOR database}

{END-IF}

**4.2.3.8** {storage.il5.length > 0 ? `Subtask 8 Storage (requirements separate from Compute/Database needs) (Unclassified - IL5)` : `Reserved`}

{IF storage.il5.length > 0}

{EXEC il5StorageCount = 1}

{FOR storage IN storage.il5}

**4.2.3.8.{=il5StorageCount}** Instance {=il5StorageCount}

* Number of instances with similar configurations: {$storage.numberOfInstances ? $storage.numberOfInstances : `N/A`}
* Storage type and size ({$storage.storageUnit ? $storage.storageUnit : `GB`}): {IF !$storage.storageType} N/A {END-IF}

{IF $storage.storageType === `BLOCK`}

* + Block Storage (Fixed-sized raw storage capacity): {$storage.storageAmount} {$storage.storageUnit}

{END-IF}

{IF $storage.storageType === `OBJECT`}

* + Object Storage (Store and serve unstructured user-generated content): {$storage.storageAmount} {$storage.storageUnit}

{END-IF}

{IF $storage.storageType === `FILE`}

* + File Storage (Store and serve shared file systems): {$storage.storageAmount} {$storage.storageUnit}

{END-IF}

{IF $storage.storageType === `ARCHIVE`}

* + Archive Storage (Store and serve for long-term data retention): {$storage.storageAmount} {$storage.storageUnit}

{END-IF}

* Statement of objectives for anticipated need/usage, to include the purpose and usage of the expected tools/capabilities:

{$storage.anticipatedNeedOrUsage}

{EXEC il5StorageCount++}

{END-FOR storage}

{END-IF}

{EXEC il5TeTools = selectedServiceExists(selectedServices, selectedInstances, `EDGE\_COMPUTING`, `il5`) && il5Level}

**4.2.3.9** {il5TeTools ? `Subtask 9 Edge Computing and Tactical Edge (TE) (Unclassified - IL5)` : `Reserved`}

{IF il5TeTools}

{EXEC il5TeToolsCount = 1}

{FOR il5TeTool IN selectedInstances[`EDGE\_COMPUTING`].il5}

**4.2.3.9.{=il5TeToolsCount}** {$il5TeTool.name} {IF $il5TeTool.name === `Mobility`}({$il5TeTool.typeOfMobility ? formatEnum($il5TeTool.typeOfMobility) : `N/A`}) {END-IF}

{IF $il5TeTool.name === `Delivery`}

* {$il5TeTool.typeOfDelivery ? formatEnum($il5TeTool.typeOfDelivery) : `N/A`}

{END-IF}

{IF $il5TeTool.otherServiceOffering}

Other

* {$il5TeTool.otherServiceOffering ? $il5TeTool.otherServiceOffering: `N/A`}

{END-IF}

{$il5TeTool.usageDescription}

{EXEC il5TeToolsCount++}

{END-FOR il5TeTool}

{END-IF}

{EXEC il5IotTools = selectedServiceExists(selectedServices, selectedInstances, `IOT`, `il5`) && il5Level}

**4.2.3.10** {il5IotTools ? `Subtask 10 Internet of Things (IoT) (Unclassified - IL5)` : `Reserved`}

{IF il5IotTools}

{EXEC il5IotToolsCount = 1}

{FOR il5IotTool IN selectedInstances[`IOT`].il5}

**4.2.3.10.{=il5IotToolsCount}** {$il5IotTool.name}

{$il5IotTool.usageDescription}

{EXEC il5IotToolsCount++}

{END-FOR il5IotTool}

{END-IF}

**4.2.3.11** {general.il5.length > 0 ? `General Infrastructure as a Service (IaaS), PaaS, and SaaS to include third party marketplace (not covered in Subtasks 4.2.3.1-4.2.3.10) (Unclassified - IL5)` : `Reserved`}

{IF general.il5.length > 0}

{EXEC il5GeneralCount = 1}

{FOR general IN general.il5}

**4.2.3.11.{=il5GeneralCount}** Instance {=il5GeneralCount}

* Statement of objectives for anticipated need/usage, to include the purpose and usage of the expected tools/capabilities:

{$general.anticipatedNeedOrUsage}

{EXEC il5GeneralCount++}

{END-FOR general}

{END-IF}

{END-IF}

**4.2.4** {il6Level ? `Unclassified - IL6` : `Reserved`}

{IF il6Level}

Location(s) and approximate number of current users:

{IF il6Level.usersPerRegion.length > 0}

{FOR regionUsers IN formatRegionUsers(il6Level.usersPerRegion)}

* {$regionUsers} users

{END-FOR regionUsers}

{END-IF}

{IF il6Level.usersIncrease || il6Level.dataIncrease}

Anticipated future needs:

{IF !il6Level.usersIncrease}

* Users: Static

{END-IF}

{IF il6Level.usersIncrease}

{IF il6Level.userGrowthEstimateType === `SINGLE`}

* Users: Estimated lifecycle growth: {il6Level.userGrowthEstimatePercentage[0]}%

{END-IF}

{IF il6Level.userGrowthEstimateType === `MULTIPLE`}

* Users: Estimated growth per period:

{FOR estimatedGrowth IN formatGrowthEstimates(il6Level.userGrowthEstimatePercentage) }

* + {$estimatedGrowth}

{END-FOR estimatedGrowth}

{END-IF}

{END-IF}

{IF !il6Level.dataIncrease}

* Data: Static

{END-IF}

{IF il6Level.dataIncrease}

{IF il6Level.dataGrowthEstimateType === `SINGLE`}

* Data: Estimated lifecycle growth: {il6Level.dataGrowthEstimatePercentage[0]}%

{END-IF}

{IF il6Level.dataGrowthEstimateType === `MULTIPLE`}

* Data: Estimated growth per period:

{FOR estimatedGrowth IN formatGrowthEstimates(il6Level.dataGrowthEstimatePercentage) }

* + {$estimatedGrowth}

{END-FOR estimatedGrowth}

{END-IF}

{END-IF}

{END-IF}

**4.2.4.1** {compute.il6.length > 0 ? `Subtask 1 Compute (Unclassified - IL6)` : `Reserved`}

{IF compute.il6.length > 0}

{EXEC instancesCount = getInstancesCount(compute.il6)}

Number of instances: {instancesCount.totalInstances}

* Dev/Test: {instancesCount.dev}
* Pre-Production: {instancesCount.preProd}
* Production/Staging: {instancesCount.prod}
* COOP/Disaster Recovery: {instancesCount.coop}

{EXEC il6ComputeCount = 1}

{FOR compute IN compute.il6}

**4.2.4.1.{=il6ComputeCount}** Instance {=il6ComputeCount}

* Number of instances with similar configurations: {$compute.numberOfInstances ? $compute.numberOfInstances : `N/A`}
* Operating environment: {$compute.operatingEnvironment ? formatEnum($compute.operatingEnvironment) : `N/A`}
* OS and licensing: {$compute.operatingSystemLicensing ? formatEnum($compute.operatingSystemLicensing) : `N/A`}
* Approximate number of vCPUs/size of compute: {$compute.numberOfVcpus ? $compute.numberOfVcpus : `N/A`}
  + Processor speed: {$compute.processorSpeed ? $compute.processorSpeed : `N/A`}
  + Operating system: {$compute.operatingSystem ? $compute.operatingSystem : `N/A`}
* Memory ({$compute.memoryType ? $compute.memoryType : `GB`}): { $compute.memoryAmount ? $compute.memoryAmount + ` ` + $compute.memoryUnit : `N/A`}
* Performance tier: {IF !$compute.performanceTier }: N/A{END-IF}

{IF $compute.performanceTier === `GENERAL`}

* + General Purpose (Provides a balance of compute, memory & network)

{END-IF}

{IF $compute.performanceTier === `COMPUTE`}

* + Compute Optimized (Supports compute-bound applications that benefit from high performance processors)

{END-IF}

{IF $compute.performanceTier === `MEMORY`}

* + Memory Optimized (Designed to deliver fast performance for workloads that process large data sets in memory)

{END-IF}

{IF $compute.performanceTier === `STORAGE`}

* + Storage Optimized (Designed for high, sequential read and write workloads to very large data sets on local storage)

{END-IF}

* Storage type and size ({$compute.storageUnit ? $compute.storageUnit : `GB`}): {IF !$compute.storageType} N/A {END-IF}

{IF $compute.storageType === `BLOCK`}

* + Block Storage (Fixed-sized raw storage capacity): {$compute.storageAmount} {$compute.storageUnit}

{END-IF}

{IF $compute.storageType === `OBJECT`}

* + Object Storage (Store and serve unstructured user-generated content): {$compute.storageAmount} {$compute.storageUnit}

{END-IF}

{IF $compute.storageType === `FILE`}

* + File Storage (Store and serve shared file systems): {$compute.storageAmount} {$compute.storageUnit}

{END-IF}

{IF $compute.storageType === `ARCHIVE`}

* + Archive Storage (Store and serve for long-term data retention): {$compute.storageAmount} {$compute.storageUnit}

{END-IF}

* Statement of objectives for anticipated need/usage:

{$compute.anticipatedNeedOrUsage}

{EXEC il6ComputeCount++}

{END-FOR compute}

{END-IF}

{EXEC il6DevTools = selectedServiceExists(selectedServices, selectedInstances, `DEVELOPER\_TOOLS`, `il6`) && il6Level}

**4.2.4.2** {il6Level && il6DevTools ? `Subtask 2 Developer Tools and Services (Unclassified - IL6)` : `Reserved`}

{IF il6DevTools}

{EXEC il6DevToolsCount = 1}

{FOR il6DevTool IN selectedInstances[`DEVELOPER\_TOOLS`].il6}

**4.2.4.2.{=il6DevToolsCount}** {$il6DevTool.name}

{$il6DevTool.usageDescription}

{EXEC il6DevToolsCount++}

{END-FOR il6DevTool}

{END-IF}

{EXEC il6AppTools = selectedServiceExists(selectedServices, selectedInstances, `APPLICATIONS`, `il6`) && il6Level }

**4.2.4.3** {il6AppTools ? `Subtask 3 Applications (Unclassified - IL6)` : `Reserved`}

{IF il6AppTools}

{EXEC il6AppToolsCount = 1}

{FOR il6AppTool IN selectedInstances[`APPLICATIONS`].il6}

**4.2.4.3.{=il6AppToolsCount}** {$il6AppTool.name}

{$il6AppTool.usageDescription}

{EXEC il6AppToolsCount++}

{END-FOR il6AppTool}

{END-IF}

{EXEC il6MlTools = selectedServiceExists(selectedServices, selectedInstances, `MACHINE\_LEARNING`, `il6`) && il6Level}

**4.2.4.4** {il6MlTools ? `Subtask 4 Advanced Technology and Algorithmic techniques (Machine Learning) and Data Analytics (Unclassified - IL6)` : `Reserved`}

{IF il6MlTools}

{EXEC il6MlToolsCount = 1}

{FOR il6MlTool IN selectedInstances[`MACHINE\_LEARNING`].il6}

**4.2.4.4.{=il6MlToolsCount}** {$il6MlTool.name}

{$il6MlTool.usageDescription}

{EXEC il6MlToolsCount++}

{END-FOR il6MlTool}

{END-IF}

{EXEC il6NetTools = selectedServiceExists(selectedServices, selectedInstances, `NETWORK`, `il6`) && il6Level}

**4.2.4.5** {il6NetTools ? `Subtask 5 Networking (Unclassified - IL6)` : `Reserved`}

{IF il6NetTools}

{EXEC il6NetToolsCount = 1}

{FOR il6NetTool IN selectedInstances[`NETWORK`].il6}

**4.2.4.5.{=il6NetToolsCount}** {$il6NetTool.name}

{$il6NetTool.usageDescription}

{EXEC il6NetToolsCount++}

{END-FOR il6NetTool}

{END-IF}

{EXEC il6SecTools = selectedServiceExists(selectedServices, selectedInstances, `SECURITY`, `il6`) && il6Level }

**4.2.4.6** {il6SecTools ? `Subtask 6 Security (Unclassified - IL6)` : `Reserved`}

{IF il6SecTools}

{EXEC il6SecToolsCount = 1}

{FOR il6SecTool IN selectedInstances[`SECURITY`].il6}

**4.2.4.6.{=il6SecToolsCount}** {$il6SecTool.name}

{$il6SecTool.usageDescription}

{EXEC il6SecToolsCount++}

{END-FOR il6SecTool}

{END-IF}

**4.2.4.7** {database.il6.length > 0 ? `Subtask 7 Database (Unclassified - IL6)` : `Reserved`}

{IF database.il6.length > 0}

{EXEC instancesCount = getInstancesCount(database.il6)}

Number of instances: {instancesCount.totalInstances}

{EXEC il6DatabaseCount = 1}

{FOR database IN database.il6}

**4.2.4.7.{=il6DatabaseCount}** Instance {=il6DatabaseCount}

* Number of instances with similar configurations: {$database.numberOfInstances ? $database.numberOfInstances : `N/A`}
* Approximate number of vCPUs/size of compute: {$database.numberOfVcpus ? $database.numberOfVcpus : `N/A`}
  + Processor speed: {$database.processorSpeed ? $database.processorSpeed : `N/A`}
  + Operating system: {$database.operatingSystem ? $database.operatingSystem : `N/A`}
* OS and licensing: {$database.operatingSystemLicensing ? formatEnum($database.operatingSystemLicensing) : `N/A`}
* DB and licensing: {$database.databaseLicensing ? formatEnum($database.databaseLicensing) : `N/A`}
* Memory ({$database.memoryType ? $database.memoryType : `GB`}): { $database.memoryAmount ? $database.memoryAmount + ` ` + $database.memoryUnit : `N/A`}
* Network performance: {$database.networkPerformance ? formatEnum($database.networkPerformance) : `N/A`}
* Storage type and size ({$database.storageUnit ? $database.storageUnit : `GB`}): {IF !$database.storageType} N/A {END-IF}

{IF $database.storageType === `BLOCK`}

* + Block Storage (Fixed-sized raw storage capacity): {$database.storageAmount} {$database.storageUnit}

{END-IF}

{IF $database.storageType === `OBJECT`}

* + Object Storage (Store and serve unstructured user-generated content): {$database.storageAmount} {$database.storageUnit}

{END-IF}

{IF $database.storageType === `FILE`}

* + File Storage (Store and serve shared file systems): {$database.storageAmount} {$database.storageUnit}

{END-IF}

{IF $database.storageType === `ARCHIVE`}

* + Archive Storage (Store and serve for long-term data retention): {$database.storageAmount} {$database.storageUnit}

{END-IF}

* Database Type: {$database.databaseType ? formatEnum($database.databaseType) : `N/A`}
* Statement of objectives for anticipated need/usage, to include the purpose and usage of the expected tools/capabilities:

{$database.anticipatedNeedOrUsage}

{EXEC il6DatabaseCount++}

{END-FOR database}

{END-IF}

**4.2.4.8** {storage.il6.length > 0 ? `Subtask 8 Storage (requirements separate from Compute/Database needs) (Unclassified - IL6)` : `Reserved`}

{IF storage.il6.length > 0}

{EXEC il6StorageCount = 1}

{FOR storage IN storage.il6}

**4.2.4.8.{=il6StorageCount}** Instance {=il6StorageCount}

* Number of instances with similar configurations: {$storage.numberOfInstances ? $storage.numberOfInstances : `N/A`}
* Storage type and size ({$storage.storageUnit ? $storage.storageUnit : `GB`}): {IF !$storage.storageType} N/A {END-IF}

{IF $storage.storageType === `BLOCK`}

* + Block Storage (Fixed-sized raw storage capacity): {$storage.storageAmount} {$storage.storageUnit}

{END-IF}

{IF $storage.storageType === `OBJECT`}

* + Object Storage (Store and serve unstructured user-generated content): {$storage.storageAmount} {$storage.storageUnit}

{END-IF}

{IF $storage.storageType === `FILE`}

* + File Storage (Store and serve shared file systems): {$storage.storageAmount} {$storage.storageUnit}

{END-IF}

{IF $storage.storageType === `ARCHIVE`}

* + Archive Storage (Store and serve for long-term data retention): {$storage.storageAmount} {$storage.storageUnit}

{END-IF}

* Statement of objectives for anticipated need/usage, to include the purpose and usage of the expected tools/capabilities:

{$storage.anticipatedNeedOrUsage}

{EXEC il6StorageCount++}

{END-FOR storage}

{END-IF}

{EXEC il6TeTools = selectedServiceExists(selectedServices, selectedInstances, `EDGE\_COMPUTING`, `il6`) && il6Level}

**4.2.4.9** {il6TeTools ? `Subtask 9 Edge Computing and Tactical Edge (TE) (Unclassified - IL6)` : `Reserved`}

{IF il6TeTools}

{EXEC il6TeToolsCount = 1}

{FOR il6TeTool IN selectedInstances[`EDGE\_COMPUTING`].il6}

**4.2.4.9.{=il6TeToolsCount}** {$il6TeTool.name} {IF $il6TeTool.name === `Mobility`}({$il6TeTool.typeOfMobility ? formatEnum($il6TeTool.typeOfMobility) : `N/A`}) {END-IF}

{IF $il6TeTool.name === `Delivery`}

* {$il6TeTool.typeOfDelivery ? formatEnum($il6TeTool.typeOfDelivery) : `N/A`}

{END-IF}

{IF $il6TeTool.otherServiceOffering}

Other

* {$il6TeTool.otherServiceOffering ? $il6TeTool.otherServiceOffering: `N/A`}

{END-IF}

{$il6TeTool.usageDescription}

{EXEC il6TeToolsCount++}

{END-FOR il6TeTool}

{END-IF}

{EXEC il6IotTools = selectedServiceExists(selectedServices, selectedInstances, `IOT`, `il6`) && il6Level}

**4.2.4.10** {il6IotTools ? `Subtask 10 Internet of Things (IoT) (Unclassified - IL6)` : `Reserved`}

{IF il6IotTools}

{EXEC il6IotToolsCount = 1}

{FOR il6IotTool IN selectedInstances[`IOT`].il6}

**4.2.4.10.{=il6IotToolsCount}** {$il6IotTool.name}

{$il6IotTool.usageDescription}

{EXEC il6IotToolsCount++}

{END-FOR il6IotTool}

{END-IF}

**4.2.4.11** {general.il6.length > 0 ? `General Infrastructure as a Service (IaaS), PaaS, and SaaS to include third party marketplace (not covered in Subtasks 4.2.4.1-4.2.4.10) (Unclassified - IL6)` : `Reserved`}

{IF general.il6.length > 0}

{EXEC il6GeneralCount = 1}

{FOR general IN general.il6}

**4.2.4.11.{=il6GeneralCount}** Instance {=il6GeneralCount}

* Statement of objectives for anticipated need/usage, to include the purpose and usage of the expected tools/capabilities:

{$general.anticipatedNeedOrUsage}

{EXEC il6GeneralCount++}

{END-FOR general}

{END-IF}

{END-IF}

**4.2.5** {tsLevel ? `Unclassified - TS` : `Reserved`}

{IF tsLevel}

Location(s) and approximate number of current users:

{IF tsLevel.usersPerRegion.length > 0}

{FOR regionUsers IN formatRegionUsers(tsLevel.usersPerRegion)}

* {$regionUsers} users

{END-FOR regionUsers}

{END-IF}

{IF tsLevel.usersIncrease || tsLevel.dataIncrease}

Anticipated future needs:

{IF !tsLevel.usersIncrease}

* Users: Static

{END-IF}

{IF tsLevel.usersIncrease}

{IF tsLevel.userGrowthEstimateType === `SINGLE`}

* Users: Estimated lifecycle growth: {tsLevel.userGrowthEstimatePercentage[0]}%

{END-IF}

{IF tsLevel.userGrowthEstimateType === `MULTIPLE`}

* Users: Estimated growth per period:

{FOR estimatedGrowth IN formatGrowthEstimates(tsLevel.userGrowthEstimatePercentage) }

* + {$estimatedGrowth}

{END-FOR estimatedGrowth}

{END-IF}

{END-IF}

{IF !tsLevel.dataIncrease}

* Data: Static

{END-IF}

{IF tsLevel.dataIncrease}

{IF tsLevel.dataGrowthEstimateType === `SINGLE`}

* Data: Estimated lifecycle growth: {tsLevel.dataGrowthEstimatePercentage[0]}%

{END-IF}

{IF tsLevel.dataGrowthEstimateType === `MULTIPLE`}

* Data: Estimated growth per period:

{FOR estimatedGrowth IN formatGrowthEstimates(tsLevel.dataGrowthEstimatePercentage) }

* + {$estimatedGrowth}

{END-FOR estimatedGrowth}

{END-IF}

{END-IF}

{END-IF}

**4.2.5.1** {compute.ts.length > 0 ? `Subtask 1 Compute (Unclassified - TS)` : `Reserved`}

{IF compute.ts.length > 0}

{EXEC instancesCount = getInstancesCount(compute.ts)}

Number of instances: {instancesCount.totalInstances}

* Dev/Test: {instancesCount.dev}
* Pre-Production: {instancesCount.preProd}
* Production/Staging: {instancesCount.prod}
* COOP/Disaster Recovery: {instancesCount.coop}

{EXEC tsComputeCount = 1}

{FOR compute IN compute.ts}

**4.2.5.1.{=tsComputeCount}** Instance {=tsComputeCount}

* Number of instances with similar configurations: {$compute.numberOfInstances ? $compute.numberOfInstances : `N/A`}
* Operating environment: {$compute.operatingEnvironment ? formatEnum($compute.operatingEnvironment) : `N/A`}
* OS and licensing: {$compute.operatingSystemLicensing ? formatEnum($compute.operatingSystemLicensing) : `N/A`}
* Approximate number of vCPUs/size of compute: {$compute.numberOfVcpus ? $compute.numberOfVcpus : `N/A`}
  + Processor speed: {$compute.processorSpeed ? $compute.processorSpeed : `N/A`}
  + Operating system: {$compute.operatingSystem ? $compute.operatingSystem : `N/A`}
* Memory ({$compute.memoryType ? $compute.memoryType : `GB`}): { $compute.memoryAmount ? $compute.memoryAmount + ` ` + $compute.memoryUnit : `N/A`}
* Performance tier: {IF !$compute.performanceTier }: N/A{END-IF}

{IF $compute.performanceTier === `GENERAL`}

* + General Purpose (Provides a balance of compute, memory & network)

{END-IF}

{IF $compute.performanceTier === `COMPUTE`}

* + Compute Optimized (Supports compute-bound applications that benefit from high performance processors)

{END-IF}

{IF $compute.performanceTier === `MEMORY`}

* + Memory Optimized (Designed to deliver fast performance for workloads that process large data sets in memory)

{END-IF}

{IF $compute.performanceTier === `STORAGE`}

* + Storage Optimized (Designed for high, sequential read and write workloads to very large data sets on local storage)

{END-IF}

* Storage type and size ({$compute.storageUnit ? $compute.storageUnit : `GB`}): {IF !$compute.storageType} N/A {END-IF}

{IF $compute.storageType === `BLOCK`}

* + Block Storage (Fixed-sized raw storage capacity): {$compute.storageAmount} {$compute.storageUnit}

{END-IF}

{IF $compute.storageType === `OBJECT`}

* + Object Storage (Store and serve unstructured user-generated content): {$compute.storageAmount} {$compute.storageUnit}

{END-IF}

{IF $compute.storageType === `FILE`}

* + File Storage (Store and serve shared file systems): {$compute.storageAmount} {$compute.storageUnit}

{END-IF}

{IF $compute.storageType === `ARCHIVE`}

* + Archive Storage (Store and serve for long-term data retention): {$compute.storageAmount} {$compute.storageUnit}

{END-IF}

* Statement of objectives for anticipated need/usage:

{$compute.anticipatedNeedOrUsage}

{EXEC tsComputeCount++}

{END-FOR compute}

{END-IF}

{EXEC tsDevTools = selectedServiceExists(selectedServices, selectedInstances, `DEVELOPER\_TOOLS`, `ts`) && tsLevel}

**4.2.5.2** {tsLevel && tsDevTools ? `Subtask 2 Developer Tools and Services (Unclassified - TS)` : `Reserved`}

{IF tsDevTools}

{EXEC tsDevToolsCount = 1}

{FOR tsDevTool IN selectedInstances[`DEVELOPER\_TOOLS`].ts}

**4.2.5.2.{=tsDevToolsCount}** {$tsDevTool.name}

{$tsDevTool.usageDescription}

{EXEC tsDevToolsCount++}

{END-FOR tsDevTool}

{END-IF}

{EXEC tsAppTools = selectedServiceExists(selectedServices, selectedInstances, `APPLICATIONS`, `ts`) && tsLevel }

**4.2.5.3** {tsAppTools ? `Subtask 3 Applications (Unclassified - TS)` : `Reserved`}

{IF tsAppTools}

{EXEC tsAppToolsCount = 1}

{FOR tsAppTool IN selectedInstances[`APPLICATIONS`].ts}

**4.2.5.3.{=tsAppToolsCount}** {$tsAppTool.name}

{$tsAppTool.usageDescription}

{EXEC tsAppToolsCount++}

{END-FOR tsAppTool}

{END-IF}

{EXEC tsMlTools = selectedServiceExists(selectedServices, selectedInstances, `MACHINE\_LEARNING`, `ts`) && tsLevel}

**4.2.5.4** {tsMlTools ? `Subtask 4 Advanced Technology and Algorithmic techniques (Machine Learning) and Data Analytics (Unclassified - TS)` : `Reserved`}

{IF tsMlTools}

{EXEC tsMlToolsCount = 1}

{FOR tsMlTool IN selectedInstances[`MACHINE\_LEARNING`].ts}

**4.2.5.4.{=tsMlToolsCount}** {$tsMlTool.name}

{$tsMlTool.usageDescription}

{EXEC tsMlToolsCount++}

{END-FOR tsMlTool}

{END-IF}

{EXEC tsNetTools = selectedServiceExists(selectedServices, selectedInstances, `NETWORK`, `ts`) && tsLevel}

**4.2.5.5** {tsNetTools ? `Subtask 5 Networking (Unclassified - TS)` : `Reserved`}

{IF tsNetTools}

{EXEC tsNetToolsCount = 1}

{FOR tsNetTool IN selectedInstances[`NETWORK`].ts}

**4.2.5.5.{=tsNetToolsCount}** {$tsNetTool.name}

{$tsNetTool.usageDescription}

{EXEC tsNetToolsCount++}

{END-FOR tsNetTool}

{END-IF}

{EXEC tsSecTools = selectedServiceExists(selectedServices, selectedInstances, `SECURITY`, `ts`) && tsLevel }

**4.2.5.6** {tsSecTools ? `Subtask 6 Security (Unclassified - TS)` : `Reserved`}

{IF tsSecTools}

{EXEC tsSecToolsCount = 1}

{FOR tsSecTool IN selectedInstances[`SECURITY`].ts}

**4.2.5.6.{=tsSecToolsCount}** {$tsSecTool.name}

{$tsSecTool.usageDescription}

{EXEC tsSecToolsCount++}

{END-FOR tsSecTool}

{END-IF}

**4.2.5.7** {database.ts.length > 0 ? `Subtask 7 Database (Unclassified - TS)` : `Reserved`}

{IF database.ts.length > 0}

{EXEC instancesCount = getInstancesCount(database.ts)}

Number of instances: {instancesCount.totalInstances}

{EXEC tsDatabaseCount = 1}

{FOR database IN database.ts}

**4.2.5.7.{=tsDatabaseCount}** Instance {=tsDatabaseCount}

* Number of instances with similar configurations: {$database.numberOfInstances ? $database.numberOfInstances : `N/A`}
* Approximate number of vCPUs/size of compute: {$database.numberOfVcpus ? $database.numberOfVcpus : `N/A`}
  + Processor speed: {$database.processorSpeed ? $database.processorSpeed : `N/A`}
  + Operating system: {$database.operatingSystem ? $database.operatingSystem : `N/A`}
* OS and licensing: {$database.operatingSystemLicensing ? formatEnum($database.operatingSystemLicensing) : `N/A`}
* DB and licensing: {$database.databaseLicensing ? formatEnum($database.databaseLicensing) : `N/A`}
* Memory ({$database.memoryType ? $database.memoryType : `GB`}): { $database.memoryAmount ? $database.memoryAmount + ` ` + $database.memoryUnit : `N/A`}
* Network performance: {$database.networkPerformance ? formatEnum($database.networkPerformance) : `N/A`}
* Storage type and size ({$database.storageUnit ? $database.storageUnit : `GB`}): {IF !$database.storageType} N/A {END-IF}

{IF $database.storageType === `BLOCK`}

* + Block Storage (Fixed-sized raw storage capacity): {$database.storageAmount} {$database.storageUnit}

{END-IF}

{IF $database.storageType === `OBJECT`}

* + Object Storage (Store and serve unstructured user-generated content): {$database.storageAmount} {$database.storageUnit}

{END-IF}

{IF $database.storageType === `FILE`}

* + File Storage (Store and serve shared file systems): {$database.storageAmount} {$database.storageUnit}

{END-IF}

{IF $database.storageType === `ARCHIVE`}

* + Archive Storage (Store and serve for long-term data retention): {$database.storageAmount} {$database.storageUnit}

{END-IF}

* Database Type: {$database.databaseType ? formatEnum($database.databaseType) : `N/A`}
* Statement of objectives for anticipated need/usage, to include the purpose and usage of the expected tools/capabilities:

{$database.anticipatedNeedOrUsage}

{EXEC tsDatabaseCount++}

{END-FOR database}

{END-IF}

**4.2.5.8** {storage.ts.length > 0 ? `Subtask 8 Storage (requirements separate from Compute/Database needs) (Unclassified - TS)` : `Reserved`}

{IF storage.ts.length > 0}

{EXEC tsStorageCount = 1}

{FOR storage IN storage.ts}

**4.2.5.8.{=tsStorageCount}** Instance {=tsStorageCount}

* Number of instances with similar configurations: {$storage.numberOfInstances ? $storage.numberOfInstances : `N/A`}
* Storage type and size ({$storage.storageUnit ? $storage.storageUnit : `GB`}): {IF !$storage.storageType} N/A {END-IF}

{IF $storage.storageType === `BLOCK`}

* + Block Storage (Fixed-sized raw storage capacity): {$storage.storageAmount} {$storage.storageUnit}

{END-IF}

{IF $storage.storageType === `OBJECT`}

* + Object Storage (Store and serve unstructured user-generated content): {$storage.storageAmount} {$storage.storageUnit}

{END-IF}

{IF $storage.storageType === `FILE`}

* + File Storage (Store and serve shared file systems): {$storage.storageAmount} {$storage.storageUnit}

{END-IF}

{IF $storage.storageType === `ARCHIVE`}

* + Archive Storage (Store and serve for long-term data retention): {$storage.storageAmount} {$storage.storageUnit}

{END-IF}

* Statement of objectives for anticipated need/usage, to include the purpose and usage of the expected tools/capabilities:

{$storage.anticipatedNeedOrUsage}

{EXEC tsStorageCount++}

{END-FOR storage}

{END-IF}

{EXEC tsTeTools = selectedServiceExists(selectedServices, selectedInstances, `EDGE\_COMPUTING`, `ts`) && tsLevel}

**4.2.5.9** {tsTeTools ? `Subtask 9 Edge Computing and Tactical Edge (TE) (Unclassified - TS)` : `Reserved`}

{IF tsTeTools}

{EXEC tsTeToolsCount = 1}

{FOR tsTeTool IN selectedInstances[`EDGE\_COMPUTING`].ts}

**4.2.5.9.{=tsTeToolsCount}** {$tsTeTool.name} {IF $tsTeTool.name === `Mobility`}({$tsTeTool.typeOfMobility ? formatEnum($tsTeTool.typeOfMobility) : `N/A`}) {END-IF}

{IF $tsTeTool.name === `Delivery`}

* {$tsTeTool.typeOfDelivery ? formatEnum($tsTeTool.typeOfDelivery) : `N/A`}

{END-IF}

{IF $tsTeTool.otherServiceOffering}

Other

* {$tsTeTool.otherServiceOffering ? $tsTeTool.otherServiceOffering: `N/A`}

{END-IF}

{$tsTeTool.usageDescription}

{EXEC tsTeToolsCount++}

{END-FOR tsTeTool}

{END-IF}

{EXEC tsIotTools = selectedServiceExists(selectedServices, selectedInstances, `IOT`, `ts`) && tsLevel}

**4.2.5.10** {tsIotTools ? `Subtask 10 Internet of Things (IoT) (Unclassified - TS)` : `Reserved`}

{IF tsIotTools}

{EXEC tsIotToolsCount = 1}

{FOR tsIotTool IN selectedInstances[`IOT`].ts}

**4.2.5.10.{=tsIotToolsCount}** {$tsIotTool.name}

{$tsIotTool.usageDescription}

{EXEC tsIotToolsCount++}

{END-FOR tsIotTool}

{END-IF}

**4.2.5.11** {general.ts.length > 0 ? `General Infrastructure as a Service (IaaS), PaaS, and SaaS to include third party marketplace (not covered in Subtasks 4.2.5.1-4.2.5.10) (Unclassified - TS)` : `Reserved`}

{IF general.ts.length > 0}

{EXEC tsGeneralCount = 1}

{FOR general IN general.ts}

**4.2.5.11.{=tsGeneralCount}** Instance {=tsGeneralCount}

* Statement of objectives for anticipated need/usage, to include the purpose and usage of the expected tools/capabilities:

{$general.anticipatedNeedOrUsage}

{EXEC tsGeneralCount++}

{END-FOR general}

{END-IF}

{END-IF}

{EXEC cdsRequired = crossDomainSolutions.crossDomainSolutionRequired}

**4.2.6** {cdsRequired ? `Cross-Domain Solutions (CDS)` : `Reserved`}

{IF cdsRequired}

CDS is required:

{FOR item IN crossDomainSolutions.trafficPerDomainPair}

{IF $item.type === "U\_TO\_S"}

* Unclassified to Secret (GB/mo): {INS $item.dataQuantity} GB

{END-IF}

{IF $item.type === "S\_TO\_U"}

* Secret to Unclassified (GB/mo): {INS $item.dataQuantity} GB

{END-IF}

{END-FOR item}

Projected file stream/type: {crossDomainSolutions.projectedFileStreamType}

Statement of objectives for anticipated need:

{crossDomainSolutions.anticipatedNeedOrUsage ? crossDomainSolutions.anticipatedNeedOrUsage : `N/A`}

{END-IF}

**4.3** Task 3 - Cloud Support Packages.

{EXEC supportPackages = sortedCloudSupportPackages}

{EXEC selectedPackages = Object.keys(supportPackages)}

{EXEC Port = supportPackages[`PORTABILITY\_PLAN`]}

{EXEC Adv = supportPackages[`ADVISORY\_ASSISTANCE`]}

{EXEC Help = supportPackages[`HELP\_DESK\_SERVICES`]}

{EXEC Train = supportPackages[`TRAINING`]}

{EXEC Doc = supportPackages[`DOCUMENTATION\_SUPPORT`]}

{EXEC Gen = supportPackages[`GENERAL\_CLOUD\_SUPPORT`]}

{EXEC il2Port = selectedServiceExists(selectedPackages, supportPackages, `PORTABILITY\_PLAN`, `il2`)}

{EXEC il2Adv = selectedServiceExists(selectedPackages, supportPackages, `ADVISORY\_ASSISTANCE`, `il2`)}

{EXEC il2Help = selectedServiceExists(selectedPackages, supportPackages, `HELP\_DESK\_SERVICES`, `il2`)}

{EXEC il2Train = selectedServiceExists(selectedPackages, supportPackages, `TRAINING`, `il2`)}

{EXEC il2Doc = selectedServiceExists(selectedPackages, supportPackages, `DOCUMENTATION\_SUPPORT`, `il2`)}

{EXEC il2Gen = selectedServiceExists(selectedPackages, supportPackages, `GENERAL\_CLOUD\_SUPPORT`, `il2`)}

{EXEC il2Section = il2Port || il2Adv || il2Help || il2Train || il2Doc || il2Gen}

**4.3.1** {il2Section **? `**Unclassified - IL2` : `Reserved`}

{IF !il2Section}

{END-IF}

{IF il2Section}

{EXEC il2AACount = 1}

{EXEC il2HDSCount = 1}

{EXEC il2TrainingCount = 1}

{EXEC il2DSCount = 1}

{EXEC il2GCSCount = 1}

{IF Port.il2.length > 0}

{FOR item IN Port.il2}

{$item.serviceType === `PORTABILITY` ? `A portability plan IAW for the basic JWCC Contract is required.` : `A portability plan IAW for the basic JWCC Contract is not required. ` }

{END-FOR item}

{END-IF}

**4.3.1.1** {il2Adv ? `Subtask 1 Advisory and assistance (Unclassified - IL2)` : `Reserved`}

{IF il2Adv}

{FOR item IN Adv.il2}

**4.3.1.1.{il2AACount}** Service {il2AACount}

* {$item.personnelOnsiteAccess ? `Services require CSP personnel to access on-site locations.` : `Services do not require CSP personnel to access on-site locations.`}
* Statement of objectives for anticipated need/usage, to include the desired outcome (purpose and usage) of the expected services: {INS $item.anticipatedNeedOrUsage}

{EXEC il2AACount++}

{END-FOR item}

{END-IF}

**4.3.1.2** {il2Help ? `Subtask 2 Help Desk Services (Unclassified - IL2)` : `Reserved`}

{IF il2Help}

{FOR item IN Help.il2}

**4.3.1.2.{il2HDSCount}** Service {il2HDSCount}

* {$item.personnelOnsiteAccess ? `Services require CSP personnel to access on-site locations.` : `Services do not require CSP personnel to access on-site locations.`}
* Statement of objectives for anticipated need/usage, to include the desired outcome (purpose and usage) of the expected services: {INS $item.anticipatedNeedOrUsage}

{EXEC il2HDSCount++}

{END-FOR item}

{END-IF}

**4.3.1.3** {il2Train ? `Subtask 3 Training (Unclassified - IL2)` : `Reserved`}

{IF il2Train}

{FOR item IN Train.il2}

**4.3.1.3.{il2TrainingCount}** Service {il2TrainingCount}

* Training required: {$item.trainingRequirementTitle ? $item.trainingRequirementTitle : `N/A`}
* Required Format: {$item.trainingFormat ? formatEnum($item.trainingFormat) : `N/A`}

{IF $item.trainingLocation}

* + Location: {$item.trainingLocation ? $item.trainingLocation : `N/A`}

{END-IF}

* Training shall be held at: {$item.trainingFacilityType ? $item.trainingFacilityType : `N/A`}
* Number of personnel requiring training: {$item.personnelRequiringTraining ? $item.personnelRequiringTraining + ` people` : `N/A`}
* Statement of objectives for anticipated need/usage, to include the desired outcome (purpose and usage) of the expected services: {INS $item.anticipatedNeedOrUsage}

{EXEC il2TrainingCount++}

{END-FOR item}

{END-IF}

**4.3.1.4** {il2Doc ? `Subtask 4 Documentation Support (Unclassified - IL2)` : `Reserved`}

{IF il2Doc}

{FOR item IN Doc.il2}

**4.3.1.4.{il2DSCount}** Service {il2DSCount}

* {$item.personnelOnsiteAccess ? `Services require CSP personnel to access on-site locations.` : `Services do not require CSP personnel to access on-site locations.`}
* Statement of objectives for anticipated need/usage, to include the desired outcome (purpose and usage) of the expected services: {INS $item.anticipatedNeedOrUsage}

{EXEC il2DSCount++}

{END-FOR item}

{END-IF}

**4.3.1.5** {il2Gen ? `Subtask 5 General Cloud Support (not covered in Subtasks 4.3.1.1 - 4.3.1.4)

(Unclassified - IL2)` : `Reserved`}

{IF il2Gen}

{FOR item IN Gen.il2}

**4.3.1.5.{il2GCSCount}** Service {il2GCSCount}

* {$item.personnelOnsiteAccess ? `Services require CSP personnel to access on-site locations.` : `Services do not require CSP personnel to access on-site locations.`}
* Statement of objectives for anticipated need/usage, to include the desired outcome (purpose and usage) of the expected services: {INS $item.anticipatedNeedOrUsage}

{EXEC il2GCSCount++}

{END-FOR item}

{END-IF}

{END-IF}

{EXEC il4Port = selectedServiceExists(selectedPackages, supportPackages, `PORTABILITY\_PLAN`, `il4`)}

{EXEC il4Adv = selectedServiceExists(selectedPackages, supportPackages, `ADVISORY\_ASSISTANCE`, `il4`)}

{EXEC il4Help = selectedServiceExists(selectedPackages, supportPackages, `HELP\_DESK\_SERVICES`, `il4`)}

{EXEC il4Train = selectedServiceExists(selectedPackages, supportPackages, `TRAINING`, `il4`)}

{EXEC il4Doc = selectedServiceExists(selectedPackages, supportPackages, `DOCUMENTATION\_SUPPORT`, `il4`)}

{EXEC il4Gen = selectedServiceExists(selectedPackages, supportPackages, `GENERAL\_CLOUD\_SUPPORT`, `il4`)}

{EXEC il4Section = il4Port || il4Adv || il4Help || il4Train || il4Doc || il4Gen}

**4.3.2** {il4Section **? `**Unclassified - IL4` : `Reserved`}

{IF !il4Section}

{END-IF}

{IF il4Section}

{EXEC il4AACount = 1}

{EXEC il4HDSCount = 1}

{EXEC il4TrainingCount = 1}

{EXEC il4DSCount = 1}

{EXEC il4GCSCount = 1}

{IF Port.il4.length > 0}

{FOR item IN Port.il4}

{$item.serviceType === `PORTABILITY` ? `A portability plan IAW for the basic JWCC Contract is required.` : `A portability plan IAW for the basic JWCC Contract is not required. ` }

{END-FOR item}

{END-IF}

**4.3.2.1** {il4Adv ? `Subtask 1 Advisory and assistance (Unclassified - IL4)` : `Reserved`}

{IF il4Adv}

{FOR item IN Adv.il4}

**4.3.2.1.{il4AACount}** Service {il4AACount}

* {$item.personnelOnsiteAccess ? `Services require CSP personnel to access on-site locations.` : `Services do not require CSP personnel to access on-site locations.`}
* Statement of objectives for anticipated need/usage, to include the desired outcome (purpose and usage) of the expected services: {INS $item.anticipatedNeedOrUsage}

{EXEC il4AACount++}

{END-FOR item}

{END-IF}

**4.3.2.2** {il4Help ? `Subtask 2 Help Desk Services (Unclassified - IL4)` : `Reserved`}

{IF il4Help}

{FOR item IN Help.il4}

**4.3.2.2.{il4HDSCount}** Service {il4HDSCount}

* {$item.personnelOnsiteAccess ? `Services require CSP personnel to access on-site locations.` : `Services do not require CSP personnel to access on-site locations.`}
* Statement of objectives for anticipated need/usage, to include the desired outcome (purpose and usage) of the expected services: {INS $item.anticipatedNeedOrUsage}

{EXEC il4HDSCount++}

{END-FOR item}

{END-IF}

**4.3.2.3** {il4Train ? `Subtask 3 Training (Unclassified - IL4)` : `Reserved`}

{IF il4Train}

{FOR item IN Train.il4}

**4.3.2.3.{il4TrainingCount}** Service {il4TrainingCount}

* Training required: {$item.trainingRequirementTitle ? $item.trainingRequirementTitle : `N/A`}
* Required Format: {$item.trainingFormat ? formatEnum($item.trainingFormat) : `N/A`}

{IF $item.trainingLocation}

* + Location: {$item.trainingLocation ? $item.trainingLocation : `N/A`}

{END-IF}

* Training shall be held at: {$item.trainingFacilityType ? $item.trainingFacilityType : `N/A`}
* Number of personnel requiring training: {$item.personnelRequiringTraining ? $item.personnelRequiringTraining + ` people` : `N/A`}
* Statement of objectives for anticipated need/usage, to include the desired outcome (purpose and usage) of the expected services: {INS $item.anticipatedNeedOrUsage}

{EXEC il4TrainingCount++}

{END-FOR item}

{END-IF}

**4.3.2.4** {il4Doc ? `Subtask 4 Documentation Support (Unclassified - IL4)` : `Reserved`}

{IF il4Doc}

{FOR item IN Doc.il4}

**4.3.2.4.{il4DSCount}** Service {il4DSCount}

* {$item.personnelOnsiteAccess ? `Services require CSP personnel to access on-site locations.` : `Services do not require CSP personnel to access on-site locations.`}
* Statement of objectives for anticipated need/usage, to include the desired outcome (purpose and usage) of the expected services: {INS $item.anticipatedNeedOrUsage}

{EXEC il4DSCount++}

{END-FOR item}

{END-IF}

**4.3.2.5** {il4Gen ? `Subtask 5 General Cloud Support (not covered in Subtasks 4.3.2.1 - 4.3.2.4)

(Unclassified - IL4)` : `Reserved`}

{IF il4Gen}

{FOR item IN Gen.il4}

**4.3.2.5.{il4GCSCount}** Service {il4GCSCount}

* {$item.personnelOnsiteAccess ? `Services require CSP personnel to access on-site locations.` : `Services do not require CSP personnel to access on-site locations.`}
* Statement of objectives for anticipated need/usage, to include the desired outcome (purpose and usage) of the expected services: {INS $item.anticipatedNeedOrUsage}

{EXEC il4GCSCount++}

{END-FOR item}

{END-IF}

{END-IF}

{EXEC il5Port = selectedServiceExists(selectedPackages, supportPackages, `PORTABILITY\_PLAN`, `il5`)}

{EXEC il5Adv = selectedServiceExists(selectedPackages, supportPackages, `ADVISORY\_ASSISTANCE`, `il5`)}

{EXEC il5Help = selectedServiceExists(selectedPackages, supportPackages, `HELP\_DESK\_SERVICES`, `il5`)}

{EXEC il5Train = selectedServiceExists(selectedPackages, supportPackages, `TRAINING`, `il5`)}

{EXEC il5Doc = selectedServiceExists(selectedPackages, supportPackages, `DOCUMENTATION\_SUPPORT`, `il5`)}

{EXEC il5Gen = selectedServiceExists(selectedPackages, supportPackages, `GENERAL\_CLOUD\_SUPPORT`, `il5`)}

{EXEC il5Section = il5Port || il5Adv || il5Help || il5Train || il5Doc || il5Gen}

**4.3.3** {il5Section ? `Unclassified - IL5` : `Reserved`}

{IF !il5Section}

{END-IF}

{IF il5Section}

{EXEC il5AACount = 1}

{EXEC il5HDSCount = 1}

{EXEC il5TrainingCount = 1}

{EXEC il5DSCount = 1}

{EXEC il5GCSCount = 1}

{IF Port.il5.length > 0}

{FOR item IN Port.il5}

{$item.serviceType === `PORTABILITY` ? `A portability plan IAW for the basic JWCC Contract is required.` : `A portability plan IAW for the basic JWCC Contract is not required. ` }

{END-FOR item}

{END-IF}

**4.3.3.1** {il5Adv ? `Subtask 1 Advisory and assistance (Unclassified - IL5)` : `Reserved`}

{IF il5Adv}

{FOR item IN Adv.il5}

**4.3.3.1.{il5AACount}** Service {il5AACount}

* {$item.personnelOnsiteAccess ? `Services require CSP personnel to access on-site locations.` : `Services do not require CSP personnel to access on-site locations.`}
* Statement of objectives for anticipated need/usage, to include the desired outcome (purpose and usage) of the expected services: {INS $item.anticipatedNeedOrUsage}

{EXEC il5AACount++}

{END-FOR item}

{END-IF}

**4.3.3.2** {il5Help ? `Subtask 2 Help Desk Services (Unclassified - IL5)` : `Reserved`}

{IF il5Help}

{FOR item IN Help.il5}

**4.3.3.2.{il5HDSCount}** Service {il5HDSCount}

* {$item.personnelOnsiteAccess ? `Services require CSP personnel to access on-site locations.` : `Services do not require CSP personnel to access on-site locations.`}
* Statement of objectives for anticipated need/usage, to include the desired outcome (purpose and usage) of the expected services: {INS $item.anticipatedNeedOrUsage}

{EXEC il5HDSCount++}

{END-FOR item}

{END-IF}

**4.3.3.3** {il5Train ? `Subtask 3 Training (Unclassified - IL5)` : `Reserved`}

{IF il5Train}

{FOR item IN Train.il5}

**4.3.3.3.{il5TrainingCount}** Service {il5TrainingCount}

* Training required: {$item.trainingRequirementTitle ? $item.trainingRequirementTitle : `N/A`}
* Required Format: {$item.trainingFormat ? formatEnum($item.trainingFormat) : `N/A`}

{IF $item.trainingLocation}

* + Location: {$item.trainingLocation ? $item.trainingLocation : `N/A`}

{END-IF}

* Training shall be held at: {$item.trainingFacilityType ? $item.trainingFacilityType : `N/A`}
* Number of personnel requiring training: {$item.personnelRequiringTraining ? $item.personnelRequiringTraining + ` people` : `N/A`}
* Statement of objectives for anticipated need/usage, to include the desired outcome (purpose and usage) of the expected services: {INS $item.anticipatedNeedOrUsage}

{EXEC il5TrainingCount++}

{END-FOR item}

{END-IF}

**4.3.3.4** {il5Doc ? `Subtask 4 Documentation Support (Unclassified - IL5)` : `Reserved`}

{IF il5Doc}

{FOR item IN Doc.il5}

**4.3.3.4.{il5DSCount}** Service {il5DSCount}

* {$item.personnelOnsiteAccess ? `Services require CSP personnel to access on-site locations.` : `Services do not require CSP personnel to access on-site locations.`}
* Statement of objectives for anticipated need/usage, to include the desired outcome (purpose and usage) of the expected services: {INS $item.anticipatedNeedOrUsage}

{EXEC il5DSCount++}

{END-FOR item}

{END-IF}

**4.3.3.5** {il5Gen ? `Subtask 5 General Cloud Support (not covered in Subtasks 4.3.3.1 - 4.3.3.4)

(Unclassified - IL5)` : `Reserved`}

{IF il5Gen}

{FOR item IN Gen.il5}

**4.3.3.5.{il5GCSCount}** Service {il5GCSCount}

* {$item.personnelOnsiteAccess ? `Services require CSP personnel to access on-site locations.` : `Services do not require CSP personnel to access on-site locations.`}
* Statement of objectives for anticipated need/usage, to include the desired outcome (purpose and usage) of the expected services: {INS $item.anticipatedNeedOrUsage}

{EXEC il5GCSCount++}

{END-FOR item}

{END-IF}

{END-IF}

{EXEC il6Port = selectedServiceExists(selectedPackages, supportPackages, `PORTABILITY\_PLAN`, `il6`)}

{EXEC il6Adv = selectedServiceExists(selectedPackages, supportPackages, `ADVISORY\_ASSISTANCE`, `il6`)}

{EXEC il6Help = selectedServiceExists(selectedPackages, supportPackages, `HELP\_DESK\_SERVICES`, `il6`)}

{EXEC il6Train = selectedServiceExists(selectedPackages, supportPackages, `TRAINING`, `il6`)}

{EXEC il6Doc = selectedServiceExists(selectedPackages, supportPackages, `DOCUMENTATION\_SUPPORT`, `il6`)}

{EXEC il6Gen = selectedServiceExists(selectedPackages, supportPackages, `GENERAL\_CLOUD\_SUPPORT`, `il6`)}

{EXEC il6Section = il6Port || il6Adv || il6Help || il6Train || il6Doc || il6Gen}

**4.3.4** {il6Section ? **`**Unclassified - IL6` : `Reserved`}

{IF !il6Section}

{END-IF}

{IF il6Section}

{EXEC il6AACount = 1}

{EXEC il6HDSCount = 1}

{EXEC il6TrainingCount = 1}

{EXEC il6DSCount = 1}

{EXEC il6GCSCount = 1}

{IF Port.il6.length > 0}

{FOR item IN Port.il6}

{$item.serviceType === `PORTABILITY` ? `A portability plan IAW for the basic JWCC Contract is required.` : `A portability plan IAW for the basic JWCC Contract is not required. ` }

{END-FOR item}

{END-IF}

**4.3.4.1** {il6Adv ? `Subtask 1 Advisory and assistance (Unclassified - IL6)` : `Reserved`}

{IF il6Adv}

{FOR item IN Adv.il6}

**4.3.4.1.{il6AACount}** Service {il6AACount}

* {$item.personnelOnsiteAccess ? `Services require CSP personnel to access on-site locations.` : `Services do not require CSP personnel to access on-site locations.`}
* Statement of objectives for anticipated need/usage, to include the desired outcome (purpose and usage) of the expected services: {INS $item.anticipatedNeedOrUsage}

{EXEC il6AACount++}

{END-FOR item}

{END-IF}

**4.3.4.2** {il6Help ? `Subtask 2 Help Desk Services (Unclassified - IL6)` : `Reserved`}

{IF il6Help}

{FOR item IN Help.il6}

**4.3.4.2.{il6HDSCount}** Service {il6HDSCount}

* {$item.personnelOnsiteAccess ? `Services require CSP personnel to access on-site locations.` : `Services do not require CSP personnel to access on-site locations.`}
* Statement of objectives for anticipated need/usage, to include the desired outcome (purpose and usage) of the expected services: {INS $item.anticipatedNeedOrUsage}

{EXEC il6HDSCount++}

{END-FOR item}

{END-IF}

**4.3.4.3** {il6Train ? `Subtask 3 Training (Unclassified - IL6)` : `Reserved`}

{IF il6Train}

{FOR item IN Train.il6}

**4.3.4.3.{il6TrainingCount}** Service {il6TrainingCount}

* Training required: {$item.trainingRequirementTitle ? $item.trainingRequirementTitle : `N/A`}
* Required Format: {$item.trainingFormat ? formatEnum($item.trainingFormat) : `N/A`}

{IF $item.trainingLocation}

* + Location: {$item.trainingLocation ? $item.trainingLocation : `N/A`}

{END-IF}

* Training shall be held at: {$item.trainingFacilityType ? $item.trainingFacilityType : `N/A`}
* Number of personnel requiring training: {$item.personnelRequiringTraining ? $item.personnelRequiringTraining + ` people` : `N/A`}
* Statement of objectives for anticipated need/usage, to include the desired outcome (purpose and usage) of the expected services: {INS $item.anticipatedNeedOrUsage}

{EXEC il6TrainingCount++}

{END-FOR item}

{END-IF}

**4.3.4.4** {il6Doc ? `Subtask 4 Documentation Support (Unclassified - IL6)` : `Reserved`}

{IF il6Doc}

{FOR item IN Doc.il6}

**4.3.4.4.{il6DSCount}** Service {il6DSCount}

* {$item.personnelOnsiteAccess ? `Services require CSP personnel to access on-site locations.` : `Services do not require CSP personnel to access on-site locations.`}
* Statement of objectives for anticipated need/usage, to include the desired outcome (purpose and usage) of the expected services: {INS $item.anticipatedNeedOrUsage}

{EXEC il6DSCount++}

{END-FOR item}

{END-IF}

**4.3.4.5** {il6Gen ? `Subtask 5 General Cloud Support (not covered in Subtasks 4.3.4.1 - 4.3.4.4)

(Unclassified - IL6)` : `Reserved`}

{IF il6Gen}

{FOR item IN Gen.il6}

**4.3.4.5.{il6GCSCount}** Service {il6GCSCount}

* {$item.personnelOnsiteAccess ? `Services require CSP personnel to access on-site locations.` : `Services do not require CSP personnel to access on-site locations.`}
* Statement of objectives for anticipated need/usage, to include the desired outcome (purpose and usage) of the expected services: {INS $item.anticipatedNeedOrUsage}

{EXEC il6GCSCount++}

{END-FOR item}

{END-IF}

{END-IF}

{EXEC tsPort = selectedServiceExists(selectedPackages, supportPackages, `PORTABILITY\_PLAN`, `ts`)}

{EXEC tsAdv = selectedServiceExists(selectedPackages, supportPackages, `ADVISORY\_ASSISTANCE`, `ts`)}

{EXEC tsHelp = selectedServiceExists(selectedPackages, supportPackages, `HELP\_DESK\_SERVICES`, `ts`)}

{EXEC tsTrain = selectedServiceExists(selectedPackages, supportPackages, `TRAINING`, `ts`)}

{EXEC tsDoc = selectedServiceExists(selectedPackages, supportPackages, `DOCUMENTATION\_SUPPORT`, `ts`)}

{EXEC tsGen = selectedServiceExists(selectedPackages, supportPackages, `GENERAL\_CLOUD\_SUPPORT`, `ts`)}

{EXEC tsSection = tsPort || tsAdv || tsHelp || tsTrain || tsDoc || tsGen}

**4.3.5** {tsSection ? `Unclassified - TS` : `Reserved`}

{IF !tsSection}

{END-IF}

{IF tsSection}

{EXEC tsAACount = 1}

{EXEC tsHDSCount = 1}

{EXEC tsTrainingCount = 1}

{EXEC tsDSCount = 1}

{EXEC tsGCSCount = 1}

{IF Port.ts.length > 0}

{FOR item IN Port.ts}

{$item.serviceType === `PORTABILITY` ? `A portability plan IAW for the basic JWCC Contract is required.` : `A portability plan IAW for the basic JWCC Contract is not required. ` }

{END-FOR item}

{END-IF}

**4.3.5.1** {tsAdv ? `Subtask 1 Advisory and assistance (Unclassified - TS)` : `Reserved`}

{IF tsAdv}

{FOR item IN Adv.ts}

**4.3.5.1.{tsAACount}** Service {tsAACount}

* {$item.personnelOnsiteAccess ? `Services require CSP personnel to access on-site locations.` : `Services do not require CSP personnel to access on-site locations.`}
* Statement of objectives for anticipated need/usage, to include the desired outcome (purpose and usage) of the expected services: {INS $item.anticipatedNeedOrUsage}

{EXEC tsAACount++}

{END-FOR item}

{END-IF}

**4.3.5.2** {tsHelp ? `Subtask 2 Help Desk Services (Unclassified - TS)` : `Reserved`}

{IF tsHelp}

{FOR item IN Help.ts}

**4.3.5.2.{tsHDSCount}** Service {tsHDSCount}

* {$item.personnelOnsiteAccess ? `Services require CSP personnel to access on-site locations.` : `Services do not require CSP personnel to access on-site locations.`}
* Statement of objectives for anticipated need/usage, to include the desired outcome (purpose and usage) of the expected services: {INS $item.anticipatedNeedOrUsage}

{EXEC tsHDSCount++}

{END-FOR item}

{END-IF}

**4.3.5.3** {tsTrain ? `Subtask 3 Training (Unclassified - TS)` : `Reserved`}

{IF tsTrain}

{FOR item IN Train.ts}

**4.3.5.3.{tsTrainingCount}** Service {tsTrainingCount}

* Training required: {$item.trainingRequirementTitle ? $item.trainingRequirementTitle : `N/A`}
* Required Format: {$item.trainingFormat ? formatEnum($item.trainingFormat) : `N/A`}

{IF $item.trainingLocation}

* + Location: {$item.trainingLocation ? $item.trainingLocation : `N/A`}

{END-IF}

* Training shall be held at: {$item.trainingFacilityType ? $item.trainingFacilityType : `N/A`}
* Number of personnel requiring training: {$item.personnelRequiringTraining ? $item.personnelRequiringTraining + ` people` : `N/A`}
* Statement of objectives for anticipated need/usage, to include the desired outcome (purpose and usage) of the expected services: {INS $item.anticipatedNeedOrUsage}

{EXEC tsTrainingCount++}

{END-FOR item}

{END-IF}

**4.3.5.4** {tsDoc ? `Subtask 4 Documentation Support (Unclassified - TS)` : `Reserved`}

{IF tsDoc}

{FOR item IN Doc.ts}

**4.3.5.4.{tsDSCount}** Service {tsDSCount}

* {$item.personnelOnsiteAccess ? `Services require CSP personnel to access on-site locations.` : `Services do not require CSP personnel to access on-site locations.`}
* Statement of objectives for anticipated need/usage, to include the desired outcome (purpose and usage) of the expected services: {INS $item.anticipatedNeedOrUsage}

{EXEC tsDSCount++}

{END-FOR item}

{END-IF}

**4.3.5.5** {tsGen ? `Subtask 5 General Cloud Support (not covered in Subtasks 4.3.5.1 - 4.3.5.4)

(Unclassified - TS)` : `Reserved`}

{IF tsGen}

{FOR item IN Gen.ts}

**4.3.5.5.{tsGCSCount}** Service {tsGCSCount}

* {$item.personnelOnsiteAccess ? `Services require CSP personnel to access on-site locations.` : `Services do not require CSP personnel to access on-site locations.`}
* Statement of objectives for anticipated need/usage, to include the desired outcome (purpose and usage) of the expected services: {INS $item.anticipatedNeedOrUsage}

{EXEC tsGCSCount++}

{END-FOR item}

{END-IF}

{END-IF}

**5.** Contract Data Requirements Lists (CDRL). The below CDRLs are included in the Joint Warfighting Cloud Capability (JWCC) Contract.

{EXEC timeAndMaterials = contractType.timeAndMaterials}

{EXEC firmFixed = contractType.firmFixedPrice}

{EXEC cdrlTable = dataRequirementsList(cloudSupportPackages,il2Level,il4Level,il5Level,il6Level,il2EdgeCount,il4EdgeCount,il5EdgeCount,il6EdgeCount,contractType)}

|  |  |  |  |
| --- | --- | --- | --- |
| **DoW Task Number** | **JWCC Contract CLIN Number** | **CDRL** | **CDRL Name** |
| {FOR data IN cdrlTable} |  |  |  |
| {$data.dowTaskNumbers} | {$data.clinNumbers} | {$data.cdrl.code} | {$data.cdrl.name} |
| {END-FOR data} |  |  |  |

{FOR data IN cdrlTable}

{IF $data.cdrl.name === "System Administrator Training Materials"}

{="\*CDRLs A004 AND A005 are required for each individual training session"}

{END-IF}

{IF $data.cdrl.name === "Portability Plan"}

{="\*\* CDRL A006 requires both XaaS and advisory and assistance services"}

{END-IF}

{IF $data.cdrl.name === "TE Device Specifications"}

{="\*\*\* CDRL A017 is required only for the initial TE delivery and redelivery if a TE specification changes"}

{END-IF}

{END-FOR data}

**6.** Performance Standards. The performance standards are established in the JWCC IDIQ Contract.

**7.** Period of Performance.

{popPeriods}

|  |  |
| --- | --- |
| **{FOR header IN popTableHeaders}** | **{$header === `B` ? `Base` : $header === `OP1` ? `OP1` : $header}{END-FOR header}** |
| {FOR row IN popTableBody} |  |
| {FOR mark IN $row} | {$mark}{END-FOR mark} |
| {END-FOR row} |  |

**8.** Security Requirements. The security requirements and compliance mandates are established in the JWCC IDIQ Contract DD254.

{IF sr.isSecurityNeeded === false}

Reserved

{END-IF}

{IF sr.isSecurityNeeded}

|  |  |  |  |
| --- | --- | --- | --- |
| **Task / Subtask** | **Clearance Level** | **Level of Classified Access** | **Justification for Access to Classified** |
| {IF sr.currentEnvSecret.length >= 1} |  |  |  |
| Provide SECRET Cloud Services and Support in performance of task 4.1 | SECRET | {sr.currentEnvSecret.join(`, `)} | Access is required in the offering and support of SECRET cloud services and support to the JWCC Contract. Access to SCI caveats and information, and SAPs is required. |
| {END-IF} |  |  |  |
| {IF sr.xaasSecret.compute.length >= 1} |  |  |  |
| {FOR service IN sr.xaasSecret.compute} |  |  |  |
| Provide SECRET Cloud Services and Support in performance of task {$service.taskNumber} | SECRET | {$service.access.join(`, `)} | Access is required in the offering and support of SECRET cloud services and support to the JWCC Contract. |
| {END-FOR service} |  |  |  |
| {END-IF} |  |  |  |
| {IF sr.xaasSecret.database.length >= 1} |  |  |  |
| {FOR service IN sr.xaasSecret.database} |  |  |  |
| Provide SECRET Cloud Services and Support in performance of task {$service.taskNumber} | SECRET | {$service.access.join(`, `)} | Access is required in the offering and support of SECRET cloud services and support to the JWCC Contract. |
| {END-FOR service} |  |  |  |
| {END-IF} |  |  |  |
| {IF sr.xaasSecret.storage.length >= 1} |  |  |  |
| {FOR service IN sr.xaasSecret.storage} |  |  |  |
| Provide SECRET Cloud Services and Support in performance of task {$service.taskNumber} | SECRET | {$service.access.join(`, `)} | Access is required in the offering and support of SECRET cloud services and support to the JWCC Contract. |
| {END-FOR service} |  |  |  |
| {END-IF} |  |  |  |
| {IF sr.xaasSecret.general.length >= 1} |  |  |  |
| {FOR service IN sr.xaasSecret.general} |  |  |  |
| Provide SECRET Cloud Services and Support in performance of task {$service.taskNumber} | SECRET | {$service.access.join(`, `)} | Access is required in the offering and support of SECRET cloud services and support to the JWCC Contract. |
| {END-FOR service} |  |  |  |
| {END-IF} |  |  |  |
| {IF sr.xaasSecret.selectedService.length >= 1} |  |  |  |
| {FOR service IN sr.xaasSecret.selectedService} |  |  |  |
| Provide SECRET Cloud Services and Support in performance of task {$service.taskNumber} | SECRET | {$service.access.join(`, `)} | Access is required in the offering and support of SECRET cloud services and support to the JWCC Contract. |
| {END-FOR service} |  |  |  |
| {END-IF} |  |  |  |
| {IF sr.cloudSupportSecret} |  |  |  |
| Cloud Training in performance of task 4.3.4.3 | SECRET | {sr.cloudSupportSecret.join(`, `)} | Access is required in support of cloud training requirements across the contract. |
| {END-IF} |  |  |  |

{END-IF}

**9.** Government Furnished Property (GFP)/Government-Furnished Equipment (GFE)/Government-Furnished Information (GFI). All compliance requirements for managing GFP, GFE, and GFI are established in the JWCC IDIQ Contract.

**10.** Other Pertinent Information or Special Considerations.

1. Identification of Potential Conflicts of Interest (COI).{IF !contractConsiderations.potentialConflictOfInterest} None. {END-IF}

{IF contractConsiderations.potentialConflictOfInterest}

{contractConsiderations.conflictOfInterestExplanation}

{END-IF}

1. Packaging, Packing, and Shipping Instructions.

* When transferring physical media between locations, the contractor shall provide a certified courier or other method of maintaining a secure chain of custody over the physical media being moved to and from a defined, secured off-site storage location. The contractor shall provide flexibility in courier pick-up and delivery time.

{IF contractConsiderations.packagingShippingOther}

* {contractConsiderations.packagingShippingOther ? contractConsiderations.packagingShippingOtherExplanation : `N/A`}

{END-IF}

{IF contractConsiderations.packagingShippingOtherNoneApply}

* None apply.

{END-IF}

1. Supply Chain Risk Management (SCRM). All applicable SCRM requirements are listed in the JWCC IDIQ Contract.
2. Training. Contractor employees may be required to take periodic mandatory training courses provided by a Federal Government organization, such as records management training and other training required by statute, regulation, DoD, or local (e.g. DISA) policy. No other training of contractor personnel shall be provided by the Government unless authorized by the Contracting Officer.
3. Personally Identifiable Information (PII). {contractConsiderations.piiPresent ? `This requirement provides for the design, development, or operation of a system of records on individuals (in whole or in part).` : ` This requirement does not provide for the design, development, or operation of a system of records on individuals (in whole or in part).`}

{IF contractConsiderations.piiPresent}

* System of records: {contractConsiderations.systemOfRecordName}

{END-IF}

1. Travel. Number of trips: {contractConsiderations.travel.length}

|  |  |  |  |
| --- | --- | --- | --- |
| Number of trips | Location | Duration | Number of Travelers |
| {FOR trip IN contractConsiderations.travel} |  |  |  |
| {$trip.numberOfTrips} | {$trip.tripLocation} | {$trip.durationInDays} days | {$trip.numberOfTravelers} travelers |
| {END-FOR trip} |  |  |  |

**11.** Section 508 Accessibility Standards for Cloud Computing. {sensitiveInformation.section508Sufficient ? ` All applicable Section 508 requirements are listed in the JWCC IDIQ Contract.` : ``}

{IF !sensitiveInformation.section508Sufficient}

No. { sensitiveInformation.accessibilityReqs508}

{END-IF}