directly support cyber resiliency as described in Appendix G. However, [SP 800-53] does not address all approaches or all aspects of any individual approach. Therefore, some examples are drawn from system reliability and system resilience practices and technologies, and/or from emerging cyber resiliency technologies. The set of approaches for a specific technique is not exhaustive and represents relatively mature technologies and practices. Thus, technologies emerging from research can be characterized in terms of the techniques they apply, while not being covered by any of the representative approaches.

TABLE E-4: CYBER RESILIENCY APPROACHES

TECHNIQUES	APPROACHES	EXAMPLES
Adaptive Response Implement agile courses of action to manage risks.	**Dynamic Reconfiguration** Make changes to individual systems, system elements, components, or sets of cyber resources to change functionality or behavior without interrupting service.	• Dynamically change router rules, access control lists, intrusion detection and prevention system parameters, and filter rules for firewalls and gateways. • Re-assign cyber defense responsibilities to personnel or operating centers.
	Dynamic Resource Allocation Change the allocation of resources to tasks or functions without terminating critical functions or processes.	• Employ dynamic provisioning. • Reprioritize messages or services. • Implement load-balancing. • Provide emergency shutoff capabilities. • Pre-empt communications.
	Adaptive Management Change how mechanisms are used based on changes in the operational environment as well as changes in the threat environment.	• Disable access dynamically. • Implement adaptive authentication. • Provide for automatic disabling of the system. • Provide dynamic deployment of new or replacement resources or capabilities.
Analytic Monitoring Monitor and analyze a wide range of properties and behaviors on an ongoing basis and in a coordinated way.	**Monitoring and Damage Assessment** Monitor and analyze behavior and characteristics of components and resources to look for indicators of adversary activity and to detect and assess damage from adversity.	• Use hardware fault detection. • Employ Continuous Diagnostics and Mitigation (CDM) or other vulnerability scanning tools. • Deploy Intrusion Detection Systems (IDSs) and other monitoring tools. • Use Insider Threat monitoring tools. • Perform telemetry analysis. • Detect malware beaconing. • Monitor open-source information for indicators of disclosure or compromise.
	Sensor Fusion and Analysis Fuse and analyze monitoring data and analysis results from different information sources or at different times together with externally provided threat intelligence.	• Enable organization-wide situational awareness. • Implement cross-organizational auditing. • Correlate data from different tools. • Fuse data from physical access control systems and information systems.

TECHNIQUES	APPROACHES	EXAMPLES
	Forensic and Behavioral Analysis Analyze adversary TTPs, including observed behavior as well as malware and other artifacts left behind by adverse events.	• Deploy an integrated team of forensic and malware analysts, developers, and operations personnel. • Use reverse engineering and other malware analysis tools.
Coordinated Protection Ensure that protection mechanisms operate in a coordinated and effective manner.	**Calibrated Defense-in-Depth** Provide complementary protective mechanisms at different architectural layers or in different locations, calibrating the strength and number of mechanisms to resource value.	• Design for defense-in-depth. • Employ multiple, distinct authentication challenges over the course of a session to confirm identity. • Combine network and host-based intrusion detection. • Provide increasing levels of protection to access more sensitive or critical resources. • Conduct sensitivity and criticality analyses.
	Consistency Analysis Determine whether and how protections can be applied in a coordinated, consistent way that minimizes interference, potential cascading failures, or coverage gaps.	• Employ unified Identity and Access Management (IdAM) administration tools. • Analyze mission/business process flows and threads. • Employ privilege analysis tools to support an ongoing review of whether user privileges are assigned consistently. • Interpret attributes consistently. • Coordinate the planning, training, and testing of incident response, contingency planning, etc. • Design for facilitating coordination and mutual support among safeguards.
	Orchestration Coordinate the ongoing behavior of mechanisms and processes at different layers, in different locations, or implemented for different aspects of trustworthiness to avoid causing cascading failures, interference, or coverage gaps.	• Coordinate incident handling with mission/business process continuity of operations and organizational processes. • Conduct coverage planning and management for sensors. • Use cyber playbooks.
	Self-Challenge Affect mission/business processes or system elements adversely in a controlled manner to validate the effectiveness of protections and to enable proactive response and improvement.	• Hardware power-on self-test. • Conduct role-based training exercises. • Conduct penetration testing and Red Team exercises. • Test automated incident response. • Employ fault injection. • Conduct tabletop exercises.
Contextual Awareness Construct and maintain current representations of the posture of missions or business	**Dynamic Resource Awareness** Maintain current information about resources, status of resources, and resource connectivity.	• Maintain real-time integrated situational awareness.

TECHNIQUES	APPROACHES	EXAMPLES
functions considering threat events and courses of action.	**Dynamic Threat Awareness** Maintain current information about threat actors, indicators, and potential, predicted, and observed adverse events.	• Track predicted or impending natural disasters. • Dynamically ingest incident and threat data. • Facilitate integrated situational awareness of threats.
	Mission Dependency and Status Visualization Maintain current information about the status of missions or business functions, dependencies on resources, and the status of those resources with respect to threats.	• Construct a broad (mission/business function-wide, organization-wide) perspective.
Deception Mislead, confuse, hide critical assets from, or expose covertly tainted assets to the adversary.	**Obfuscation** Hide, transform, or otherwise obfuscate information from the adversary.	• Encrypt data at rest. • Encrypt transmitted data (e.g., using a Virtual Private Network [VPN]). • Encrypt authenticators. • Conceal or randomize communications patterns. • Conceal the presence of system components on an internal network. • Mask, encrypt, hash, or replace identifiers. • Obfuscate traffic via onion routing. • Apply chaffing to communications traffic. • Add a large amount of valid but useless information to a data store. • Perform encrypted processing.
	Disinformation Provide deliberately misleading information to adversaries.	• Post questions to a public forum based on false information about the system. • Create false (“canary”) credentials and tokens (e.g., honeytokens).
	Misdirection Maintain deception resources or environments and direct adversary activities there.	• Establish and maintain honeypots, honeynets, or decoy files. • Maintain a full-scale, all-encompassing deception environment.
	Tainting Embed covert capabilities in resources.	• Use beacon traps. • Employ internal network table cache poisoning (e.g., Domain Name System (DNS), Address Resolution Protocol (ARP)). • Include false entries or steganographic data in files to enable them to be found via open-source analysis.
Diversity Use heterogeneity to minimize common mode failures, particularly threat events exploiting common vulnerabilities.	**Architectural Diversity** Use multiple sets of technical standards, different technologies, and different architectural patterns.	• Use auditing/logging systems on different OSs to acquire and store audit/logging data. • Apply different audit/logging regimes at different architectural layers. • Deploy diverse operating systems. • Support multiple protocol standards.

TECHNIQUES	APPROACHES	EXAMPLES
	Design Diversity Use different designs to meet the same requirements or provide equivalent functionality.	• Employ N-version programming. • Employ mixed-signal design diversity (using both analog and digital signals). • Employ mixed-level design diversity (using both hardware and software implementations).
	Synthetic Diversity Transform implementations of software to produce a variety of instances.	• Implement address space layout randomization. • Use randomizing compilers.
	Information Diversity Provide information from different sources or transform information in different ways.	• Apply different analog-to-digital conversion methods to non-digitally-obtained data. • Use multiple data sources.
	Path Diversity Provide multiple independent paths for command, control, and communications.	• Establish alternate telecommunications services (e.g., ground-based circuits, satellite communications). • Employ alternate communications protocols. • Use out-of-band channels.
	Supply Chain Diversity Use multiple independent supply chains for critical components.	• Use a diverse set of suppliers.
Dynamic Positioning Distribute and dynamically relocate functionality or system resources.	**Functional Relocation of Sensors** Relocate sensors or reallocate responsibility for specific sensing tasks to look for indicators of adverse events.	• Relocate (using virtualization) or reconfigure IDSs or IDS sensors.
	Functional Relocation of Cyber Resources Change the location of cyber resources that provide functionality or information, either by moving the assets or by transferring functional responsibility.	• Change processing locations (e.g., switch to a virtual machine on a different physical component). • Change storage sites (e.g., switch to an alternate data store on a different storage area network).
	Asset Mobility Securely move physical resources.	• Move a mobile device or system component (e.g., a router) from one room in a facility to another while monitoring its movement. • Move storage media securely from one room or facility to another room or facility. • Move a platform or vehicle to avoid collision or other physical harm, while retaining knowledge of its location.
	Fragmentation Fragment information and distribute it across multiple components.	• Implement fragmentation and partitioning for distributed databases.
	Distributed Functionality Decompose a function or application into smaller functions and distribute those functions across multiple components.	• Architect applications so that constituent functions can be located on different system components.

TECHNIQUES	APPROACHES	EXAMPLES
Non-Persistence Generate and retain resources as needed or for a limited time.	**Non-Persistent Information** Refresh information periodically, or generate information on demand, and delete it when no longer needed.	• Delete high value mission information after it is processed. • Off-load audit records to off-line storage. • Use one-time passwords or nonces.
	Non-Persistent Services Refresh services periodically, or generate services on demand and terminate services when no longer needed.	• Employ time-based or inactivity-based session termination. • Re-image components. • Refresh services using virtualization.
	Non-Persistent Connectivity Establish connections on demand, and terminate connections when no longer needed.	• Implement software-defined networking. • Employ time-based or inactivity-based network disconnection.
Privilege Restriction Restrict privileges based on attributes of users and system elements as well as on environmental factors.	**Trust-Based Privilege Management** Define, assign, and maintain privileges associated with active entities based on established trust criteria consistent with principles of least privilege.	• Implement least privilege. • Employ time-based account restrictions.
	Attribute-Based Usage Restriction Define, assign, maintain, and apply usage restrictions on systems containing cyber resources based on the criticality of missions or business functions and other attributes (e.g., data sensitivity).	• Employ Role-Based Access Control (RBAC). • Employ Attribute-Based Access Control (ABAC). • Restrict the use of maintenance tools.
	Dynamic Privileges Elevate or decrease privileges assigned to a user, process, or service based on transient or contextual factors.	• Implement time-based adjustment to privileges due to status of mission or business tasks. • Employ dynamic account provisioning. • Disable privileges based on a determination that an individual or process is high-risk. • Implement dynamic revocation of access authorizations. • Implement dynamic association of attributes with cyber resources and active entities. • Implement dynamic credential binding.
Realignment Align system resources with current organizational mission or business function needs to reduce risk.	**Purposing** Ensure systems containing cyber resources are used consistently with mission or business function purposes and approved uses.	• Use whitelisting to prevent installation of such unapproved applications as games or peer-to-peer music sharing. • Use whitelisting to restrict communications to a specified set of addresses. • Ensure that privileged accounts are not used for non-privileged functions.
	Offloading Offload supportive but non-essential functions to other systems or to an external provider that is better able to support the functions.	• Outsource non-essential services to a managed service provider. • Impose requirements on and perform oversight of external system services.

TECHNIQUES	APPROACHES	EXAMPLES
	Restriction Remove or disable unneeded functionality or connectivity, or add mechanisms to reduce the chance of vulnerability or failure.	• Configure the system to provide only essential capabilities. • Minimize non-security functionality.
	Replacement Replace low-assurance or poorly understood implementations with more trustworthy implementations.	• Remove or replace unsupported system components to reduce risk.
	Specialization Modify the design of, augment, or configure critical cyber resources uniquely for the mission or business function to improve trustworthiness.	• Re-implement or custom develop critical components. • Develop custom system elements covertly. • Define and apply customized configurations.
Redundancy Provide multiple protected instances of critical resources.	**Protected Backup and Restore** Back up information and software (including configuration data and virtualized resources) in a way that protects its confidentiality, integrity, and authenticity, and enable restoration in case of disruption or corruption.	• Retain previous baseline configurations. • Maintain and protect system-level backup information (e.g., operating system, application software, system configuration data).
	Surplus Capacity Maintain extra capacity for information storage, processing, or communications.	• Maintain spare parts (i.e., system components). • Address surplus capacity in service-level agreements with external systems.
	Replication Duplicate hardware, information, backups, or functionality in multiple locations and keep them synchronized.	• Provide alternate audit capability. • Shadow database. • Maintain one or more alternate storage sites. • Maintain one or more alternate processing sites. • Maintain a redundant secondary system. • Provide alternative security mechanisms. • Implement a redundant name and address resolution service.
Segmentation Define and separate system elements based on criticality and trustworthiness.	**Predefined Segmentation** Define enclaves, segments, or other types of resource sets based on criticality and trustworthiness so that they can be protected separately and, if necessary, isolated.	• Use virtualization to maintain separate processing domains based on user privileges. • Use cryptographic separation for maintenance. • Partition application from system functionality. • Isolate security functions from non-security functions. • Isolate security tools and capabilities using physical separation. • Isolate components based on mission or business function. • Separate subnets that connect to different security domains. In

TECHNIQUES	APPROACHES	EXAMPLES
		particular, provide a DMZ for Internet connectivity. • Employ system partitioning. • Employ process isolation. • Implement sandboxes and other confined environments. • Implement memory protection.
	Dynamic Segmentation and Isolation Change the configuration of enclaves or protected segments, or isolate resources while minimizing operational disruption.	• Implement dynamic isolation of components. • Implement software-defined networking and VPNs to define new enclaves. • Create a virtualized sandbox or detonation chamber for untrusted attachments or URLs.
Substantiated Integrity Ascertain whether critical system elements have been corrupted.	**Integrity Checks** Apply and validate checks of the integrity or quality of information, components, or services.	• Use tamper-evident seals and anti-tamper coatings. • Use automated tools for data quality checking. • Use blockchain technology. • Use non-modifiable executables. • Use polling techniques to identify potential damage. • Implement cryptographic hashes. • Employ information input validation. • Validate components as part of SCRM. • Employ integrity checking on external systems.
	Provenance Tracking Identify and track the provenance of data, software, or hardware elements.	• Employ component traceability as part of Supply Chain Risk Management (SCRM). • Employ provenance tracking as part of SCRM. • Implement anti-counterfeit protections. • Implement trusted path. • Implement code signing.
	Behavior Validation Validate the behavior of a system, service, or device against defined or emergent criteria (e.g., requirements, patterns of prior usage).	• Employ detonation chambers. • Implement function verification. • Verify boot process integrity. • Implement fault injection to observe potential anomalies in error handling.
Unpredictability Make changes randomly or unpredictably.	**Temporal Unpredictability** Change behavior or state at times that are determined randomly or by complex functions.	• Require re-authentication at random intervals. • Perform routine actions at different times of day.
	Contextual Unpredictability Change behavior or state in ways that are determined randomly or by complex functions.	• Rotate roles and responsibilities. • Implement random channel-hopping.