Mutual Monitoring in the Cloud

Problem Statement

Cloud service providers must cater to customers in regulated sectors with high barriers to entry. One barrier is evaluation of the provider's cybersecurity posture, mostly with centralized bureaucracies. Although preemptively limiting cybersecurity risk, such mandatory evaluation often means significant delay and investment before regulated customers can use new or changing building blocks they urgently need for their digital services. Are these bureaucracies an optimal solution or a last resort that failed to keep pace with cloud technology? If the latter, is there a better way?

Deliverables

Critical Analysis of FedRAMP

Brief History and Context

Current FedRAMP Process

Initial Authorization

Continuous Monitoring

Agency versus JAB Authorizations

20x

Continuous Monitoring in Detail

Benefits

Challenges and Problems

Mutual Monitoring Architecture

Prototype Code

Foundational Quantitative Metrics Framework

Appendix

Research

Transparency systems and architecture

- (Rescorla, 2023b)

  - The key word in this system is trust: the endpoints need to trust that the authentication service doesn't falsely attest to a binding for the wrong person (technical term: "misissuing").
  - There are a number of potential approaches for defending against this problem but the one that the community seems to have settled on is what's called a transparency system. The basic concept of such a system is that you retain the idea of a trusted authentication service but add on a layer in which it publishes the bindings it is attesting to so that anyone can check that it's not misissuing.
  - The basic idea behind a transparency system is not to prevent misissuance but to detect it.
  - The obvious thing to to do is for the AS to just publish the list of certificates it has issued on its Web site, but this isn't secure. Consider what happens if the AS gives different answers to different people, like so [...]
  - Moreover, it's not really required that everyone get a full copy of the database: consider the case where we have a fake certificate for example.com.
  - The problem, obviously, is that this kind of flood fill is incredibly inefficient: Let's Encrypt alone has about 300 million valid certificates; at 1K each, this would be a database of 300GB, not something you want to be storing on your phone, let alone having to send to everyone else you come into contact with—ignoring for the moment the question of how you're going to transmit the database around. Clearly, this simple system is not practical.
  - The idea behind a Merkle Tree is to allow a way to efficiently commit to a set of values without actually publishing any of the values.
  - The inclusion proof is comparatively small; using Let's Encrypt as our reference point, it will be about 600-700 bytes.
  - This isn't perfect in that you still have to actually detect misissuance, which isn't always straightforward for the reasons I discussed above, but at least it's not possible to have covert misissuance.
- (Rescorla, 2023a)
  - This design has several advantages. First, it makes life easier for the CAs, who don't have to run logs. This may not seem like a big deal, but it turns out that running a log is a lot of work for reasons we'll get into below, and indeed very few CAs actually run their own logs today. Instead, some entity with a lot of operational resources and experience (i.e., Google), could run a log that supports multiple CAs, hopefully making it easier for the CAs to deploy.
  - Second, having a relatively small number of logs improves the scaling properties of the system somewhat: much of the overhead for the clients comes in the form of getting an authentic copy of the signed root (what CT calls a signed tree head (STH)), and if each CA has its own tree, that means one root for each CA. If there's just a small number of logs then you need a correspondingly smaller number of

roots.

- Finally, the log design makes it possible to publish certificates even for CAs which don't participate because the log can just unilaterally ingest those certificates.
- In order to address this issue, Google introduced a new concept, the signed certificate timestamp (SCT). An SCT is a signed promise that the log will add the certificate to their tree soon, even though they haven't yet. The figure below shows the issuance flow with SCTs.
- The good news is that CT with SCTs is minimally disruptive while also allowing the browser to enforce the use of CT. The bad news is that it has totally different and much weaker security properties from the system we started with. The problem is that the SCT is just a promise that the log will incorporate the certificate into their Merkle tree, rather than a proof that it actually did, so you're reduced to trusting the log not to lie.
- Because the source of the problem is that the client isn't verifying inclusion of the certificate (by checking the inclusion proof) but only that the log says it would include it (by checking the SCT), the obvious fix is to have the client somehow verify that the certificate actually was included. This turns out to be somewhat challenging and there have been a number of attempts, none of which really work.
- In order to prevent this form of tracking, we need some way for the client to retrieve the inclusion proof anonymously. There are a number of possible options here (VPNs or proxies) or Private Information Retrieval. As far as I know, no log deploys any kind of PIR—it would probably be quite expensive—and while proxies or VPNs are technically feasible, they're not free to run. There are similar problems with clients reporting certificates which are not included but should have been. I'm not aware of any major browser which verifies certificate inclusion proofs [Update 2023-12-25] by default (Chrome had some ideas about using DNS,[2] but seems to have abandoned them.[3]), though see below.
- this seemed kind of impractical when CT was originally designed, but in the intervening 10 years, automatic certificate issuance has become far more common (specifically, a protocol called ACME, originally developed for Let's Encrypt), and so it wouldn't be that hard to imagine modifying ACME to send an updated certificate. Importantly, this is something that could be deployed incrementally, because clients have to be able to fall back to SCTs anyway. However, it doesn't seem to be something that's happening.
- Even if we did have some mechanism for verifying the inclusion proof, we still have the problem of getting consensus on the STHs. The original CT design assumed a flood fill technique (what they called "gossip") like I described in part I, but was frustratingly short on specifics [. . .]
- The problem is that it's expensive futureproofing, both in terms of

protocol complexity and in terms of operational brittleness. A fairly large fraction of the CT RFC is concerned with specifying the Merkle trees, the machinery of Merkle tree proofs, and the like. All of this could just go away if we were to just treat CT as a "countersign + publish" protocol, leaving a dramatically simpler protocol that would be a thin layer on top of HTTP.
  - Worse yet, CT logs turn out to be hugely operationally complex to run correctly. I haven't personally operated one, but the basic problem seems to be tight timing requirements combined with the immutability of the Merkle tree structure.
  - Despite everything I've said above about the limitations of CT verifiability, it's still proven to be exceedingly useful.
  - By contract, CT is yet another patch on top of the WebPKI, but was incrementally deployable. Imperfect though it is, it has gone a long way towards improving the system, both by making undetected misissuance harder and by making simple misbehavior easier to spot and address.
- (Rescorla, 2024)
  - My long-time collaborator Richard Barnes[1] used to say that "in security, trust is a four letter word", and yet the dominant experience of using any software-based system—which is, you know, pretty much anything electronic—is trusting the manufacturer.
  - The release then gets uploaded to the vendor's website, which is probably hosted on some cloud service like Amazon or Netlify. You can also host the binaries on GitHub. In principle, users could just download the binaries directly from your site (or GitHub) but it's common to instead use a content distribution network (CDN) like Cloudflare or Fastly which retrieves a copy of the binary once, caches it, and then gives out copies to each user. CDNs are designed for massive scaling, thus saving both load on your servers and cost.
  - The basic problem with code signing systems is that they rely heavily on user diligence, because the OS only verifies that the code was signed but doesn't know who was supposed to sign it.
  - Of course, if you have to download software over secure transport anyway, this raises the natural question of why bother to sign the code at all? Why not just have all downloads happen over secure transport? One reason is that is that signing allows for third party hosting.
  - If you require software to be signed by some key that chains back to some non-free credential, then this allows you to increase the level of friction to distribute all software, but especially malicious software.
  - Using secure transport to the package repository is standard practice, but it doesn't help against either of these threats because the problem is the data on the package manager itself is compromised. In theory it seems like signatures offer a way out of this: if packages are signed then even if the attacker compromises the repository they won't be

able to replace the package with their own. Unfortunately package signing isn't a complete solution for the same kind of identity reasons as before.
  - Of course, this was all warmup for the real problem we want to solve. Everything up to now was about ensuring that you get the binary that the publisher wanted to send you. This still leaves you trusting the publisher, which you shouldn't, both because it's bad security practice to have to trust people and because there is plenty of evidence of software publisher misbehavior.
  - The process starts with the publisher releasing the source code. As a practical matter, some kind of review of the source code is a necessary but not sufficient precondition to being able to have confidence in a piece of software. Reviewing the binary is not really practical on any kind of scalable level; it is of course possible to reverse engineer binaries, but it's incredibly time consuming even for experts.
  - BT is like Certificate Transparency (CT) but instead of publishing every certificate, the publisher instead publishes a hash of every binary they release.
  - The sigstore project provides generic tooling for binary signing, reproducible builds, and binary transparency and seems to be getting some uptake. However, we're not seeing the kind of large-scale deployment that we have for certificate transparency, and there don't seem to be any generic logs in wide use like there are with CT. Facebook has also deployed a system called Code Verify to provide a form of binary transparency for Facebook Messenger.
  - We're well over 7000 words already, but I do just want to briefly touch on the topic of the Web. The Web has a number of properties that do make the problem somewhat easier [...] However, it also has several important properties that make the problem much harder: [...]
- (Kuerbis & Mueller, 2023)
  - Data enclosure is defined here as the process by which the information about user activity generated by digital operations are withdrawn from an open or shared arrangement with other operators and made more exclusive to the service providers whose operations generate the data. (p. 1)
  - But data enclosures do not just respond to users' and governments' privacy concerns, they may also provide a competitive advantage to a platform by excluding their competitors or other players from access to data generated by their users. Internalizing the costs of protecting privacy can also privatize the economic benefits of data. Any realistic appraisal of the platform economy must look at both sides of this equation. (p. 2)
  - 'Data' is often reified as the resource of the digital economy, but this can be misleading. It implies that the platform economy's value stems from a recorded and stored pile of bits and bytes that, like oil ("data is the new oil"), can be "mined" for its economic value. This view

of data as a static commodity obscures the operational and business reality of the platform economy. Platforms are multisided markets (Evans & Schmalensee, 2016). (p. 2)

- Contrary to the popular critique characterizing users as exploited "unpaid data producers" (Pistor, 2020), a mutual economic benefit underlies the user-platform interaction. Consumers are "paid" by the platforms' provision of free information services and matches. Many of these services are incredibly valuable but now seem to be taken for granted, perhaps because of the platforms' success at making them ubiquitous: search engines, email, informational and entertainment content, document storage, and thousands of connections to products, other individuals and groups. Economists working out the logic of multi-sided markets have repeatedly demonstrated how it makes business sense for one side to subsidize entry or participation costs of another side (Rochet & Tirole, 2006; Parker et al., 2016). It is also misleading to suggest that the users alone are the "producers" of the data, or that data is being "extracted" from them. While their activities do provide content, without the platform infrastructure itself there are no data and no useful applications for the data. Even severe but realistic critics of the digital market economy (Benthall & Goldenfein, 2021) admit that the data is co-produced. (p. 2)
- Data enclosure is a way to create exclusive access to data, and thus raises the issue of property rights in information. (p. 2)
- In a purely abstract sense, digital data seems to resist exclusivity. Digital data is nonrival in consumption (i.e., one person's use of it does not consume or "use up" the resource). Because of the rapid, practically costless way it can be duplicated and transmitted, it is also notoriously difficult to contain. (p. 2)
- In their data enclosure initiatives, platforms and other digital service providers are establishing de facto control over the data by contractual and technological means. Platforms withhold from others operational data generated by their users, instead of sharing it. (p. 3)
- An analysis of data enclosure suggests that in a contested, multi sided market, a digital platform provider competing for users would have two distinct incentives: 1) an incentive to use the privacy and security of their users' data as a product differentiator; 2) an incentive to assert or maintain exclusivity over the data co-generated by its users and its infrastructure to maintain or increase a competitive advantage in the provision of intermediation. (p. 3)
- However, the privacy differentiation incentive might also push them toward limiting potential ways of monetizing the data/users to which they have exclusive access. (p. 3)
- For its part, Google did not come up with the idea for DoH, did not promote it as part of its competitive strategy, nor did it implement it in a way that could have reinforced its advantages. Google could have leveraged the dominance of its Chrome browser to implement

DoH in a way that defaulted DNS resolution to its own DoH resolver service. It refrained from doing so. Google went along with DoH, but did not lead it. But then, it did not need it. (p. 7)

- (Laurie, 2014)
  - I have written extensively on what is wrong with Bitcoin (for example, it is the least green invention ever, and all of its history could be destroyed by a sufficiently powerful adversary if it were truly decentralized, which it is not). Nevertheless, people continue to worship irrationally at the altar of Bitcoin, and this worship extends to DNS and keys—for example, DNSChain (https://github.com/okTurtles/dnschain). Apart from being an extremely costly solution (in terms of wasted energy, in perpetuity), it also introduces new trusted third parties (those who establish the "consensus" in the block chain) and has no mechanism for verification.
- (Aldribi, Traore, & Letourneau, 2015)
  - If a provider is unwilling to disclose information about their security features, we propose allowing clients to deploy their own network level security. Unfortunately, the current cloud model does not allow clients any network level access [2] (see section 3). Therefore, there is a fundamental relationship between the decision making of stakeholders and security transparency problems. (p. 18)
  - Cloud slicing uses hardware partitioning (sometimes named logical partitioning) to allow a client access to the network level of the server. With network level access, clients can implement further protection measures to give assurance to themselves and their customers. (p. 18)
  -

Continuous assessment and assessment methodologies

Continuous cloud monitoring

- (Campitelli, Catteddu, & Maria, 2020)
  - As users and the uses of cloud computing evolve, so must the supporting governance models – this includes the maturity of governance and risk management programs designed to review and evaluate the selection, adoption, and migration of traditional operations to Cloud Service Providers(CSPs). (p. 5)
  - This position paper serves to provide an impartial look at risk by identifying and examining gaps introduced over the last 10 years by the rapid adoption of Cloud Computing. (p. 6)
  - Regulatory compliance is a basic part of doing business. Noncompliance can be an expensive proposition; it diverts the organization's attention from normal operations, attracting scrutiny from regulators that can result in additional fines, repetitive audits, potential legal action, and reputational downturn. (p. 7)

  - The market size of public cloud computing has grown from $58.6Bin 2008 to a projected volume of $266.4B in 2020. (p. 7)
  - Hence, the design of their cloud ecosystem becomes just as important as its operational performance. These changes need to be recognized by the risk management process since it can raise both the category and level of risk. At stake is the entire enterprise's ability to meet its strategic objectives, including the ability to survive in the marketplace. (p. 8)
  - Introduction of new complexities: to create, operate, manage, measure and report upon performance in a cloud-based ecosystem, the organization's risk management system must accommodate a host of new complexities that have dramatically increased the scope and scale of the risk equation (p. 9)
  - This reality of [shared responsibility in] the cloud computing model exacerbates the traditional risk management consideration for third-party IT services. It is complicated by the factors of scale due to the automation of processes and procedures which might create issues of mistake propagation, as well as the risk of hidden interdependencies; visibility, due to the absence of evidence (required logs) to support control process operation, and performance; scope, due to the volume of cloud services and service providers in the supply chain; and continuous monitoring, due to the necessity of real-time, inline information consumption and production and alerts. (p. 10)
  - For instance, CSPs typically prohibit security scanning, penetration testing, and first-party audit by contract. Business continuity and/or disaster recovery testing is often not possible; customer security and privacy policies, including related standards and procedures such as hardening, may conflict with the service provider's efforts; access to key logs and visibility of alerts may be unattainable. (p. 10)
  - Moreover, the key contracting feature of prescriptive SLAs may not be available or measured and reported transparently to the customer if available. (p. 10)
  - While some have described these issues as loss of governance, we perceive them as a driver for a new governance model over cloud service providers. (p. 10)
  - The contractual relationship between the SaaS provider and their cloud usage is not transparent to the customer, creating a problem of information asymmetry. (p. 11)
  - Building on the concept of indirect control discussed in the previous paragraph, such a lack of visibility into the supply chain raises the simple question of how a cloud customer can perform a proper assessment of the security and privacy risks without having access to complete information – a typical issue of accountability. (p. 11)
- (Kunz, Schneider, & Banse, 2022)
  - A central challenge in cloud security certification is to evaluate if a set of measurements is sufficient to prove a pre-defined certification

requirement. The goal of our approach is to quickly assess new threats, rather than complying with a pre-defined set of requirements. As such, we do not only decouple expert analyses from the application of their results, but integrate them in periodic re-assessments. (p. 1)
- Also, configuration monitoring tools and approaches, like GMonE and Clouditor, exist. Yet, they lack the possibility to identify combinatorial CWs, since they just identify vulnerable configurations of single resources. (p. 1)
- In summary, current literature does not sufficiently integrate regular expert-based threat analyses in the risk assessment process. (p. 2)
- Before the continuous process starts, the risk assessment activities are prepared. For instance, relevant assets, i.e. cloud resources that should be assessed, are identified (p. 3).
- ... the company employs several IoT Hubs that accept traffic and store messages in the blob storage. The underlying configurations that enable this threat therefore include public reachability of the IoT Hub, and its access to the blob storage (see also Figure 3). (p. 4)
- Newly identified CWs may also be shared with collaborators by contributing them to the shared repository. (p. 4)
- Module 2 covers the continuous application of the previously defined threat profiles to the cloud system. We describe the activities of this module on a high-level here, and present an implementation in Section IV. (p. 5)
- Risk Calculation: The risk calculation is performed according to the initially defined risk model. In our example we multiply the threat value of an asset's protection goal with the respective impact value—both of which are within the interval of [1, 3]—resulting in a risk value in the interval of [1, 9]. This calculation is done again using Rego policies. (p. 6)
- The inputs for the risk calculation policies are the identified threats, as well as the threat and impact values defined in Module 1. The output then contains the risk scores for all assets and their protection goals which are calculated again using a Rego policy. This policy calculates the risk values for all assets and their protection goals according to the threat values that have been assessed before before. (p. 6)
- Manual analyses: A limitation of our approach is that it does not completely alleviate the necessity for conducting manual risk assessments since the Module 1 activities still need to be performed periodically. (p. 7)
- Concerning the shared threat profile repositories, cloud users may see a risk in sharing their threat profiles with others, since they reveal information about the architecture of their system. (p. 7)
- Our approach is furthermore limited by the properties that IaC templates offer, i.e. resource configurations on infrastructure-level. Risk assessments, however, may also include other types of CWs,

such as social engineering or application-level threats. Note that such threats can still be integrated into the threat profiles, e.g. by adding a social engineering weakness to every resource of the system. (p. 7)
  - The set up and maintenance of the Module 2 components can introduce an overhead in comparison to traditional security audits and infrastructure monitoring tools. The biggest overhead our approach introduces may be the maintenance of threat profiles which can be necessary for several reasons. (p. 8)
- (Majumdar et al., 2019)
- (Torkura, Sukmana, Cheng, & Meinel, 2021)
  - Most of these attacks are caused by customer misconfigured cloud resources e.g. over-privileged users, publicly exposed databases, and lack of audit logging (MacDonald, 2019). (p. 1)
  - However, the upper layer of cloud infrastructure (control plane), which customers are responsible for securing has increasingly become vulnerable to attacks introduced by misconfigurations and human errors. According to Gartner, 99% of cloud failures will be directly caused by customer errors, until 2025. (p. 1)
  - For example, Cloud compute and cloud networks leverage traditional security control like firewalls and Intrusion Detection Systems (IDS) (Takabi et al., 2010) to enforce detective and preventive security controls. However, these security controls are ineffective for protecting some cloud services including cloud storage and IAM. (p. 2)
  - The Shared Security Responsibility Model (SSRM) is a popular security model employed by public CSPs (AWS, 2020b; Institute, 2019b). It clearly defines security responsibilities for cloud stakeholders. The ability to detect and mitigate the attacks highlighted in the running example is the responsibility of cloud users, however, most cloud users do not clearly understand these responsibilities, or lack the tools to implement commensurate technical and organizational measures (p. 3)
  - However, there is a gap between these high-level benchmarks and commensurate low-level, technical implementation (Majumdar et al., 2019). A typical example of this gap is the lack of mapping of wide adopted security metrics to the CIS benchmarks.
  - While compliance benchmarks are good, they do not equal to security. Therefore, another category of policies called Enterprise Security Policy (ESP) are implemented. ESPs are aligned with enterprise security goals and internal policies including access control regulations for various organizational units e.g. Finance and Human Resources. (p. 4)
  - When cloud resources are already deployed on one or more cloud platforms, a discovery process is employed to enumerate all cloud resources. The discovered resources are subsequently adopted as the expected-state (or added to an already established expected-state). The discovery process leverages cloud APIs to acquire the

comprehensive list of deployed cloud resources. During the discovery process (as illustrated in Fig. 3) CSBAuditor queries the cloud infrastructure for important information of the deployed resources based on the already defined object (using IaS paradigms). (p. 5)
  - Due to the rapid changes that occur in the cloud, it is imperative to deploy a control mechanism that leverages automation, hence the need for the SCC. The SCC leverages the following mechanisms ensure dynamic SecCM ... (p. 5)
  - Early detection of malicious changes is crucial as such events become important IoCs, hence the need for continuous monitoring and auditing strategies (CSA, 2020). (p. 5)
  - The RCA is imperative for investigate detected changes or failures for security reasons. This is a critical requirement for security assurance, the knowledge of what went wrong? is essential for employing detective and preventive counter-measures since changes e.g. disablement of logging, might indicate on-going attacks. (p. 6)
  - However, logs are not available in real-time for public cloud systems. Log delivery takes about 20 min on AWS, and over 90 min on Google Cloud Platform (GCP). Therefore it is imperative to evolve techniques that overcome this gap that exposes a window of opportunity for attackers. (p. 6)
  - Section 3.7 should describe alerting and reporting, but only talks about alerting. (p. 8)
  - However, we assume that the CSP may be trusted for the integrity of the audit data (e.g., access policies and IAM policies) collected through API calls. (p. 10)
  - A limitation of our methodologies: reconciler pattern and state transition analysis is the inability to reverse changes against some cloud resources. These resources include databases, object storage. Essentially, advanced efforts are required to achieve these since the actual content cannot represented in the expected-state using either IaC or IaS. (p. 17)
  - CSPM is a relatively new category of security tools and there are very few implementations. Most CSPM are commercial and proprietary therefore it is quite difficult to compare these tools with CSBAuditor for evaluation reasons. Similarly, traditional security tools like firewalls and IDS are much easier to evaluate since their are public data sets available for conducting experiments. This is not the case also for CSPM. These two major factors pose challenges for deeper and comparative evaluation. (p. 19)
- (Weir & Aßmuth, 2024)

FedRAMP

- (Lewis Commission, 2025)
  - "Fully automate FedRAMP processes. Eliminate written, qualitative

assessments and document review in favor of automated security controls that can be verified through the continual assessment of network security telemetry. Rather than requiring cloud service providers (CSPs) to provide proof that they have met control requirements that cannot be automated and have those artifacts reviewed, CSPs should be allowed to 'attest' that they have met requirements for areas such as personnel training and documentation." (p. 2)
  - "The slow pace of government cloud adoption increases both costs and risks for federal operations and hinders the provision of better services to citizens. Cloud technologies offer advantages in modernization, efficiency, cybersecurity, and resilience. An agency can access and use cloud computing resources without having to buy, maintain, or modernize hardware—actions that are problematic given cumbersome federal budgeting and certification processes." (p. 3)
  - "Despite the potential gains, law, regulation, policy, and agency culture all remain obstacles to reaping the benefits of federal cloud adoption." (p. 3)
  - "To place FedRAMP on more solid legal footing, in 2022 Congress passed the FedRAMP Authorization Act. The act strongly encouraged automation and continuous monitoring to speed up the accreditation process." (p. 5)
- (FedRAMP, 2025)
- (Gartner, 2024)
- (stackArmor, 2024)

Aldribi, A., Traore, I., & Letourneau, G. (2015). Cloud slicing a new architecture for cloud security monitoring. *2015 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)*, 18–22. IEEE. https://doi.org/10.1109/pacrim.2015.7334802

Campitelli, V., Catteddu, D., & Maria, J. D. (2020). *CSA's perspective on cloud risk management*. cloudsecurityalliance.org. Retrieved from https://cloudsecurityalliance.org/artifacts/csa-s-perspective-on-cloud-risk-management

FedRAMP. (2025). *FedRAMP CSP authorization playbook*. Retrieved from https://web.archive.org/web/20250413105351/https://www.fedramp.gov/assets/resources/documents/CSP_Authorization_Playbook.pdf

Gartner. (2024). *Gartner forecasts worldwide public cloud end-user spending to total $723 billion in 2025*. Retrieved from https://web.archive.org/web/20250415023404/https://www.gartner.com/en/newsroom/press-releases/2024-11-19-gartner-forecasts-worldwide-public-cloud-end-user-spending-to-total-723-billion-dollars-in-2025

Kuerbis, B., & Mueller, M. (2023). Exploring the role of data enclosure in the digital political economy. *Telecommunications Policy*, *47*(8), 102599. https://doi.org/10.1016/j.telpol.2023.102599

Kunz, I., Schneider, A., & Banse, C. (2022). *A continuous risk assessment methodology for cloud infrastructures*. arXiv. https://doi.org/10.48550/ARXIV.2206.07323

Laurie, B. (2014). Certificate transparency: Public, verifiable, append-only logs. *Queue*, *12*(8), 10–19. https://doi.org/10.1145/2668152.2668154

Lewis Commission. (2025). *Faster into the cloud: Accelerating federal use of cloud services for security and efficiency*. Center for Strategic; International Studies. Retrieved from https://web.archive.org/web/20250116234900/https://csis-website-prod.s3.amazonaws.com/s3fs-public/2025-01/250115_Lewis_Cloud_Commission.pdf

Majumdar, S., Madi, T., Wang, Y., Tabiban, A., Oqaily, M., Alimohammadifar, A., ... Debbabi, M. (2019). *Cloud security auditing*. Cham, Switzerland: Springer Nature. Retrieved from https://link.springer.com/book/10.1007/978-3-030-23128-6

Rescorla, E. (2023a). *A hard look at certificate transparency: CT in reality*. Retrieved from https://educatedguesswork.org/posts/transparency-part-2/

Rescorla, E. (2023b). *A hard look at certificate transparency, part i: Transparency systems*. Retrieved from https://educatedguesswork.org/posts/transparency-part-1/

Rescorla, E. (2024). *Why it's hard to trust software, but you mostly have to anyway*. Retrieved from https://educatedguesswork.org/posts/ensuring-software-provenance/

stackArmor. (2024). *How much does FedRAMP compliance cost?* Retrieved from https://web.archive.org/web/20240808151743/https://stackarmor.com/how-much-does-fedramp-compliance-cost/

Torkura, K. A., Sukmana, M. I. H., Cheng, F., & Meinel, C. (2021). Continuous auditing and threat detection in multi-cloud infrastructure. *Computers &Amp; Security*, *102*, 102124. https://doi.org/10.1016/j.cose.2020.102124

Weir, G. R. S., & Aßmuth, A. (2024). *Strategies for intrusion monitoring in cloud services*. https://doi.org/10.48550/ARXIV.2405.02070