NOTE: The header contains tokens that will automatically update when you process this document in Veeva. **DO NOT EDIT THE HEADER**. This template contains suggested/example language in blue text and Track Changes is fully functional. Please remove all blue text language when creating a controlled document. N/A any sections that are not applicable.

1. PURPOSE

The purpose of this work instruction is to provide guidance on repeatable security steps that should be followed during factory or site acceptance testing phases of all operational projects.

2. SCOPE

This work instruction applies to all automation, control system and instrumentation equipment for facilities, utilities, engineering, manufacturing, physical security, building automation and adjacent automation and control systems applications, infrastructure and networks that directly support those systems and devices.

3. DEFINITIONS

Provide a definition for all acronyms and terms included in the work instruction and arrange them alphabetically.

Term	Definition
BAS	Building Automation System
BMS	Building Management System
EMS	Environmental or Energy Management System
DCS	Distributed Control System
FAT	Factory Acceptance Test
FBD	Function Block Diagram
FRS	Functional Requirement Specification
GRC	Governance, Risk, Compliance
HDS	Hardware Design Specification
HMI	Human Machine Interface
ICS	Industrial Control System
IL	Instruction List
IO	Input/Output module
IQ	Installation Qualification
LD	Ladder Logic/Diagram
MES	Manufacturing Execution System
OEM	Original Equipment Manufacturer
OS	Operating System
OT	Operational Technology
OQ	Operational Qualification
OWASP	Open Web Application Security Project
P&ID	Piping & Instrumentation Diagram
PFD	Process Flow Diagram
PLC	Programmable Logic Controller

Term	Definition
PQ	Performance Qualification
RTOS	Real-time Operating Systems
SAT	Site Acceptance Test
SCADA	Supervisory Control and Data Acquisition
SCAP	Security Content Automation Protocol
SCC	STIG SCAP Compliance Checker
SCL	Structure Control Language
SDS	System Design Specification
SFC	Sequential Function Chart
SRS	System Requirements Specification
ST	Structure Text
STIG	Security Technical Implementation Guide
URS	User Requirement Specification
VFD	Variable Frequency Drive
VLAN	Virtual Local Area Network
VM	Virtual Machine
VSD	Variable Speed Drive

4. EQUIPMENT AND MATERIALS

4.1. Some of the equipment and materials that will be needed consist of but are not limited to as follows:

- Laptops or Workstations
- Virtual Machines
- Network equipment and connection peripherals (e.g., converter gateways, switches, firewalls, conduit connectors etcetera)
- Software related to the equipment, systems, and applications in scope of the FAT and SAT
- PLC/controller
- HMI or tablets
- VFD
- VSD
- Security tools (e.g., ICS OT protocol specific tools, scanning, testing and exploitation tools)
- Documentation for all equipment, systems, networks, applications, procedures etcetera in scope of FAT and SAT (e.g., P&ID, PFD, panel drawings, HDS, SDS, SRS, user manuals, equipment spec sheets etcetera)

5. SAFETY

When working on ICS OT assets with electrical, mechanical, chemical, and biological related equipment, following existing safety policies, procedures and best practices is recommended. If suspected unsafe conditions exist in the process or scope of conducting a security FAT/SAT then work should be stopped immediately and suspected safety issues should be elevated to local site, regional or global environmental, health and safety staff for guidance and support prior to the continuation of work.

6. PROCEDURE

Through product feature checks, vulnerability scans, pentesting, fuzzing and exploit testing project teams will discover both known and potentially unknown vulnerabilities in ICS systems and devices throughout their lifecycle. Factory and Site acceptance testing (FAT/SAT), systems testing and validation, gate reviews etcetera are very critical check points during an ICS project prior to commissioning and turn over to production operations where correcting security weaknesses in a system is the most optimal time to do so.

Note: *In some cases, some tools, checks, and tests may only be possible during the design build phase or during plant shutdown, redesign and retrofit phases due to the intrusive nature of the some of the tests and tools involved. Thus, it is very important for security FAT/SAT to be included in the project schedule, scope, budget, staffing, tools, and time.*

6.1. Overview

ICS OT Security Engineers, IT support, Automation Engineers, System Integrators, OEMs, and contractors need to perform several security due diligence steps during FAT and SAT. Some of these steps may also need to be performed during IQ, OQ, PQ especially in cases where a comprehensive security FAT/SAT was not in scope or was not possible due to staff, budget, scope, and time.

Some of the steps and approaches consist of but are not limited to the following:

●Checking configurations for firewalls, routers, switches, gateways, access points, conduit/protocol converters, servers, workstations, laptops, PLCs, RTUs, HMIs, VFDs and other assets within the package/project/production scope – leveraging tools such as PowerShell, bash command line, Windows System Internals, Nipper etc.

● Network discovery, foot-printing and enumerations scans using tools such as Nmap, p0f, arp-scan, ncat, netcat, softperfect network scanner etc.

● Vulnerability scans using tools such as Nessus, Nexpose, Qualys, OpenVAS

● Sniffing sample traffic and analyzing protocols with tools such as TCPdump, Networkminer, Wireshark, Security Onion w/ ELK, TenableOT/Indegy, Claroty etc.

● Review architecture design for weaknesses in security zones and conduits

● Code, software, protocol, application, and hardware Vulnerability analysis using tools such as w3af, Burp Suite, WebScarab, OWASP ZAP, PeachFuzzer, SonarQube, Postman etc.

● Attempting basic exploits and stress testing such as protocol manipulation, fuzzing etc. with tools such as SCADA Pack+, Metasploit, Kali, PeachFuzzer, Nmap, custom scripts etc.

● Produce a listing and ranking of key findings with their associated potential consequences, potential impacts, and mitigation recommendations for each assessed asset within the package/project/production scope of the FAT/SAT.

● Ensuring list of findings and recommended mitigations are also mapped back to the System and Device level security controls in ISA/IEC 62443-3-3 and 4-2 and any applicable ICS security program controls in the asset owner target state design from ISA/IEC 62443-2-1, US DHS ICS security best practices and other applicable standards, guidelines, practices and regulations.

The daily FAT, SAT punch list should be updated with some of the following information for tracking security checks, findings, recommendations, completed items for the day etcetera.

ITEM #	Asset Type	Asset Name	Collection/ Review/ Tested Date	Collected/ Reviewed /Tested By	Provided To/Witnessed by	IP Address	MAC Address	Protocol	Station Name	Comments/ Collection type/Fix Recommendation/Open issue
1										
2										
3										
4										
5										

Note: *Security FAT/SAT reviews are only a snapshot in time during design build projects. Unresolved security issues found during FAT and SAT that cannot be resolved by the type IQ, OQ, PQ and Commissioning are complete for production turnover should all be reported on the Digital Security GRC form here.*

6.2. Configuration Feature Review

There are X numbers of PLCs, VFDs, workstations, HMIs, laptops, servers, firewalls, routers, switches, gateways, and conduit/protocol converters that will need to be reviewed during an ICS Cybersecurity FAT/SAT.

Project Teams (e.g. PM, production, maintenance, facilities, engineering, utilities, manufacturing, automation, QA, QC etcetera) will be responsible for working with ICS OT Security Engineers and 3rd parties to ensure we can collect and access all configurable features and configuration files for ICS OT security reviews, analysis, testing, verification and attestation.

For firewalls, gateways, switches, protocol, or conduit converters access to and copies of current configs and logs will be required. Network and local access to equipment, systems, applications and hosts for ICS OT Security Engineer tools, laptops etcetera is required.

For HMIs, tablets, workstations, laptops and VM a temporary local admin account for ICS OT Security will need to be provided in order for ICS OT Security Engineers, Automation Engineers, 3rd parties or IT support to run various security tools, such as SCC to verify windows host settings against the applicable windows OS STIG, for example.

Note: *STIG scan XCCDF files should be uploaded to OpenRMF portal tool for collaborative review and mitigation on specific STIG settings flagged as failed. The OpenRMF runbook should be followed for access and usage procedures.*

For field devices such as PLCs, VFDs, pumps etcetera, the project team will need to provide access to the programming logic, configurations, settings, applications, specifications, P&ID, network diagrams, line diagrams, process flow diagrams, software files etcetera. Tools used for fuzzing, review of code for security practices such as I/O validation, vulnerability scanning tools etcetera will be used as part of looking for unknown zero day and known vulnerabilities in field devices.

6.2.1. Operating Systems

For reviewing features and settings on operating systems usage of the temporary ICS OT Security Admin account will be needed to run certain commands and export certain information for offline security analysis, security review, and copies for attestation of verification of features and settings meeting or not meeting desired security state.

For Windows some of the items to check and commands to run include but are not limited to:

- Windows command line:
  - netstat -abnov > applicationname_OSverion_VM_netstat.txt
  - ipconfig /all > applicationname_OSverion_VM_ipconfigall.txt
  - systeminfo > applicationname_OSverion_VM_systeminfo.txt
- Windows Powershell command line:
  - Get-Process | Out-File applicationname_OSverion_VM_ProcessList.txt
  - Get-Service | Out-File applicationname_OSverion_VM_ProcessList.txt
- Windows items to export for review:
  - Full system registry text file dump named applicationname_OSversion_VM_registry.txt
  - Export of Windows host logs such as (e.g., Security Events Logs, Application Event Logs, Powershell Event logs, System Event Logs etcetera)
  - Run STIG scan checker tools (e.g., OpenSCAP) and upload the results to OpenRMF for collaborative review
  - Review local host user and group policy settings (e.g., including password policies, user permissions etcetera)

For Linux related operating systems there are various Linux distributions. Some of the common distributions found in ICS OT and example commands that should be run include but are not limited to:

- Ifconfig – get IP config information
- ls /etc./init.d/ - get a picture of all start/stop and other scripts
- ls /etc./passwd - get information on users
- List all installed software/packages
    - Red Hat/Fedora Core/CentOS Linux
        - # rpm -qa | less
    - Debian Linux
        - # dpkg --get-selections
    - Ubuntu Linux
        - # sudo dpkg --get-selections
    - FreeBSD
        - # pkg_info | less
        - # pkg_info apache
        - # pkg_version | less
        - # pkg_version | grep 'lsof'
    - OpenBSD
        - # pkg_info | less
        - # pkg_info apache
- For Real-time Operating Systems (RTOS) such as VxWorks, QNX, Nucleus, Greenhill Integrity, pOS and others some research may be needed to locate specific commands that can be used if terminal level interface access is granted or possible on field devices such as PLCs, VFD, VSD.

Examples of some VxWorks commands:

Command	Description
i	Show information on all running tasks
cd	Change Directory; just like UNIX cd
pwd	Print Working Directory; just like UNIX pwd
ls	List directory contents
ll	Long listing of directory contents; like UNIX "ls -l"
ld	Load a module
sp(FUNCPTR func, int arg1..9)	Spawns a new task with the entry point given by func, passing the given arguments to the function (arg1..9 optional)

devs	List all devices
inetstatShow	Show network connections (like netstat on UNIX)
memShow(int type)	Show free memory summary; additional details if argument is non-zero

6.2.2. Network Devices

For reviewing features of routers, switches, protocol/conduit converters, gateways, firewalls, access points and wireless radios etcetera gaining access to command line terminals and configuration tools should suffice for reviewing settings for weaknesses.

Example for Cisco based switches using the “show run” command will dump all the settings of interest that can be exported and then run into a security tool such as Nipper-ng using Kali Linux that runs security reviews against the existing settings for best practices.

Some of the items to check regardless of vendor product type include but are not limited to:

- Proper configuration of VLANs to facilitate security zone micro segmentation for east-west traffic
- Firewall rule features for ICS OT protocols (e.g., Palo Alto app ID for BACnet, ENIP, profinet, OPC UA, Modbus etcetera)
- User accounts
- Services and IT protocol interfaces that are commonly exploited (e.g., SMB, Telnet, DNS, HTTP, NetBIOS, SNMP, FTP etcetera). Only the secure versions or secure alternative protocols should be used (e.g., HTTPS, SCP, Secure FTP etc.) and all insecure protocols should be disabled and blocked by design.
-

6.3. Security Scanning

During FAT and SAT running scanning tools that enumerate common attacker tools and practices such as reconnaissance, enumeration, vulnerability scanning etcetera should be run.

For general network scanning and enumeration one of the most common tools is Nmap. Some of the Nmap commands that should be run include but are not limited to:

- nmap -T4 -O -A -vvv IP address
- nmap -Su -vvv IP address
- nmap -sS -vvv IP address
- nmap -sT -vvv IP address
- nmap -sV -vvv IP address

- use -Pn for no ping and -F for fragmenting
- --script to run scripts like SSL, RPC, RDP, SQL, ICS scripts and DoS scripts
- --version-intensity (light or 1 to 5) for aggression level
- To filter output into text file for slice and dice analysis leverage Nmap (- oA for file outputs in all formats, parse up hosts – cat filename | grep "Status:Up" > filename.txt) (port scans of up hosts)

Some other network scanning tools that can be used for reconnaissance and enumeration also include tools like softperfect network scanner.

For vulnerability scanning tools such as Nessus and OpenVAS should be used against equipment, systems, networks, and applications in scope to discover, document and review known vulnerabilities.

Note: For native protocol and device feature testing from a security perspective it is recommended to also include using protocol query and explorer tools. Some tools such as YABE for BACnet, Profinet Commander for Profinet, OPC UA and DA explorer tools etcetera are good examples. These tools are native tools that technicians and engineers use to troubleshoot. These same tools can be used from an attacker reconnaissance or manipulation perspective. Thus, it is good to leverage them during a FAT and SAT from a security perspective to understand what an attacker both insider threat and external actors could see and manipulate. Below are some of the many tools that can be used for this part of the test. This is not an exhaustive list.

6.1. PeachFuzzer

Download peach fuzzer here http://www.peach.tech/resources/peachcommunity/

6.2. OWASP ZAP

Download ZAP here https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project

6.3. Burp Suite

Download Burp here https://portswigger.net/burp/communitydownload

6.4. w3af

download w3af here http://w3af.org/download
http://docs.w3af.org/en/latest/

6.5. Kali

Many of the above tools are also available within the Kali suite of tools (includes free version of Metasploit)
Download Kali here https://www.kali.org/downloads/
https://www.kali.org/kali-linux-documentation/

6.6. ControlThings IO

https://www.controlthings.io/platform

6.7. SCADA+ Pack

SCADA+ Pack Exploit kit - http://gleg.net/agora_scada.shtml

6.8. Network Sniffing & Protocol Analysis

It is good ICS OT security practice to have applications, systems, networks, and equipment running during FAT and SAT to simulate how it would be used in production. When production simulation with sample data occurs the ICS OT Security engineer should use tools such as but not limited to:

- Wireshark
- Network Miner
- Grassmarlin
- Indegy (Tenable.OT)
- TCP Dump
- Claroty
- Nozomi

These tools will allow the ICS OT Security Engineer to capture ICS OT protocol level attestation, threat analysis, attack mapping for types of expected protocols, protocol commands, objects, tags and function codes and component communications or connections in use. This will also give a snapshot in time emulated view of what an attacker could potentially observe or capture on the network level between components.

Note: *For maximum effect it is best to do an unfiltered capture with Wireshark and then dump the full PCAP into Indegy, Network Miner and Grassmarlin etc for their various use cases so that the other tools do not pre-filter out information the ICS OT security engineer may want to be aware of.*

6.9. System & Network Architecture Design Review

For architecture design review ICS cybersecurity review all engineering diagrams (e.g. P&ID, PFD, line diagrams, wiring diagrams, as-builts etcetera) both drafts of to be state and current officially signed off engineering diagrams available during the gate reviews, validation, FAT and SAT. Review architecture for security zones, conduits, bad practices for ICS specifically such as no DMZs and segmentation between various subsystems and zones as well as connections to IT and internet without going through the ICS DMZ. Additionally, ensure that the ICS zones have their own security stack and separate Active Directory, Work group or identity and authentication domain from the DMZ, IT, the internet, and third-party zones.

Note: *This review should look for and push to improve compliance with the desired to be Security Architecture playbook and requirements.*

6.10. ICS OT Asset Inventory Review

The best time to know what you have and what attackers could target is to capture a detailed inventory and bill of materials during design build phases of projects. At a minimum, the follow information but not limited to should be captured in inventory sheets, tag lists, IO lists and bill of material sheets.

- Protocols (e.g., OPC UA, Profinet, ENIP, BACnet, Modbus, telnet, SNMP, SMB, NetBIOS, DNS, HTTPS etcetera)
- Services
- Ports
- Tags (e.g., read, write, bool, float etcetera)
- Conduits (e.g., fiber, ethernet, serial, fieldbus, RF/wireless)
- RTOS version (e.g., Windows CE, Windows IoT, VxWorks, Nucleus, QNX etcetera)
- OS (Windows 11, Server 2019, Ubuntu, CentOS etcetera)
- Firmware version
- Software version
- Application modules/containers/component versions
- Brand make and model (e.g., Siemens PXC, Rockwell Control Logix etc)
- Asset type (e.g., PLC, VFD, application, IPC, IED, HMI, protocol or conduit converter, router, switch, relay etcetera)
- Site
- Custodian/Engineer/System Owner
- IP address
- MAC address
- Station Name/Number
- I/O slot number or ID
- Host Name

6.11. Programming & Code Review

Reviewing PLC, VFD, VSD, controller, and application logic and code during the design build phases of projects is often the best time prior to production turnover to capture and correct security issues.

Note: Leverage code backup, recovery, change control tools like (e.g., Copia, AMDT Octoplant, FactoryTalk Asset Center etc) to track, review, approve and update PLC, RTU, Relay, VFD, HMI logic and code. Also leverage the programming IDE tools for one off reviews during FAT and SAT (e.g., Codesys, TIA Portal, RS Logix, Arduino IDE, etc)

6.11.1. PLC Coding Practices

PLC/controller programming logic languages are different than traditional IT based software development languages. IEC 61131-3 is the international standard that defines what ladder logic

(LD), function block diagram (FBD), sequential function chart (SFC), instruction list (IL) and structure text (ST) are. A global ICS OT security community of volunteers collaborated to vote on top 20 PLC secure coding practices that mostly apply to LD, FBD, SFC and IL.

The latest version of the top 20 PLC practices can always be found on the community site here: https://plc-security.com/

The detailed pdf that explains each practice in detail and maps them to security attack tactics, techniques and international ICS security standards can be downloaded in multiple languages from the site.

At a summary level the top 20 include the following practices:

1. Modularize PLC Code
Split PLC code into modules, using different function blocks (sub-routines). Test modules independently.

2. Track operating modes
Keep the PLC in RUN mode. If PLCs are not in RUN mode, there should be an alarm to the operators.

3. Leave operational logic in the PLC wherever feasible
Leave as much operational logic e.g., totalizing or integrating, as possible directly in the PLC. The HMI does not get enough updates to do this well.

4. Use PLC flags as integrity checks
Put counters on PLC error flags to capture any math problems.

5. Use cryptographic and / or checksum integrity checks for PLC code
Use cryptographic hashes, or checksums if cryptographic hashes are unavailable, to check PLC code integrity and raise an alarm when they change.

6. Validate timers and counters
If timers and counters values are written to the PLC program, they should be validated by the PLC for reasonableness and verify backward counts below zero.

7. Validate and alert for paired inputs / outputs
If you have paired signals, ensure that both signals are not asserted together. Alarm the operator when input / output states occur that are physically not feasible. Consider making paired signals independent or adding delay timers when toggling outputs could be damaging to actuators.

8. Validate HMI input variables at the PLC level, not only at HMI

HMI access to PLC variables can (and should) be restricted to a valid operational value range at the HMI, but further cross-checks in the PLC should be added to prevent, or alert on, values outside of the acceptable ranges which are programmed into the HMI.

9. Validate indirections
Validate indirections by poisoning array ends to catch fence-post errors.

10. Assign designated register blocks by function (read/write/validate)
Assign designated register blocks for specific functions to validate data, avoid buffer overflows and block unauthorized external writes to protect controller data.

11. Instrument for plausibility checks
Instrument the process in a way that allows for plausibility checks by cross-checking different measurements.

12. Validate inputs based on physical plausibility
Ensure operators can only input what is practical or physically feasible in the process. Set a timer for an operation to the duration it should physically take. Consider alerting when there are deviations. Also alert when there is unexpected inactivity.

13. Disable unneeded / unused communication ports and protocols
PLC controllers and network interface modules support multiple communication protocols that are enabled by default. Disable ports and protocols that are not required for the application.

14. Restrict third-party data interfaces
Restrict the type of connections and available data for 3rd party interfaces. The connections and/or data interfaces should be well defined and restricted to only allow read/write capabilities for the required data transfer.

15. Define a safe process state in case of a PLC restart
Define safe states for the process in case of PLC restarts (e.g., energize contacts, de-energize, keep previous state).

16. Summarize PLC cycle times and trend them on the HMI
Summarize PLC cycle time every 2-3 seconds and report to HMI for visualization on a graph.

17. Log PLC uptime and trend it on the HMI
Log PLC uptime to know when it has been restarted. Trend and log up time on the HMI for diagnostics.

18. Log PLC hard stops and trend them on the HMI
Store PLC hard stop events from faults or shutdowns for retrieval by HMI alarm systems to consult before PLC restarts. Time sync for more accurate data.

19. Monitor PLC memory usage and trend it on the HMI
Measure and provide a baseline for memory usage for every controller deployed in the production environment and trend it on the HMI.

20. Trap false negatives and false positives for critical alerts
Identify critical alerts and program a trap for those alerts. Set the trap to monitor the trigger conditions and the alert state for any deviation.

These practices should be reviewed, tested, and documented as applicable with the vendors, integrators, and internal staff responsible for programming controllers throughout a plant.

Note: *Not all 20 practices always apply in all cases. It is best to document examples for attestation of which specific practices in the 20 are used on which controllers for a process, a skid, a field panel, or critical operational functions especially any safety functions.*

6.11.2. Application & Structure Text Review

Structure text and higher-level languages such as C, Python, Java, and other languages used by vendors such as Siemens Structure Control Language (SCL) based on Pascal, all have traditional IT attack surfaces. Traditional IT secure coding practices such as those created by OWASP top 10 and used in static and or dynamic security code review tools should be used in cases where structured text languages are in use. This is particularly important when using python for scripting in SCADA for example.

For ICS during FAT and SAT phases of projects it is best to use static analysis tools. One recommended tool is SonarQube. It checks against some of the OWASP top 10 and generates a security report that also provides recommended fixes for specific lines of code. For the latest top 10 practices visit OWASP site here:
https://owasp.org/Top10/

Note: for API and IoT testing leverage the OWASP checklists and tools like Postman, OWASP ZAP, Burp suite, Nmap, Nessus, Nikto and others that can check for API and IoT specific top 10 issues.

6.12. Stress Testing, Exploitation & Fuzzing

During stress testing and fuzzing the project team will provide ICS cybersecurity with at least one of each type of asset that ICS cybersecurity can run denial of service, protocol manipulation, buffer overflow, credential escalation and other more aggressive testing against to test the robustness of each asset type. For this testing ICS cybersecurity only requires at least one of each type of asset and application in the system design scope of the validation, FAT and SAT such as one of each brand of firewall, router, switch, gateway, converter, access point, HMI, PLC/RTU, VFD, sensor, actuator, engineering workstation, Historian, laptop, Operating Systems, RTOS etcetera.

Below are some of the many tools that can be used for this part of the testing include but are not limited to:

- PeachFuzzer http://www.peach.tech/resources/peachcommunity/
- OWASP ZAP https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project
- Burp Suite https://portswigger.net/burp/communitydownload
- w3af http://w3af.org/download
- Kali https://www.kali.org/downloads/
  Note: many tools mentioned in this work instruction can be run from within a virtual Kali image. It is one of the most common open source offensive exploitation platforms.
- ControlThings IO https://www.controlthings.io/platform
- SCADA+ Pack Exploit kit - http://gleg.net/agora_scada.shtml

7. REFERENCES

7.1. List of documents (SOPs, other WIs, forms, logs, equipment manuals, etc.) referenced in the body of the Work Instruction.

7.2. List documents alphabetically by document number and include document number and title for controlled documents.

Document No.	Document Title

8. RECORDS

Digital Security, Digital Automation, Digital Site Ops, Site Facilities, engineering, manufacturing, and automation should have copies of testing and finding results, mitigations, and open items stored in collaborative share folders (e.g., box, onedrive, google drive) for the project in scope of the FAT or SAT.

Note: For compliance mapping and traceability needs, leverage dynamic documentation and workflow collaboration tools like Kneat Validation suite that works well with online and offline dynamic generation and linking specification and requirements docs with testing and quality procedures etc. This is great for real-time collaboration between suppliers, customers, end users, contractors, and internal staff. Also great for scaling reusable templates to baseline a mature level of design and testing rigor to meet compliance needs.

9. APPENDICES

9.1. Use appendices to provide information that clarifies or illustrates the procedure. This information may be presented in the form of diagrams, drawings, flow charts, pictures, tables, and/or text. List all the appendices included in the procedure as follows:

9.2. Title of Appendix 1

9.3. Title of Appendix 2

10. REVISION HISTORY

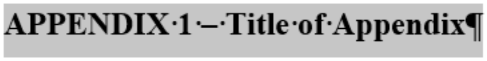
Use Revision History to provide a high-level overview of what has changed between the last version and the current version. For a detailed look at this change, Document Control can access all previous versions and detailed metadata. If this document is revised due to a regulatory commitment, Revision History will state this and call out the specific section that is revised. When creating a document please remove these instructions**.**

Note: if using dynamic documentation creation and tracking tools (e.g., Kneat Validation suite) the tools will track all changes, versions, revisions, notes of why changes, date and who changed etc. It can be included in workflows and approvals via assigned roles and user groups etc. It is best to avoid overly manual document approaches in the interest of rigor, quality, compliance, traceability, and speed.

Version No.	Description of Change	Reason for Change
1.0	New Document	N/A

APPENDIX 1 – Title of Appendix

NOTE: The above APPENDIX Title uses a built-in Word style [Appendix Title] that includes an automatic page break before the title. To remove this Appendix heading and its associated page-break, select the entire text-string including the space after 'Appendix' as shown in the following screenshot and hit Delete.



APPENDIX·1·–·Title·of·Appendix¶

Be sure to remove this blue-text and screenshot if no Appendix is needed.