Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

A Cloud-Enabled Disaster Recovery Framework for Resilient Recovery of Prioritized Enterprise Applications in Distributed IT Environments

Mohammed Mazher Khalid[1]
[1]Lead Specialist, IT Products & Strategy Management, IT Strategy & GRC,
Saudi Arabian Mining Company (Maaden), Saudi Arabia

Mohammed Shoukatuddin[2], Mohammed Aqheel[3], Mohammed Afzal[4]
[2]Senior Specialist – OT Network & Cybersecurity, Maaden Aluminum Company, Saudi Arabia
[3]Senior IT Specialist, Maaden Aluminum Company, Saudi Arabia
[4]Specialist I – Systems Administration, Saudi Arabian Mining Company (Maaden), Saudi Arabia

Abstract **-** The migration of enterprise workloads onto distributed and cloud-mediated platforms has raised both the expectations and the engineering difficulty of disaster recovery (DR). This paper sets out a cloud-enabled DR framework that delivers resilient recovery for prioritized enterprise applications while meeting defined Recovery Time Objective (RTO) and Recovery Point Objective (RPO) targets, improving service continuity, and reducing operational risk in hybrid environments that combine on-premises infrastructure, public cloud regions, edge sites, and SaaS dependencies. Building on the established DR patterns — backup and restore, pilot-light, warm standby, and active-active — the framework binds application tiering derived from business impact analysis (BIA) to architectural choices for replication, orchestration, and observability. The paper treats RTO and RPO as measurable engineering constraints rather than declared intent, and argues that recovery readiness is sustained only through continuous testing and a control loop that closes the gap between designed and demonstrated behavior. Hybrid risk partitioning is examined as a way to preserve regulatory and operational technology (OT) controls without sacrificing the elasticity of public cloud. A reference architecture, a tier-to-pattern mapping, an RTO/RPO control loop, and a hybrid DR diagram are presented to support the discussion. The paper concludes that a cloud-enabled DR framework, when treated as an enterprise capability with disciplined validation, reduces downtime and data-loss exposure while improving agility and cost efficiency for distributed enterprises.

Keywords **-** disaster recovery; cloud computing; business continuity; recovery time objective; recovery point objective; enterprise architecture; hybrid cloud; operational risk; application tiering; OT/IT convergence.

I. INTRODUCTION

Many large enterprises now operate a deliberately distributed estate: on-premises data centres, regional edge sites, software-as-a-service platforms, and one or more public cloud regions. That distribution carries a quiet cost. Every additional integration point becomes a potential failure path on the day a region goes dark, a ransomware event encrypts a domain controller, or a misrouted change leaves database replication lagging by hours instead of seconds. Disaster recovery (DR), once treated as a periodic exercise tied to a single secondary site, has become a continuous engineering discipline.

The central question this paper addresses is practical: how can a cloud-enabled DR framework deliver resilient recovery for prioritized enterprise applications, meet defined Recovery Time Objective (RTO) and Recovery Point Objective (RPO) targets, improve service continuity, and reduce operational risk across heterogeneous IT environments? The motivation is concrete rather than theoretical. Plants stop producing when SCADA historians lose visibility into an ERP. Logistics platforms drift out of sync when an identity provider fails over but DNS does not converge. Finance close cycles slip when reporting databases recover at a point earlier than the ledgers they were meant to mirror. Each of these failures is recoverable on paper; few are recoverable in fifteen minutes without rehearsal.

Cloud platforms have made certain DR patterns dramatically more accessible. Object storage at single-digit cents per gigabyte, cross-region replication of managed databases, programmable network fabrics, and infrastructure-as-code (IaC) for repeatable rebuilds together reduce the marginal cost of warm standby for workloads that once would have required a duplicate data centre [1], [7]. Yet the same elasticity exposes new design responsibilities. Identity continuity, key-management portability, network reachability after failover, and the integrity of the recovery copy itself all depend on choices made long before an incident.

The remainder of the paper is organized as follows. Section II develops the framework — application tiering, pattern selection, and a reference architecture. Section III treats RTO and RPO as measurable engineering constraints and discusses the automation and observability needed to sustain them. Section IV addresses operational-risk reduction through hybrid models, security continuity during failover, and continuous testing. Section V concludes.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

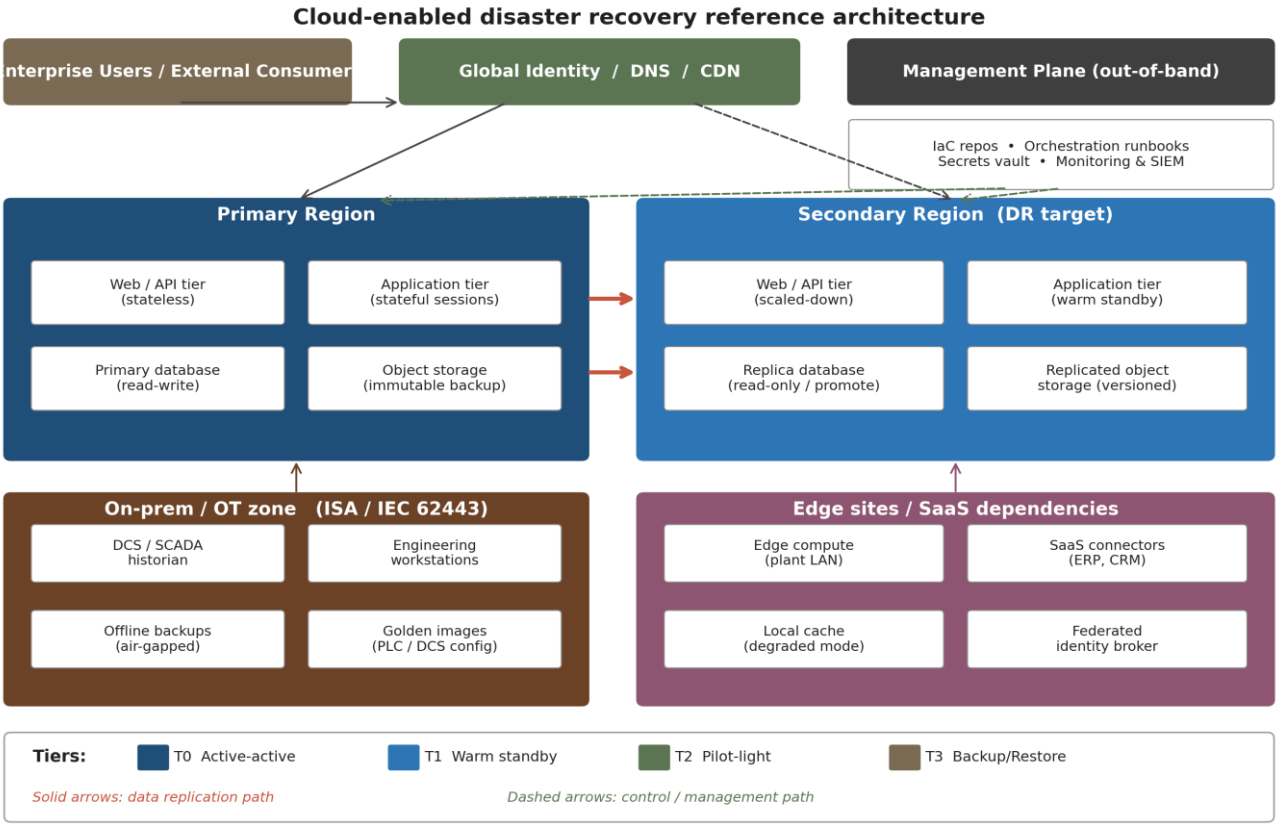

Fig. 1. Cloud-enabled DR reference architecture for prioritized enterprise applications across on-premises, public cloud, edge, and SaaS planes.

II. FRAMEWORK FOR RESILIENT RECOVERY

A. Designing a Cloud-Based DR Strategy

A resilient DR strategy starts with a business impact analysis (BIA) that establishes which services must come back first, how much data loss the business can absorb for each one, and what dependencies are buried inside the application stack. The BIA's deliverable is not a document; it is a tier assignment per workload that downstream design decisions are obligated to honor [9], [12].

Once tiers are assigned, four reference patterns dominate enterprise practice. Backup and restore remains the only economical choice for cold archives and low-tier reporting systems. Pilot-light keeps a minimal footprint warm — typically databases replicating asynchronously and a small control plane — while compute is scaled out at failover, trading minutes of RTO for substantial run-rate savings. Warm standby maintains a continuously running but downsized environment that can absorb traffic immediately after a route change. Active-active configurations distribute traffic across two or more regions in steady state and are reserved for workloads where outage windows are measured in seconds [1], [2].

Recent managed services blur the boundaries between these patterns. AWS Elastic Disaster Recovery replicates entire server volumes continuously and provisions compute only at recovery, behaving operationally like pilot-light while delivering RTO closer to warm standby. Azure Site Recovery offers comparable block-level replication with orchestrated runbooks. VMware Cloud DR and Zerto, deployed on Azure or AWS, extend journal-based recovery to virtualized estates already standardized on those hypervisors [5], [7]. These services have not eliminated the need for architectural thought, but they have shortened the operational distance between tiers, so that moving a workload from pilot-light to warm standby is now a configuration change rather than a project.

From an enterprise-architecture standpoint, DR sits across application, data, technology, and security domains rather than being owned by infrastructure alone. The architecture must therefore answer questions that span boundaries: which data sources are authoritative under a split-brain condition, how network paths are re-advertised when a region withdraws BGP announcements, whether privileged access works when the bastion that normally mediates it is itself offline, and how security monitoring continues when the primary SIEM ingest path is disrupted. These cross-cutting properties are easy to design and easy to forget, and they account for the majority of "the failover worked but we still had an outage" post-mortems.

B. Application Tiering and Pattern Selection

Table I shows the tiering that most enterprises converge on after a serious BIA, paired with the DR pattern that satisfies the typical RTO/RPO bounds for each tier without overbuilding for the remainder of the estate. Figure 2 illustrates the same mapping in visual form and adds representative workloads.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

TABLE I. APPLICATION TIERING ALIGNED TO CONTINUITY REQUIREMENTS AND DR PATTERNS

Tier	Continuity requirement	Typical DR pattern	Representative workloads
Tier 0	RTO < 1 min, RPO ≈ 0	Active-active / hot standby	Identity, DNS, payment authorization, control planes
Tier 1	RTO < 15 min, RPO < 60 s	Warm standby	Customer-facing apps, ERP front-end, plant scheduling
Tier 2	RTO < 4 h, RPO < 15 min	Pilot-light / cold standby	Internal services, analytics, document management
Tier 3	RTO < 24 h, RPO < 24 h	Backup & restore	Batch / reporting, archives, dev-test environments

Two observations from practice are worth emphasizing. First, tier 0 should be reserved for services on which several other tiers depend — identity providers, DNS, certificate authorities, secret stores, network firewalls — rather than for the customer-facing applications that BIA workshops gravitate towards. A customer portal that fails over correctly into a region without working identity recovers in name only. Second, the cost gradient between tiers is steep: moving a workload from tier 2 to tier 1 can multiply its infrastructure cost by a factor of two or three, which is one reason why mature programs aggressively defend tier boundaries against well-intentioned reclassification.

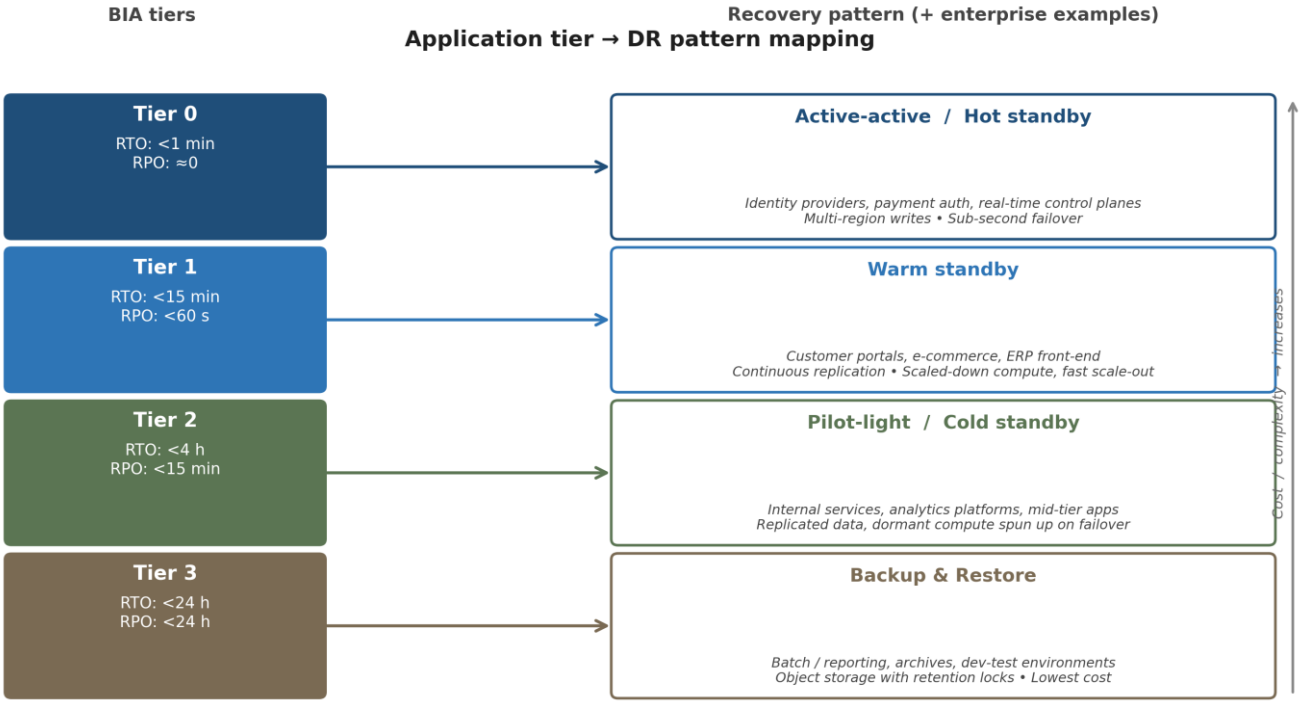

Fig. 2. Mapping of application tiers to commonly adopted DR patterns, with example workloads and the cost/complexity trend.

C. Reference Architecture

Figure 1 sketches the reference architecture used in the remainder of the paper. The primary site hosts production workloads across application, data, and shared-services planes. Continuous replication paths carry block-level changes to a secondary region, while object storage holds immutable backup copies protected by retention locks and, increasingly, object-level write-once-read-many (WORM) policies to defend against ransomware tampering [10]. A separate management plane — IaC repositories, orchestration runbooks, secrets and key vaults, and the monitoring fabric — sits outside both regions so that the recovery operation does not depend on the site it is recovering. Identity and DNS converge through globally available services to avoid creating a single regional dependency under the recovery path.

In industrial deployments, an OT zone is included on the same diagram but governed by separate controls. Plant historians, engineering workstations, and configuration backups for distributed control systems (DCS) and programmable logic controllers (PLC) sit behind ISA/IEC 62443 zone-and-conduit segmentation [13]. Their recovery strategy emphasizes redundant on-premises capacity, golden-image restoration, and offline backups rather than public-cloud replication. The framework must hold both planes coherently because the business processes — production scheduling, predictive maintenance, regulatory reporting — span the boundary.

III. MEETING RTO AND RPO TARGETS

A. Translating BIA into Engineering Constraints

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

RTO and RPO are useful only when they are observable. An RTO of fifteen minutes that nobody measures is indistinguishable from no RTO at all. The first engineering step, therefore, is to make each target a metric with a definition, an owner, and an alarm threshold.

RPO is bounded by the replication mechanism. Asynchronous replication produces an RPO equal to the worst-case lag plus the failure-detection window, while synchronous replication forces RPO toward zero at the cost of write latency [2], [10]. Synchronous replication is feasible only within latency budgets that practical inter-region distance often violates, which is why dual-region synchronous designs typically apply only to control-plane data within a metropolitan area, with longer-distance asynchronous tiers for the bulk of state.

RTO is bounded by the slowest step in the failover sequence. In the absence of automation, that step is usually human: the time taken to convene a recovery team, locate the current version of a runbook, find credentials that have not expired, and execute commands in the correct order. Orchestration collapses those minutes into the time the underlying systems actually need to converge.

B. Automation and Orchestration

Infrastructure-as-code (Terraform, AWS CloudFormation, Azure Bicep, Pulumi) eliminates environment drift between primary and secondary by ensuring both sites are built from the same definitions. When the secondary region is rebuilt from code at failover, the team is restoring data into a known-good substrate rather than into whatever the secondary happened to drift into since the last manual change.

Orchestration layers — AWS Systems Manager, Azure Automation, Ansible, Rundeck — sequence the recovery itself. A typical sequence begins by promoting replicated databases, then warms application servers, repoints DNS or load-balancer targets, runs application-level smoke tests, and only then opens external traffic. Each step has explicit dependencies and explicit failure handling. Manual runbooks executed under incident pressure rarely achieve this discipline [4], [6].

Workload mobility tools — Velero for Kubernetes, Carbonite Migrate, Veeam Recovery Orchestrator, AWS Elastic Disaster Recovery — provide a recovery path that does not depend on the primary platform being functional. This matters especially for ransomware scenarios, where the primary may be intact but no longer trusted; recovery into a clean cloud environment is sometimes the fastest verified path to service restoration [10].

C. Observability and Telemetry

Table II lists the observability metrics that a credible DR program actually measures, as distinct from the ones it claims to.

TABLE II. DR OBSERVABILITY METRICS AND THEIR OPERATIONAL PURPOSE

Metric / evidence	What it confirms	Operational use
Replication lag	RPO exposure stays within target bounds	Alerting and escalation when lag breaches thresholds
Backup success rate	Backups complete as designed	Trend monitoring; detect silent failures early
Restore verification	Data is recoverable and application-consistent	Periodic drills; audit evidence
Failover runbook duration	Measured RTO is achievable in practice	Continuous improvement; bottleneck removal
Access-control readiness	Identity and privileged access work during DR	Pre-incident validation; reduce recovery friction

Replication lag is the most basic metric and the one most often missed. A backup job that has been failing silently for nine days is a recurring finding in DR audits, and it is invariably surprising to the team that owns the source system [3]. Centralized monitoring with synthetic checks against replication endpoints — not just job status but the freshness of the data on the secondary — closes this gap.

Restore verification deserves separate emphasis. A backup that cannot be restored is not a backup; it is a storage cost. Periodic restoration into an isolated environment, followed by application-consistency checks — database integrity verification, message-queue replay, application self-tests — converts assumed recoverability into measured recoverability.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

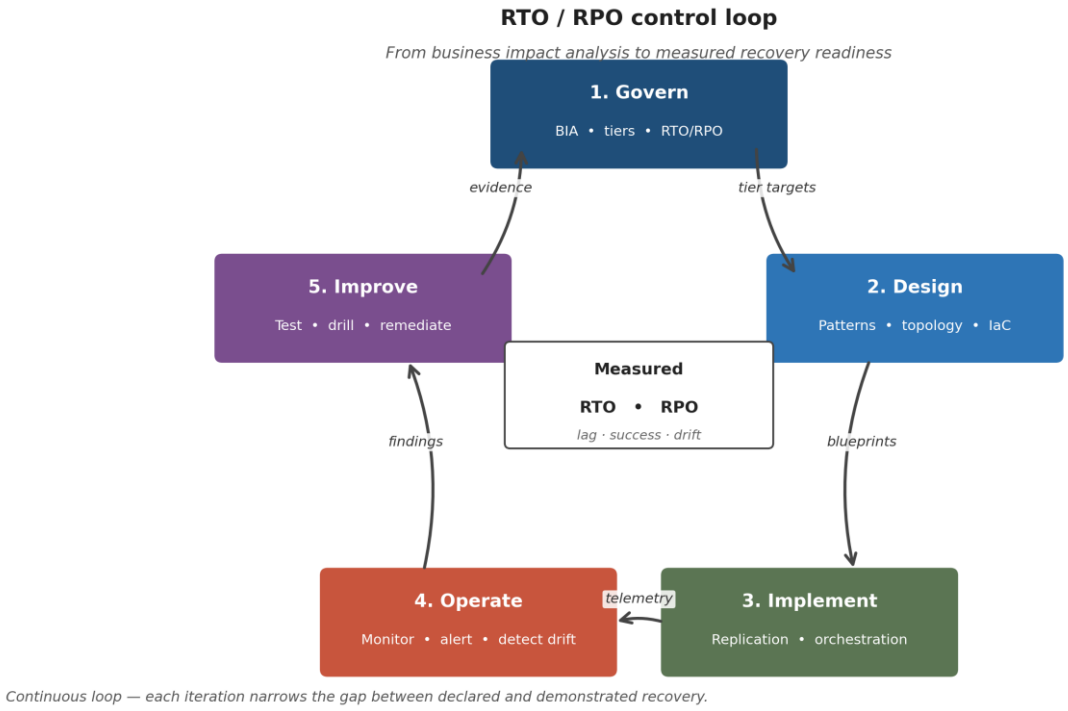

Fig. 3. RTO/RPO control loop. Governance, design, implementation, operation, and improvement feed measured outcomes that, in turn, drive the next iteration of the framework.

IV. ENHANCING CONTINUITY AND REDUCING OPERATIONAL RISK

A. Hybrid Cloud Risk Partitioning

Most large enterprises operate hybrid environments, not because hybrid is fashionable but because regulated data, latency-sensitive control systems, and amortized on-premises investment all resist consolidation. A practical DR framework accepts this constraint and partitions risk accordingly: regulated workloads remain in controlled facilities with dedicated recovery capacity, elastic cloud regions absorb tier-2 and tier-3 recovery and provide geographic separation for tier-0 control planes [10], [14].

In industrial environments — refineries, petrochemical plants, water treatment, power generation — the partition is sharper still. OT systems sit behind ISA/IEC 62443 zone and conduit segmentation and rarely cross into public cloud at all [13]. Their DR strategy emphasizes redundant on-premises servers, golden-image restoration of engineering workstations, and offline backups of PLC and DCS configurations. The IT side of the same enterprise, in contrast, makes aggressive use of cloud DR services. The framework must hold both halves coherently because the business processes that depend on them — production scheduling, predictive maintenance, regulatory reporting — span the boundary.

B. Security Continuity During Failover

Failover is a privileged operation. It modifies routing, promotes secondary copies to authoritative, and frequently issues new credentials. Each of these is a target. The framework should therefore treat the secondary site as production from a security standpoint from the moment it is provisioned: the same patch levels, the same logging, the same access controls, and the same monitoring coverage. A "DR site we will harden later" is a foothold waiting for an adversary [3].

Key management deserves specific attention. If data at rest in the secondary region is encrypted with keys held only in the primary, the recovery cannot proceed when the primary is unreachable. Cloud key-management services — AWS KMS multi-region keys, Azure Key Vault with geo-replication — address this when configured deliberately; they fail it silently when used with default single-region settings.

Federated identity is the other recurring weak point. Recovery teams should never depend on credentials that the failed component itself issues. Break-glass accounts, stored in hardware tokens or in offline-recoverable form, are an unglamorous but essential control [11].

C. Continuous Testing and Validation

DR plans decay. Applications change, dependencies shift, credentials expire, IaC drifts from running state, and the runbook that worked at the last quarterly drill no longer matches reality. The only defense is continuous testing.

A mature testing program combines several modes. Tabletop exercises validate decision-making and communications. Component failover tests exercise individual services without disrupting production. Full regional failover exercises — performed on a schedule visible to the business — measure end-to-end RTO under realistic conditions and frequently expose dependencies that nobody documented. Chaos-engineering

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

tools (Gremlin, AWS Fault Injection Service, Azure Chaos Studio) inject controlled failures into production to verify that resilience controls behave the way the design assumed [4], [8].

Table III maps the operational-risk areas a DR program must address to the controls that mitigate them and the validation activity that proves those controls work.

TABLE III. RISK AREAS, CONTROLS, AND VALIDATION ACTIVITIES FOR A CLOUD-ENABLED DR PROGRAM

Risk area	Typical control	Validation activity
Architecture drift	IaC baselines and configuration management	Rebuild tests and drift detection reports
Data loss / inconsistency	Tiered replication and immutable backup policies	Restore tests; checksum and consistency checks
Access failure during DR	Federated identity and privileged-access continuity	Credential expiry checks; DR access drills
Network reachability	Segmented connectivity and tested failover routing	Failover routing tests; firewall-rule audits
Operational readiness	Runbooks and automation with role-based ownership	Game-days; post-exercise remediation tracking

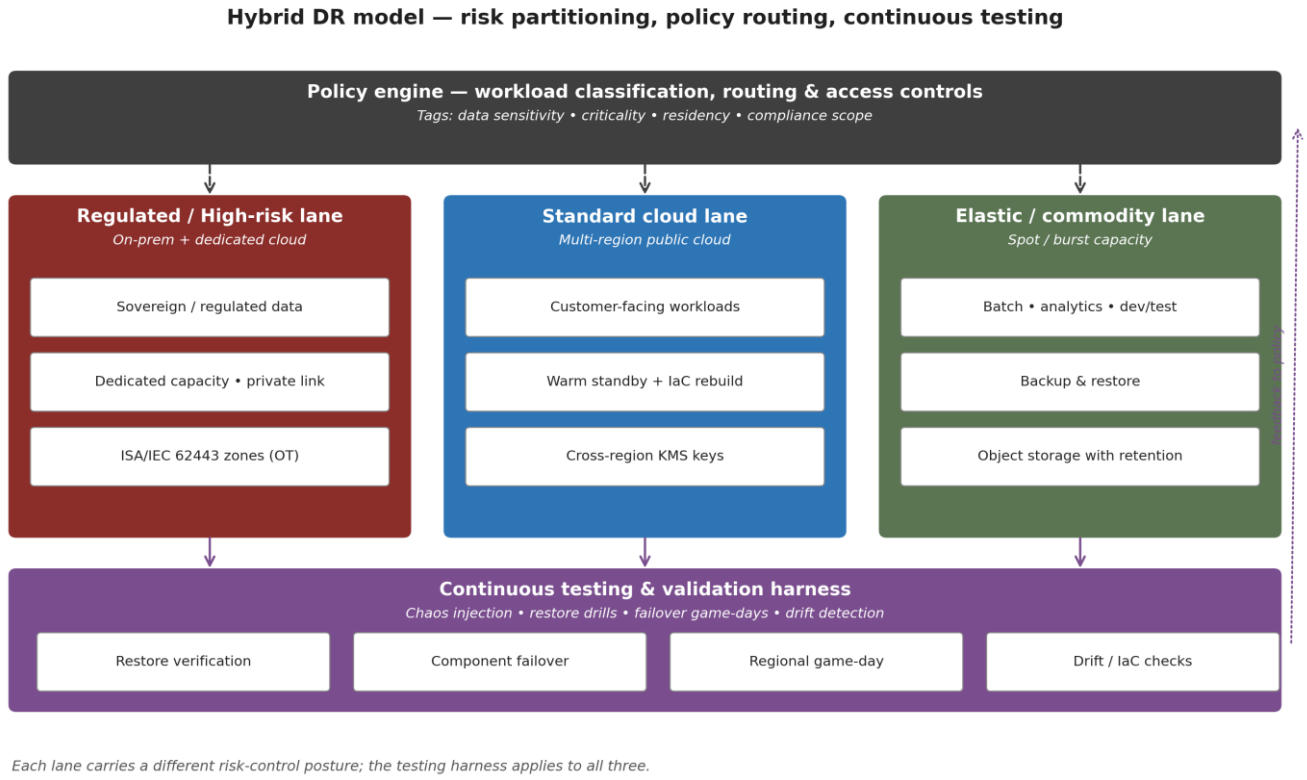

Fig. 4. Hybrid DR model showing risk partitioning across regulated, standard-cloud, and elastic lanes, governed by a policy engine and validated by a continuous testing harness.

V. CONCLUSION

A cloud-enabled DR framework delivers resilient recovery for prioritized enterprise applications when it is treated as an enterprise capability rather than as an infrastructure feature. The governing principle is alignment: BIA-derived tiers determine DR patterns, patterns dictate replication mode and failover sequencing, sequencing demands automation and observability, and observability feeds the continuous testing that keeps the program honest. Each layer is necessary; none is sufficient alone.

Cloud platforms make this alignment cheaper and faster to implement than at any prior point in enterprise IT. They do not, however, remove the work. Identity must remain trustworthy during failover, keys must be portable across regions, security controls must follow the workload, and the framework must hold coherently across the OT/IT boundary that defines so many industrial enterprises.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

Recovery readiness is a property of organizations that test relentlessly, not of organizations that document thoroughly. The framework presented in this paper gives that testing something concrete to verify: that tier-0 services come back in single-digit minutes, that tier-1 RPO stays within seconds, that hybrid risk partitioning holds under realistic incident conditions, and that the recovery path the runbook describes is the one the systems actually take. Enterprises that hold themselves to those measurements turn DR from a periodic exercise into a continuously demonstrated capability, and reduce both downtime exposure and operational risk in the process.

REFERENCES

[1] A. Abualkishik, A. Alwan, and Y. Gulzar, "Disaster recovery in cloud computing systems: An overview," Int. J. Adv. Comput. Sci. Appl., vol. 11, no. 9, pp. 686–693, 2020.

[2] M. M. Alshammari, A. Alwan, A. Nordin, and A. Abualkishik, "Data backup and recovery with a minimum replica plan in a multi-cloud environment," Int. J. Grid High Perform. Comput., vol. 12, no. 2, pp. 102–120, 2020.

[3] M. Anisetti, C. Ardagna, E. Damiani, and F. Gaudenzi, "A semi-automatic and trustworthy scheme for continuous cloud service certification," IEEE Trans. Services Comput., vol. 13, no. 1, pp. 30–43, Jan.–Feb. 2020.

[4] W. Cerroni, L. Foschini, G. Grabarnik, F. Poltronieri, L. Shwartz, C. Stefanelli, and M. Tortonesi, "BDMaaS+: Business-driven and simulation-based optimization of IT services in the hybrid cloud," IEEE Trans. Netw. Service Manage., vol. 19, no. 1, pp. 322–337, Mar. 2022.

[5] G. Gurulakshmanan and R. N. Amarnath, "Efficient and robust disaster recovery system using cloud-based algorithms with data integrity," Indones. J. Electr. Eng. Comput. Sci., vol. 35, no. 1, pp. 388–396, 2024.

[6] O. O. Ibrahim, "Impact of cloud computing on business continuity and disaster recovery," J. Technol. Syst., vol. 6, no. 4, pp. 1–15, 2024.

[7] I. Kumar, "Cloud-computing-based disaster recovery," Turkish J. Comput. Math. Educ., vol. 11, no. 1, 2020.

[8] F. Meng, "Enterprise data backup and validation strategy selection based on cloud computing," in Proc. Int. Symp. Soc. Sci. (ISSS 2015), Atlantis Press, 2015.

[9] F. Márquez and M. Papaelias, "Introductory chapter: Introduction to dependability engineering," in Dependability Engineering. IntechOpen, 2018.

[10] Y. Q. Zhao and S. Zhu, "Research on data risk control strategies for hybrid cloud," J. Inf. Anal., 2024.

[11] National Institute of Standards and Technology, "Contingency planning guide for federal information systems," NIST SP 800-34 Rev. 1, Washington, DC, USA, May 2010.

[12] International Organization for Standardization, Societal Security — Business Continuity Management Systems — Requirements, ISO 22301:2019, 2019.

[13] International Society of Automation, Security for Industrial Automation and Control Systems — Part 3-3: System Security Requirements and Security Levels, ISA/IEC 62443-3-3, 2013.w

[14] U.S. Congress, Sarbanes–Oxley Act of 2002, Public Law 107-204, 116 Stat. 745, 2002.