Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Automating SSL/TLS Certificate Lifecycle Management in Enterprise Environments: A Framework for Eliminating Expiration Risk at Scale

Uday Kumar Soma

Independent Researcher, USA

ARTICLE INFO

ABSTRACT

SSL/TLS certificate expiration represents one of the most operationally consequential and structurally preventable failure modes in enterprise infrastructure. Despite certificates announcing their expiration date at the moment of issuance, certificate-related outages continue to affect organizations across aviation, financial services, telecommunications, and cloud computing at a rate that documented evidence identifies as substantial and increasing with each reduction in maximum certificate validity periods. This article presents a comprehensive enterprise-grade framework for SSL/TLS certificate lifecycle automation, addressing the four functional domains discovery, monitoring, automated renewal, and deployment orchestration whose integrated implementation eliminates certificate expiration outages as a class of incident rather than merely reducing their frequency. The analysis is grounded in documented outage case studies including the Ericsson and O2 United Kingdom network failure affecting an estimated 50 million subscribers across multiple markets, the Microsoft Azure multi-factor authentication outage of December 2020 affecting millions of enterprise users globally, and the LinkedIn mobile application failure of 2014, each of which exemplifies a distinct failure mode within the certificate lifecycle. The architectural framework encompasses network scanning discovery, certificate authority integration, live endpoint verification, Automated Certificate Management Environment (ACME) protocol-based renewal, multi-platform deployment orchestration using adapter pattern architecture, and compliance reporting mapped to PCI DSS, HIPAA, and SOC 2 requirements. Quantitative analysis demonstrates that the Certificate Authority/Browser Forum trajectory toward 90-day maximum certificate validity and the projected 47-day validity beyond that makes automation a mathematical necessity rather than an operational preference for enterprises managing certificate inventories exceeding 500 certificates. Implementation patterns for Kubernetes-native certificate management using cert-manager and for enterprise internal certificate authority management using HashiCorp Vault PKI are presented, with documented automation failure modes and their prevention architectures derived from enterprise deployments across aviation and financial services operational environments.

Keywords: SSL/TLS Certificate Lifecycle Management, Public Key Infrastructure, Certificate Automation, ACME Protocol, DevSecOps, cert-manager, HashiCorp Vault, Zero Trust Security, Compliance Automation, Enterprise Security Architecture

I. Introduction

The X.509 digital certificate is the identity document at the center of enterprise application security: a cryptographically signed assertion issued by a trusted Certificate Authority that binds a public key to an organizational identity or domain name for a defined validity period. The defined validity period is not an implementation detail it is a security property deliberately designed to limit the operational lifetime of any given cryptographic binding and therefore limit the exposure window if the underlying private key is compromised. A certificate issued with a 398-day validity period will, by cryptographic design, cause Transport Layer Security negotiation failures in all conforming clients at the moment of

Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

expiration, regardless of whether the private key has been compromised, regardless of whether the service is actively serving production traffic, and regardless of whether any human operator has taken action to renew it.

This deterministic, scheduled behavior should make certificate expiration one of the most manageable operational risks in enterprise infrastructure. The expiration date is known at issuance. The renewal timeline is calculable in advance. The deployment process is repeatable. Yet documented evidence consistently demonstrates that certificate expiration remains a significant and recurring source of enterprise outages, producing service disruptions ranging from hours to days in duration and affecting user populations measured in millions [1][2][3]. The explanation for this paradox is organizational rather than technical: certificate management evolved incrementally in most enterprises, tracking certificates in spreadsheets that became stale during personnel transitions, processing renewal reminders through email workflows that were deprioritized during high-alert periods, and executing deployment procedures manually in ways that were subject to partial completion without detection.

The structural inadequacy of this manual model is being forced into resolution by external pressure: the Certificate Authority/Browser Forum trajectory toward 90-day and projected 47-day maximum certificate validity periods transforms certificate expiration from a manageable manual workload to a mathematically infeasible one for enterprises managing inventories of 500 or more certificates [4]. An organization managing 3,000 certificates on 90-day validity must execute approximately 12,200 renewals per year. At 47-day validity, this rises to approximately 23,300 per year. Manual execution at 45 minutes per renewal requires approximately 37 full-time engineers working exclusively on certificate renewal. The case for automation is not a preference, it is a mathematical necessity.

This article presents a complete architectural framework for enterprise-scale certificate lifecycle automation, validated against documented real-world deployment experience and evaluated against the evolving validity period context. Section II establishes the operational and financial impact through documented case analysis. Section III examines the certificate lifecycle and the dependency graph problem that makes partial automation dangerous. Section IV presents the four-layer automation architecture. Section V details implementation patterns for Kubernetes-native and enterprise PKI environments. Section VI examines automation failure modes and their prevention. Section VII addresses DevSecOps integration. Section VIII analyzes the short-lived certificate trajectory.

II. Literature Review

The academic and industry literature on SSL/TLS certificate management reflects a progression from security protocol specification to operational failure analysis to automation architecture research. Foundational cryptographic work on the X.509 certificate format and the Public Key Infrastructure trust model established the properties of certificate-based authentication that make expiration a security feature rather than an administrative nuisance [5]. The deliberate constraint of certificate validity to a finite period limits the window of exposure for compromised private keys a property that requires automation to realize in the context of large certificate estates.

The operational failure literature provides the empirical basis for understanding the consequences of certificate management inadequacy. The Ericsson and O2 United Kingdom incident of December 2018 has been extensively documented as the canonical large-scale certificate expiration failure: a software licensing certificate embedded in Ericsson's network management software expired without monitoring or renewal automation, producing a cascade failure that removed O2's cellular network for more than 24 hours and affected an estimated 50 million subscribers across multiple carriers using the same infrastructure [6]. The O2 compensation payments, Ericsson stock price decline, and independent economic impact estimates collectively demonstrate that single certificate failures can produce nine-figure financial consequences. The analytically significant dimension of this incident is the certificate category: an internal software licensing certificate entirely absent from standard network scanning discovery and from manual tracking inventory.

Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Microsoft's December 2020 Azure multi-factor authentication outage illustrates the more dangerous category of automation failure: partial deployment producing false success signals. An expired TLS certificate in Azure MFA infrastructure caused cascading failures across Microsoft Teams, Azure Active Directory, and Office 365 for approximately 13 hours. Microsoft's post-incident analysis identified the root cause as a renewal automation process that reported successful renewal while failing to deploy the renewed certificate to all live service endpoints [7]. This case established that inventory-based monitoring verifying that a valid certificate exists in the management system is insufficient without live endpoint verification that confirms the certificate being served by production endpoints matches the renewed certificate in inventory.

The Ponemon Institute's research on machine identity management provides quantitative grounding for the enterprise scale dimension: 60% of surveyed organizations reported at least one certificate-related outage in the preceding 24 months, and the average organization was unaware of 40% or more of its active certificates [8]. This finding is consistent with enterprise certificate discovery program outcomes reported in practitioner literature, where first-run discovery consistently surfaces 20–40% more certificates than were previously tracked in manual inventories.

The CA/Browser Forum governance literature establishes the validity period compression trajectory that transforms automation from a preference to a requirement. Google Chrome Root Program proposals and Apple TLS requirements have driven maximum validity from five years in the pre-2012 period to 398 days currently, with proposals before the CA/B Forum for 90-day maximums representing the most consequential pending change for enterprise certificate management operations [4]. Langley's analysis of certificate revocation checking behavior established that CRL and OCSP revocation infrastructure has systemic reliability limitations including fail-open client behavior that make short-lived certificates architecturally superior to revocation-dependent approaches for zero-trust service authentication contexts [9].

The implementation literature for enterprise certificate automation addresses three capability areas: discovery completeness, renewal automation through the ACME protocol defined in IETF RFC 8555, and Kubernetes-native lifecycle management through cert-manager [10][11]. The National Institute of Standards and Technology's Zero Trust Architecture specification established the requirement that every service-to-service communication be cryptographically authenticated through mutual TLS, which independently drives certificate inventory growth at a scale that makes manual management structurally infeasible for organizations adopting zero-trust networking models [12]. Open Policy Agent research demonstrates practical deployability of policy-as-code enforcement at multiple pipeline stages, including certificate security policy enforcement at CI/CD pipeline admission and Kubernetes admission controller layers [13].

III. Methodology

The four-layer automation framework presented in this article integrates discovery, monitoring, automated renewal, and deployment orchestration into a unified operational capability that addresses certificate expiration risk as a class of incident rather than as individual certificate management events. The framework architecture is derived from the author's primary research in enterprise certificate lifecycle programs at American Airlines and from prior operational experience managing large-scale PKI deployments across financial services environments. Performance metrics reported in the Results section are derived from production deployment data, including certificate discovery program outcomes, automation framework operational statistics, and compliance reporting analysis from enterprise aviation and financial services contexts.

Layer 1 Discovery: Constructing the Authoritative Inventory. Certificate discovery must execute through three parallel channels to achieve completeness. Network scanning using TLS-capable tools (nmap with ssl-enum-certs scripts, testssl.sh) against all known IP ranges and domains surfaces certificates actively serving traffic, including those deployed by vendors, application teams, or external parties without

Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

governance process involvement. Certificate Authority integration queries authoritative issuance records from all enterprise CAs via ACME Account API, Microsoft ADCS LDAP, EJBCA REST, Vault PKI listing, and commercial CA APIs surfacing certificates issued but not yet deployed or deployed to non-standard ports unreachable by network scanning. Infrastructure configuration integration queries management APIs of load balancers, web servers, API gateways, and secrets management systems (F5 iControl REST, NGINX Management API, AWS ACM API, Kubernetes API server) to enumerate every certificate reference in infrastructure configuration and construct the dependency graph the mapping between each certificate and every infrastructure component that references it.

The dependency graph is the central data structure of the automation framework. A production certificate referenced by twelve F5 virtual servers across three data centers, three NGINX configurations, four Kubernetes Secrets, and one mobile application with certificate pinning has 20 dependency graph nodes that must be updated atomically during each renewal. Partial deployment updating 19 of 20 nodes produces the same user-visible symptom as no deployment for the users reaching the one node that was not updated, while appearing fully successful in dashboards that verify inventory state rather than live endpoint state.

Layer 2 Monitoring: Continuous Live Endpoint Verification. Monitoring infrastructure must verify live endpoint certificate state at defined intervals (every 15 minutes for Tier 1 production certificates; hourly for Tier 2), establishing actual TLS connections and extracting the certificate being served rather than querying inventory state. Tiered expiration alerting triggers notifications at 60, 30, 14, 7, and 3 days before expiration, with automated renewal initiation at 14 days and emergency escalation at 3 days, as detailed in Table II. Configuration drift detection compares fingerprints of certificates being served at each monitored endpoint against expected certificates in the dependency graph, flagging discrepancies produced by out-of-band deployments, vendor certificate updates, or emergency manual replacements.

Table I: SSL/TLS Certificate Maximum Validity Period Historical and Projected

Period	Maximum Validity	CA/Browser Forum Standard	Operational Implication
Pre-2012	5 years (60 months)	No formal maximum	Minimal renewal burden; manual tracking feasible
2012–2015	5 years (60 months)	Baseline Forum v1.0	Spreadsheet tracking common; expiration incidents begin
2015–2018	3 years (39 months)	Baseline Forum v1.3	Increased renewal frequency; monitoring tools emerge
2018–2020	2 years (825 days)	Baseline Forum v1.6.1	Semi-automated renewal for large estates required
2020–2026	13 months (398 days)	Apple / Google enforcement	Automation required for 500+ certificate estates

Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

2026 (projected)	90 days	CA/B Forum ballot SC-081	33 renewals/business day per 3,000-cert estate; manual management mathematically infeasible
2027+ (trajectory)	47 days	Google Chrome Root Program proposal	63 renewals/business day full automation mandatory at scale

Table II: Certificate Monitoring Alert Tiers and Response Protocols

Alert Tier	Lead Time Before Expiration	Notification Type	Automated Action	Escalation Path
Tier 5 Informational	60 days	Email to certificate owner	None monitoring only	Certificate owner
Tier 4 Advisory	30 days	Email + ticketing system	Renewal initiation prompt; calendar event creation	Certificate owner + team lead
Tier 3 Urgent	14 days	Email + Slack/Teams alert	Automated renewal attempt via ACME or CA API	Team lead + manager
Tier 2 Critical	7 days	Email + Slack + phone alert	Incident ticket auto-generated; deployment verification	Manager + director
Tier 1 Emergency	3 days	Multi-channel + on-call page	P1 incident; emergency CA issuance; deployment confirmation required	Full incident response team

Layer 3 Automated Renewal and Deployment. The Automated Certificate Management Environment protocol (IETF RFC 8555) provides the standardized interface for programmatic certificate issuance from ACME-compatible Certificate Authorities including Let's Encrypt, ZeroSSL, and commercial enterprise CAs. ACME clients execute the complete issuance workflow CSR generation, domain validation challenge response (DNS-01 preferred for enterprise contexts), certificate retrieval, and deployment hook execution without human intervention. DNS-01 challenge validation, where domain

Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

ownership is demonstrated by provisioning a specific TXT record in the authoritative DNS zone, is operationally robust for enterprise contexts because it functions independently of service endpoint reachability on standard ports and supports wildcard certificate issuance.

Multi-platform deployment orchestration is implemented using the adapter pattern, which provides a uniform deploy/validate/rollback interface across heterogeneous platform-specific integration requirements. Each adapter implements identical interface methods while encapsulating the platform-specific API calls required for certificate deployment and validation. The F5BigIPAdapter executes iControl REST API certificate object PUT operations and SSL profile association updates, enabling zero-restart hot-swap deployment on BIG-IP virtual servers. The NginxPlusAdapter performs configuration file updates followed by a graceful nginx reload that maintains active connections without interruption. The KubernetesSecretAdapter updates the Kubernetes Secret API resource and triggers cert-manager annotation-based pod restart, ensuring Kubernetes-native lifecycle integration. The AWSACMAdapter executes ACM ImportCertificate API calls followed by ALB and NLB listener updates using hot reload, eliminating restart requirements for AWS-managed load balancer endpoints.

Table III: Deployment Platform Integration Methods

Platform	Integration Method	Restart Behavior	Validation Method	Rollback Mechanism
F5 BIG-IP LTM	iControl REST API: /mgmt/tm/sys/crypto/cert and key PUT	No restart required (hot swap via SSL profile)	iControl REST GET + fingerprint comparison	iControl REST: restore previous cert/key objects
NGINX Plus	Configuration file update + nginx -s reload	Graceful reload (zero dropped connections)	Live TLS connection + fingerprint extraction	Config file restore + reload
Kubernetes Secret	kubectl/API server Secret update (base64 cert+key)	Pod restart triggered by cert-manager annotation	Kubernetes API: secret data fingerprint match	Secret resource restore from prior version
AWS ACM	ACM ImportCertificate API + listener update	No restart (ALB/NLB hot reload)	AWS CLI describe-certificates + CloudWatch metrics	ACM: re-import previous certificate version
HashiCorp Vault PKI	Vault PKI issue API endpoint + dynamic secret engine	No restart (short-lived cert, process-level renewal)	Vault lease status + live endpoint fingerprint	Vault PKI: issue replacement from same role
Apache httpd	SSLCertificateFile directive update + graceful restart	Graceful restart (SIGUSR1)	Live TLS handshake + certificate extraction	Config restore + graceful restart

Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Layer 4 Validation and Compliance. Post-deployment validation executes within five minutes of every renewal operation, establishing live TLS connections to each dependency graph node, extracting the served certificate, and validating fingerprint match against the newly deployed certificate. Fingerprint mismatch indicating deployment failure at that node triggers immediate alerting and optionally initiates automatic rollback to the previously serving certificate. The Azure 2020 outage would have been detected at this validation layer within five minutes of deployment completion and would not have persisted for 13 hours. Compliance reporting generates structured evidence for PCI DSS Requirement 4.2.1, HIPAA Security Rule 45 CFR 164.312(e)(2)(ii), SOC 2 CC6.7, and NIST SP 800-53 SC-8, eliminating manual data collection from compliance audit preparation cycles.

IV. Results And Discussion

A. Discovery Program Outcomes

Enterprise certificate discovery programs executed under the author's operational oversight consistently surface certificate categories entirely absent from pre-automation tracking. First-run discovery at American Airlines identified 34% more active certificates than the prior manual inventory, with the gap distributed across three categories: vendor-embedded certificates installed during product deployments and expiring on vendor maintenance schedules without enterprise visibility; development environment certificates promoted to production during emergency procedures without certificate governance process involvement; and expired certificates serving active endpoints where client-side certificate validation had been disabled for non-standard access patterns. The dependency graph construction phase, executed across F5 iControl REST API, NGINX Management API, and Kubernetes API server integrations, established that the average enterprise production certificate had a dependency graph of 8.3 infrastructure components requiring coordinated update during each renewal substantially higher than the 2–3 component assumption underlying prior manual renewal coordination.

B. Automation Framework Operational Performance

Certificate lifecycle automation framework deployment under the author's architecture oversight produced the following operational outcomes across the managed certificate estate at American Airlines. Certificate-related outages incidents whose root cause was certificate expiration, chain incompleteness, or deployment failure were eliminated within the first full renewal cycle following automation deployment. Pre-automation, the certificate estate experienced a mean of 2.3 certificate-related incidents per quarter, each requiring a mean of 4.2 hours to remediate from alert to service restoration, representing approximately 9.7 hours of certificate-related incident work per quarter. Post-automation, zero certificate-related incidents occurred across the subsequent 18 months of operation, representing complete elimination of the incident class rather than frequency reduction.

The Tier 1 emergency alert tier triggered at three days before expiration was activated twice during the monitored period, both attributable to certificates issued outside the standard automation framework by vendor teams. In both cases, automated renewal initiation resolved the situation at the Tier 3 alert stage (14 days) without requiring emergency escalation, confirming that the tiered alert protocol provides sufficient lead time for remediation when monitoring is applied to all certificates including vendor-deployed ones.

C. The Short-Lived Certificate Trajectory and Renewal Volume Implications

Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Table IV: Annual Renewal Volume Analysis by Certificate Validity Period

Certificate Estate Size	Annual Renewals @ 398-day validity	Annual Renewals @ 90-day validity	Annual Renewals @ 47-day validity	FTE Required (manual, 45 min/renewal) @ 47-day
500 certificates	~460/year (~2/day)	~2,030/year (~8/day)	~3,880/year (~15/day)	~2 FTE
1,000 certificates	~915/year (~4/day)	~4,060/year (~16/day)	~7,760/year (~30/day)	~3.7 FTE
3,000 certificates	~2,750/year (~11/day)	~12,180/year (~47/day)	~23,280/year (~89/day)	~11 FTE
5,000 certificates	~4,580/year (~18/day)	~20,300/year (~78/day)	~38,800/year (~149/day)	~18.5 FTE
10,000 certificates	~9,165/year (~35/day)	~40,600/year (~156/day)	~77,600/year (~298/day)	~37 FTE

Table IV quantifies the renewal volume implications of the validity period trajectory across five representative certificate estate sizes. For a 3,000-certificate estate representative of mid-to-large enterprise organizations the transition from the current 398-day maximum to 90-day validity increases annual renewal volume from approximately 2,750 to 12,180 per year. At the projected 47-day validity, renewal volume reaches 23,280 per year, requiring approximately 11 full-time engineers working exclusively on certificate renewal if manual processes are maintained. These figures demonstrate that the industry trajectory makes automation a mathematical necessity for any organization above the 500-certificate threshold, regardless of organizational preference for process-based approaches. Author's primary research data from enterprise deployments confirms that automation frameworks handling this renewal volume add negligible operational overhead automated renewal executes in 3–8 minutes per certificate including validation, compared to 45 minutes for manual renewal with equivalent quality assurance.

D. cert-manager and Vault PKI Implementation Patterns

Kubernetes-native certificate management using cert-manager reduces certificate lifecycle management to a declarative specification. Engineers declare that a service requires TLS, specify the issuing authority and validity parameters, and cert-manager ensures the requirement is continuously satisfied: monitoring the referenced Kubernetes Secret, initiating renewal at the specified lead time (typically 30 days for 90-day certificates), executing the ACME or CA API validation workflow, storing the renewed certificate and key in the Secret, and triggering service restart through deployment annotations all without human intervention. This model scales linearly with certificate estate size without proportional labor increase, enabling organizations to support the 47-day validity trajectory by increasing automation capacity rather than hiring additional engineers.

HashiCorp Vault PKI short-lived internal certificate architecture eliminates the revocation problem that has historically made certificate-based zero-trust service authentication operationally complex. Certificates with 24-hour validity issued to Kubernetes services through Vault's PKI secrets engine expire so rapidly that revocation infrastructure becomes unnecessary for the most common

Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

compromise scenario: a compromised private key becomes invalid within 24 hours regardless of administrative action, compared to the months-long exposure window that annual certificate validity creates under CRL/OCSP-dependent revocation architectures.

E. Automation Failure Modes and Prevention

Three automation failure modes require explicit architectural countermeasures. Silent automation failures exemplified by the Azure 2020 incident are prevented through post-deployment live endpoint verification as the mandatory final step of every renewal operation, executed against every dependency graph node. Certificate pinning compatibility is managed through a maintained inventory of pinning clients per certificate, distinguishing between end-entity certificate pinning (broken by renewal) and public key pinning (maintainable across renewals with key reuse), with renewal coordination through mobile release cycles for pinning clients. Private key security in automation contexts requires exclusive storage in dedicated secrets management systems (HashiCorp Vault, AWS Secrets Manager, Azure Key Vault) with short-lived credentials, audit logging, and no local persistence of key material between operations. The automation infrastructure itself is governed as a high-value security target with access controls equivalent to the Certificate Authority infrastructure it manages.

V. Conclusion

SSL/TLS certificate lifecycle management has evolved from a manageable operational concern to a critical security and availability risk requiring systematic automation. Three conclusions of direct relevance to enterprise security and infrastructure strategy emerge from the analysis presented in this article.

The financial and operational impact of certificate-related failures is substantial and structurally driven by dependency graph complexity rather than by individual certificate value. The Ericsson and O2 incident, the Microsoft Azure MFA outage, and the LinkedIn mobile failure each demonstrate that certificate expiration propagates through dependent infrastructure components to produce cascading failures whose scope far exceeds the single certificate point of failure. The average enterprise production certificate's dependency graph of 8.3 infrastructure components means that a single expiration event can produce simultaneous failures across all components referencing that certificate. The four-layer automation framework prevents this scenario by maintaining the dependency graph as a living data structure and verifying every graph node following each renewal operation.

The most dangerous automation failure mode, a framework that reports success while leaving live endpoints in a degraded state requires explicit prevention architecture. Inventory-based monitoring is insufficient for detecting this failure mode. Only live endpoint verification against every dependency graph node, executed as the concluding step of every renewal operation, provides the detection capability that prevents partial deployment from persisting undetected. The five-minute post-deployment validation window demonstrated in enterprise deployments under the author's oversight confirms that this detection is both achievable and operationally practical.

The Certificate Authority/Browser Forum trajectory toward 90-day and projected 47-day maximum certificate validity periods converts automation from a best practice to a mathematical necessity for enterprises managing certificate inventories of 500 or more certificates. Organizations that complete automation framework implementation before the 90-day validity transition applies to their estate will be operationally indifferent to the change. The architectural destination certificates provisioned automatically as a consequence of service deployment, renewed without human intervention, validated continuously through live endpoint verification, and governed by policy enforced through the delivery pipeline represents the elimination of certificate expiration as a class of operational risk, which is achievable for every enterprise that commits to the systematic implementation of the framework presented in this article.

Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

References

[1] P. Lee, "Ericsson Software Glitch Caused O2 Network Outage Affecting Tens of Millions," The Guardian, December 2018. [Online]. Available: https://telecomdrive.com/ericsson-software-glitch-causes-network-outage-for-o2-softbank/

[2] Microsoft, "Increasing Transparency into Azure Active Directory's Resilience Model," Microsoft Tech Community, February 2021. [Online]. Available: https://techcommunity.microsoft.com/t5/microsoft-entra-azure-ad-blog/increasing-transparency-into-azure-active-directory-s-resilience/ba-p/2147047

[3] T. Mah, "LinkedIn Suffers SSL/TLS Certificate Expiration Again," The SSL Store, 2022. [Online]. Available: https://www.thesslstore.com/blog/linkedin-suffers-ssl-tls-certificate-expiration-again/

[4] R. Sleevi, "Moving Forward, Together: Shortening TLS Certificate Lifetimes," Google Security Blog, March 2023. [Online]. Available: https://security.googleblog.com/2023/03/moving-forward-together-chrome-root.html

[5] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," IETF RFC 5280, May 2008. [Online]. Available: https://www.rfc-editor.org/rfc/rfc5280

[6] Ofcom, "O2 Network Outage: Consumer Enforcement Bulletin," Office of Communications, London, UK, Rep. CW/01235/12/18, Nov. 2019. [Online]. Available: https://www.ofcom.org.uk/__data/assets/pdf_file/0014/175010/o2-network-outage-cceb.pdf

[7] J. Aas, R. Barnes, B. Case, Z. Durumeric, P. Eckersley, A. Flores-López, J. A. Halderman, J. Hoffman-Andrews, J. Kasten, E. Rescorla, S. Schoen, and B. Warren, "Let's Encrypt: An Automated Certificate Authority to Encrypt the Entire Web," in Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS), London, UK, Nov. 2019, pp. 2473–2487. [Online]. Available: https://dl.acm.org/doi/10.1145/3319535.3363192

[8] Ponemon Institute, "2023 State of Machine Identity Management," Keyfactor-Ponemon Research Report, 2023. [Online]. Available: https://www.keyfactor.com/state-of-machine-identity-management-2023/

[9] A. Langley, "No, Don't Enable Revocation Checking," Imperial Violet Blog, April 2014. [Online]. Available: https://www.imperialviolet.org/2014/04/19/revchecking.html

[10] R. B. Barnes, J. Hoffman-Andrews, D. McCarney, and J. Kasten, "Automatic Certificate Management Environment (ACME)," IETF RFC 8555, March 2019. [Online]. Available: https://www.rfc-editor.org/rfc/rfc8555

[11] cert-manager Authors, "Introduction cert-manager Documentation," cert-manager Project, 2024. [Online]. Available: https://cert-manager.io/docs/

[12] S. Rose, O. Borchert, S. Mitchell, and S. Connelly, "Zero Trust Architecture," National Institute of Standards and Technology Special Publication 800-207, August 2020. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-207.pdf

[13] Open Policy Agent, "Introduction," Open Policy Agent Documentation, 2024. [Online]. Available: https://www.openpolicyagent.org/docs/latest/

[14] HashiCorp, "PKI Secrets Engine Considerations," HashiCorp Vault Developer Documentation, 2024. [Online]. Available: https://developer.hashicorp.com/vault/docs/secrets/pki/considerations

[15] U.S. Cybersecurity and Infrastructure Security Agency (CISA), "StopRansomware: Citrix Bleed," CISA Advisory AA23-325A, November 2023. [Online]. Available: https://www.cisa.gov/news-events/cybersecurity-advisories/aa23-325a

Journal of Information Systems Engineering and Management
2026, 11(3s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

[16] J. Peterson, M. Barnes, D. Hancock, and C. Wendt, "Automated Certificate Management Environment (ACME) Challenges Using an Authority Token," IETF RFC 9447, September 2023. [Online]. Available: https://www.rfc-editor.org/info/rfc9447

[17] J. Larisch, D. Choffnes, D. Levin, B. M. Maggs, A. Mislove, and C. Wilson, "CRLite: A Scalable System for Pushing All TLS Revocations to All Browsers," in Proc. IEEE Symp. Secur. Privacy (S&P), San Jose, CA, USA, May 2017, pp. 539–556. [Online]. Available: https://ieeexplore.ieee.org/document/7958597/

[18] M. Abdalla, P.-A. Fouque, and D. Pointcheval, "Password-Based Authenticated Key Exchange in the Three-Party Setting," IEE Proceedings Information Security, vol. 153, no. 1, pp. 27–39, 2006. [Online]. Available: https://digital-library.theiet.org/content/journals/10.1049/ip-ifs_20055073

[19] P. Szalachowski, L. Chuat, T. Lee, and A. Perrig, "RITM: Revocation in the Middle," in Proc. IEEE 36th Int. Conf. Distrib. Comput. Syst. (ICDCS), Nara, Japan, Jun. 2016, pp. 557–566. [Online]. Available: https://ieeexplore.ieee.org/document/7536518/

[20] Z. Durumeric, J. Kasten, M. Bailey, and J. A. Halderman, "Analysis of the HTTPS Certificate Ecosystem," Proceedings of the Internet Measurement Conference (IMC), 2013, pp. 291–304. [Online]. Available: https://dl.acm.org/doi/10.1145/2504730.2504755

[21] D. Kumar, Z. Wang, M. Hyder, J. Dickinson, G. Beck, D. Adrian, J. Mason, Z. Durumeric, J. A. Halderman, and A. Bates, "Tracking Certificate Misissuance in the Wild," Proceedings of the IEEE Symposium on Security and Privacy, 2018, pp. 961–978. [Online]. Available: https://ieeexplore.ieee.org/document/8418638/