Lavontay Johnson

Prof. Leech

CYSE 431

24 April 2024

RMF Playbook

Abstract

This playbook documents how I applied the NIST Risk Management Framework (RMF) to a made up but realistically designed system that I created called SignalShield, which is a secure, decentralized emergency communication platform that I imagined being utilized in disaster response where infrastructure is unreliable or completely unavailable. SignalShield was envisioned based on three underlying technologies: portable mesh networking, satellite connectivity, and a secure cloud dashboard. These elements are incorporated to assist responders, public authorities, and field personnel during crises. I followed the RMF process established in NIST SP 800-37 Rev. 2 (NIST, 2020a) and went through each of the lifecycle phases, including categorization, control selection, implementation, assessment, authorization, and ongoing monitoring.

Throughout this playbook, I've tried to do more than simply tell you what I did, but also, I write about why I made certain decisions, how I balanced real world usability concerns, and what I found out about how to design controls for high impact, high risk systems. I think that the application of RMF to a system such as SignalShield demonstrates the adaptability and utility of the framework, especially if accomplished with a focus on mission success rather than compliance checkboxes. I also think that this project made me view cybersecurity in a more

human light. It is not merely data that is being protected, it is people who are being protected and critical missions facilitated. My hope for this playbook is to demonstrate how RMF can be applied not only to the technical aspects of cybersecurity but also to the operational, emotional, and ethical duties involved in creating systems that will likely be used when lives are at stake.

1. Executive Summary

SignalShield is an emergency communication system I came up with for this project, but I kind of designed and constructed it with real world applications in mind. I thought about it by asking what happens to regular communications infrastructure, cell towers, landlines, and broadband networks when disaster hits. My question to myself was: how do rescue services communicate with each other to coordinate rescue action when there is no good mechanism to do so? That led me to think of SignalShield as one that is portable, secure, and fault tolerant to bridge the void. I think you do need to treat systems such as these as something greater than technical exercises. If you are going to implement them where lives are at stake, then each and every design decision matters.

The architecture consists of an outdoor mesh network of hardened nodes, field deployable satellite uplinks, and an AWS GovCloud-hosted dashboard. These all come together to provide access to the users wherever they are. For first responders in the path of the hurricane, for example, they'd be able to communicate with each other using the mesh network and the satellite uplinks to stay in contact with national command centers. I would say the cloud based dashboard provides incident managers with the ability to track the location of users, send notifications, and make decisions in real time. I think the multi level design provides both on the ground management and top level management, which is important in emergency response.

SignalShield customers are government entities, including FEMA, municipalities, National Guardsmen, and medical responders. Their users may then have very disparate degrees of technical knowledge, so I aimed to provide an experience that was safe but not frustrating. I did not want to let security become intrusive. I wanted it to be so seamlessly integrated that users were not even aware of it at all and they could positively count on it. This perspective influenced how I handled access controls through the type of training that I recommended.

I used the NIST RMF from SP 800-37 Rev. 2 to craft this playbook, using each of the six steps and the continuing activities that connect them. I did not treat RMF as a checklist. I treated RMF as an adaptable roadmap to consider how to defend an existing system under duress, uncertainty, and risk. I believe that's what RMF is necessarily meant to be in my opinion. It's not for data centers, it's for actual systems with actual stakes. That's how SignalShield was implemented, and that's what this playbook is meant to emulate.

2. System Categorization and Selection

When I first thought of SignalShield, I kind of knew that it was going to be different from the majority of information systems. It wasn't going to be in an office park or datacenter. It was going to be deployed where things are disordered, dangerous, and unpredictable, places such as along the path of wildfires, hurricanes, or in the aftermath of an earthquake. Toward that end, I had to consider the system boundaries and system use differently. I believe knowing the unique characteristics of the environment was the basis of the entire remainder.

The architecture has three main components. The first is the mesh network layer, which consists of small, mobile nodes able to relay communications between one and the next independently of fixed infrastructure. The second is a satellite uplink system for long distance connectivity when

ground infrastructure is disrupted. The third is an assured cloud dashboard in the AWS GovCloud with operational monitoring, user administration, and centralized visualizations. These are the system boundary components of the technology, but I include the human users responders, administrators, and technicians too because it is within the boundary of operation where so many threats to the system occur.

In system categorization, I followed FIPS 199 (NIST, 2004), which provides a framework for evaluating the potential impact of confidentiality, integrity, and availability (the CIA triad). I followed SP 800-60 (NIST, 2008) to categorize the kinds of information SignalShield would deal with responder geolocating, triage status, incident reports filed in the field, and command messages. “SP 800 60 provided a framework for mapping SignalShield’s information types to proper impact levels” (NIST, 2008). I considered confidentiality to be moderate because unauthorized disclosure would reveal operations or put personnel at risk. I considered integrity and availability to be high because corrupted or missing data during the response to an emergency situation would amount to catastrophe in the form of mis deployed resources or delayed evacuations. I believe these ratings are reasonable, specifically when I envisioned someone using SignalShield to report the collapse of a bridge or requesting medical aid.

From that process, I determined that SignalShield is an impact focused system, and that decision guided all I did after, all the way from what controls I chose to what I decided to evaluate them on. I also had to consider the operating environment. Power loss, weather damage, and compromised networks were all situations where controls couldn't rely on durable infrastructure. Offline operation and local authentication, for example, were top priorities. I think doing that step enabled me to internalize something that RMF is very good at teaching: security needs to reflect the actual world the system is in, not the one we'd prefer it to be.

3. Control Selection

Having categorized SignalShield as a high impact system, I validated NIST SP 800-53 Rev. 5 (NIST, 2020b) to determine the suitable baseline controls. “Baseline controls for high-impact systems were selected from SP 800 53 Rev. 5 and then tailored for the operational environment” (NIST, 2020b). The high impact baseline is comprised of hundreds of controls in various control families, and I was somewhat confused at first, but as I worked through them, I found myself considering not so much the quantity of controls, but how they might be crafted to reflect the particular mission and environment for which SignalShield was designed. It is at this process of tailoring, I think, that RMF starts to no longer be a checkbox exercise and instead becomes a deliberate design process.

I started by going through all the control families and highlighting those that I knew right away would be most relevant. Access Control (AC), System and Communications Protection (SC), Contingency Planning (CP), and Incident Response (IR) were the natural priorities. These areas are directly related to SignalShield's core goals: making sure people can connect securely, respond to and detect threats, and keep operating even when things go wrong. I also prioritized Audit and Accountability (AU) and Media Protection (MP) because SignalShield would consist of many field-based devices with local data storage.

Then I tailored the baseline to match the reality of my system being deployed. For example, I deleted or adapted controls that presumed the standard enterprise environment such as ones addressing HVAC monitoring or cable shielding inside physical data centers. Those controls weren't applicable to a device operating in the back of an ambulance or atop a mountain. I inserted or highlighted controls addressing offline credential verification, GPS signal spoofing

detection, and tamper evident housing. I also ensured that I considered how controls would function when devices were disconnected for extended durations. I believe one of the finest examples of this was designing a policy for certificate expiry and reauthentication in low connectivity environments.

Inherited controls from AWS GovCloud were a factor as well. “GovCloud inherited controls were used for identity management, encryption at rest, and audit logging” (Amazon Web Services). My view is that inheriting controls is not a matter of cutting corners, it is a matter of investing in the areas of the system you do control. So for example, AWS does encryption at rest, identity management, and access logs at the cloud layer. That allowed me to invest more time in the mesh networking layer, device hardening locally, and offline authentication issues. I clearly documented inherited controls in the SSP and ensured I included instructions on how to ensure they were set up correctly.

In retrospect, I would say this phase of the RMF process made me consider risk ownership more precisely. Tailoring is not merely removing controls, it is also shaping them. I believe the outcome was a set of controls that was mission aligned, was sensitive to the operational realities, and provided the system with a firm foundation for success.

4. Control Implementation

Plugging controls into SignalShield wasn't merely an exercise in best practice, it was a matter of considering how those controls would really stand up in the midst of an emergency. I needed to consider everything from hardware protections down to end-user behaviors, and I believe that's why this was such a difficult but rewarding aspect of the RMF process. Every control needed to be pragmatic, resilient, and simple enough to utilize without compromising security.

Starting with encryption, I employed AES-256 for data at rest and TLS 1.3 for data in motion. I believe these are known standards, and I chose them because they balance security with performance even on mobile devices. I also introduced mesh layer encryption using session based keys that rotate automatically. This was needed because field equipment will need to go on and off the network in a timely manner, and I did not wish to utilize static keys where, if a device were lost, the system would be breached.

Access control was established on the principle of least privilege and adapted for operational use. Administrative users accessing the AWS hosted console utilize role based access control (RBAC) and are required to perform MFA through a hardware token. Field users, however, authenticate based on digital certificates stored on their device. The certificates are bound to user roles and can be remotely revoked or expired. Because field conditions may not favor real time checking of certificates, I implemented a time out based cache. I believe this was an unavoidable compromise, it maintains system availability in the event of failure and continues to enforce security policy.

Patch and update management was also a huge area of emphasis. Devices can be updated with digitally signed updates via mesh or satellite if connected. If neither of those are feasible, updates can be manually applied via a trusted USB stick. I know this might sound perhaps a bit retro, but I think that sort of redundancy is needed. You have no idea what part of the infrastructure is going to be operational during a disaster, so this diversity of routes gives updates a chance to happen regardless.

Device hardening was also a big part of control implementation. All devices run on a stripped down Linux OS with secure boot. I disabled all unnecessary services and introduced tamper

detection via software and hardware triggers. If the case is tried to be opened or the bootloader altered, the device has the ability to shut down and alert the nearest admin on reconnect.

I also placed strong emphasis on logging and audit compliance. As the field devices can be offline, they save encrypted logs locally. When they connect back, the logs are auto forwarded to the cloud based SIEM for analysis. To avoid log tampering, I employed cryptographic signing and hash chaining. This maintains the integrity of the logs and allows follow up forensic analysis in case something goes wrong.

Finally, but certainly not least, I instituted a training and onboarding process. I created role based guides and short video modules to help users learn how to secure their devices safely. I find security training can sometimes get forgotten, but in a system like this, the user is just as critical as the hardware. If a single member of the organization does not know how to identify a phishing email or how to report a stolen device, the entire system is vulnerable.

5. Control Assessment

Once I had the controls in place, I needed to understand how effective they were in the real world. I used the evaluation procedure in NIST SP 800-53A Rev. 5 (NIST, 2020c), which promotes manual and automated testing together. “Assessment procedures were based on the methodologies provided in SP 800 53A Rev. 5” (NIST, 2020c). This is the step where things get real, the place where you find out if your assumptions were correct and where your system needs more attention.

My test approach included three components: automatic vulnerability scanning, manual control verification, and scenario testing. Under automatic scanning, I used tools such as Nessus for

scanning the cloud infrastructure and OpenSCAP for field devices. These tools enabled me to scan for missing patches, vulnerable configurations, and unauthorized services. I utilized AWS Inspector for scanning GovCloud for CIS benchmark compliance and other hardening requirements. I believe automation allowed me to get through a lot in a short while, and caught some of the smaller problems ahead of time like default SSH ports and additional packages not included in the base image.

For manual testing, I conducted a set of walkthroughs using the test procedures defined in SP 800-53A. I verified controls were implemented as documented and did spot checks in all the major families. For example, I verified certificate based authentication worked offline, devices continued to log events in an disconnected state, and revocation lists were being updated appropriately. I also did user testing interviews and had the testers walk through typical use scenarios like logging into the dashboard, beginning a secure message, or falling back to fallback communication. I'd guess that this uncovered user experience issues that technical testing would not have caught.

The most informative and third component of the evaluation was scenario testing. I designed real life events that would happen to a team during a disaster, then watched how the system reacted. One was a field device theft in a deployment. I watched how soon it could be disabled, if it still contained sensitive information, and if alarm notifications were activated. One test simulated an attacker mesh node trying to introduce malformed packets into the network. I watched how the intrusion detection system responded, and I was happy to note that it had detected and isolated the node before any damage was done.

I documented all the findings in a Plan of Action and Milestones (POA&M) document. Every finding, its severity, and suggested mitigation timeline were placed in the document. Some were low risk, such as inconsistent log timestamps. Others, such as a temporary exposure window during credential caching, were more risky and necessitated that action be taken sooner rather than later. I believe the POA&M process was particularly valuable since it kept me on my toes. Even in a good system, there are always some things that can be done better and documenting those openly is part of building trust.

This entire experience taught me that security is not a binary concept. It's not "secure" or "insecure." It's on a continuum, and analysis determines where you stand on it. I think it's intelligent, honest analyses that lend security credibility, and that's what I attempted to do with SignalShield.

6. Authorization Package

Gathering the authorization package was akin to collecting all that I had accomplished, from design to the implementation of controls to risk assessment. It is not just paperwork; it is the phase where the system is evaluated not only for how technically effective it is but whether or not it is sound and ready to be put into action in the real-world environment. I followed the framework in SP 800-37 Rev. 2 (NIST, 2020a) and constructed three primary components: the System Security Plan (SSP), the Risk Assessment Report (RAR), and the Recommendation for Authorization to Operate (ATO) memo.

The SSP was the most extensive of the documents. It covered the system architecture, the customized control set, inherited controls from the AWS GovCloud, and evaluation evidence for the controls. I included narrative rationale for all tailoring decisions and explicit implementation

details for all the relevant controls. I also pointed out the limitations and assumptions of each and every one of the controls. For example, I described why authentication needed to happen offline in the field and how it was protected using caching expiration policies and regular certificate rotation. I believe transparency in these trade-offs enables decision makers to better understand the system design context.

The RAR was all about defining, evaluating, and ranking threats. I identified threats including device loss, injection of malicious nodes, GPS spoofing, and credential leakage. I then determined the likelihood and potential effect for each of them and followed it by describing the existing mitigations. Under GPS spoofing, for example, I described how ground devices authenticate location data through signal triangulation and optionally alert users in the event of discrepancies being found. I described limitations as well, how for example, spoofing is still able to function in most satellite denied areas. I feel that the RAR thoroughly instilled the reality of the risks involved in risk management into me. No system is ever completely foolproof, and this report was all about showing how seriously I'd considered what might go wrong and how I'd address it.

Lastly, I drafted an ATO recommendation memo to the prospective Authorizing Official. I summarized the system evaluation's purpose, scope, impact categorization, and findings. I recommended authorization for three years barring high-priority findings within the POA&M list being remediated within 90 days and quarterly re evaluation on new models of the devices included within the deployment. I submitted a statement of residual risk, no system is ever 100% safe, but I did believe the SignalShield design and controls made it adequately safe to use within the operational environment in cases of disaster emergency.

I learned one thing from doing this exercise: how critical clear communication is to decision-making about cyber-security. The individuals who read the ATO memo are never going to be security experts, but still, you need them to make an informed decision. I believe it is our duty as security professionals to be upfront and honest about the facts, to explain why we believe what we do, and to establish trust through transparency. I tried to present all areas of the authorization package with that in consideration.

7. Continuous Monitoring Strategy

Continuing monitoring is probably the least admired discipline of RMF, but one I've found to be one of the most important especially for something like SignalShield. Emergencies do not happen on the calendar, and new threats emerge on an ongoing basis. Without monitoring, upkeep, and ongoing upgrade to the system, the system starts to lose sight of its objective. It is for these reasons that I made monitoring proactive and resilient and centered on actual world use cases rather than best practices in theory.

Central to the strategy is a cloud based SIEM that consumes and analyzes the logs of the field devices and the dashboard platform. Logs are stored locally on each device and in an encrypted, tamper-detection fashion. Upon the device reconnection, either through satellite, mesh, or USB relay, those logs are automatically sent. To facilitate the same, I had to compensate for time synchronization differences, network jitter, and offline disparities. I implemented hash chaining and time-stamp validation to still rely on the logs after prolonged disconnection.

Security alerts are triaged by severity, frequency, and impact. One failure to log in is probably low-priority, but failures on many devices in succession may escalate. I created SOAR playbooks to automatically react, to revoke the device credentials temporarily, alert supervisors in the field,

and to create incident tickets. I think the automation is essential, not just to act fast, but to prevent people from getting overwhelmed through the course of an ongoing event when the alerts are just starting to pile up.

Each device is assigned a risk score on the basis of the update status, alert history, and behavioral patterns. For example, devices which fail to report in three or higher are marked as "needs attention" and are also shown on the dashboard. This reduces the effort to triage action items without having to go through the many log entries. Also, this type of risk-based triage allows sparse staff to handle large-scale distributed structures with greater ease. “The monitoring plan also aligns with guidance from DHS’s Continuous Diagnostics and Mitigation (CDM) Program” (U.S. Department of Homeland Security, 2021).

I also set the cadence for regular review of the controls. The threat intelligence is reviewed by the team on every 90 days and the set of controls gets updated accordingly. We twice yearly conduct an overall check on device images, baseline configurations, and admin privileges. I set an ongoing policy wherein after every real incident or deployment, the system is scanned through a post-incident analysis to determine the visibility or response time gaps.

To keep the strategy current, I also added tracking of user behavior. I don't necessarily mean observing what the users are doing, I mean monitoring for trends which suggest friction or confusion. For instance, if users are typing credentials incorrectly repeatedly, that is likely a UI issue and not so much an abuse of policy. I find monitoring is best when you are utilizing it to improve the user experience as much as identify bad actors.

Overall, I believe ongoing monitoring has to be an extension of the system function, not an afterthought. For SignalShield, the function is being there when people need it most. Monitoring guarantees readiness day in and day out, whether anyone is watching or not.

8. Lessons Learned and Recommendations

I anticipated to be using RMF primarily on an imaginary system and check-marking the steps in between. What actually occurred was much bigger than that, I became able to use RMF not just to handle risk, but to actually design a system to fit the people who use it and the mission being supported. I'd say the greatest thing I gained is that RMF is not about filling out paperwork, but about creating safe systems that function when it counts.

Of all the things I learned, one of the most useful was the interplay between usability and security. These are frequently presented as contradictory objectives in most situations, but I don't think that necessarily needs to be so. SignalShield is going to be put into the hands of users already under extreme pressure, firefighters, paramedics, dispatchers and the system must work around them, not impede or confuse them, or else it is failure regardless of how secure it is. That is why I concentrated on adaptable access controls, on making authentication function offline, and on having updates applied in more than one manner. Security must be robust, but simultaneously it must be unobtrusive when it can be and comprehensible when it cannot be.

Another key takeaway was the art of tailoring. I was resistant to tweaking or cutting anything on the high-impact baseline of SP 800-53 at first. But the more I worked within the system, the more I realized that not all controls apply to all environments, and that is the way it is meant to be. The art is to understand what the intent of the control is and how to implement the intent in

your specific situation. I believe that tailoring is no short cut but an art. It is how you get RMF flexible enough to address the real function of a system.

I also learned much about communication. Whether writing the ATO memo, crafting the SSP, or talking about risk mitigation techniques in the RAR, I needed to think about how to communicate accurately and concisely. The people approving or signing off on the system are not going to be cybersecurity experts, and do not need to be. My role as architect is to translate the technical risk into operational terms. I believe that if you communicate truthfully, you establish trust, and on emergency systems, trust is paramount.

If I were to give one piece of advice to someone doing something similar, I would tell them to start with the mission and not the controls. Think about what the system has to do, who is going to use it, and what failure constitutes. Once you understand that, the rest of the RMF process starts to become much clearer. I'd also tell them not to be afraid to think outside the box. RMF offers you the template, but where the real benefit is in taking the template and using it on real-world problems.

Conclusion

SignalShield, of course, is fictional, but I did not design it as such. I designed it as though it were actually going to be deployed in an actual emergency by actual people with actual lives at stake. That was the mindset behind every decision I made, from the design of the controls to how I instructed the operators, and I believe it's what made the project feel real to me.

Operating with the Risk Management Framework using this system enlightened me to the fact that RMF is more than a process, but a way of thought. It's how you handle risks, how you weigh

priorities against each other, and how you build security into a system from the outset—not as an add-on, but as a foundation. I now view RMF as a dynamic and mission-driven thing, not as inflexible or bureaucratic. And I believe that mindset is one I'll carry with me long after the project has been completed.

I left with an ever-stronger belief that cybersecurity needs to be human. It cannot be just locks and firewalls. It has to consider how the human does think and act under stress, and what it is going to take to let them do their work securely. For something like SignalShield, that is about building trust, reducing friction, and always expecting the unexpected. I'd say good cybersecurity not just secures the systems, but it enables the humans who are operating them.

This project has stretched me in ways I never expected and for which I'm grateful. It stretched me to be wiser, more perceptive, and more attuned to the larger landscape. I think that if SignalShield were ever to become reality, then it would serve its purpose, and I hope to do the same in my commitment to do the right thing, no matter the obstacles, no matter what is at stake.

References

Amazon Web Services. (n.d.). AWS GovCloud (US). https://aws.amazon.com/govcloud-us/

National Institute of Standards and Technology. (2004). Standards for security categorization of federal information and information systems (FIPS 199). https://doi.org/10.6028/NIST.FIPS.199

National Institute of Standards and Technology. (2008). Guide for mapping types of information and information systems to security categories (SP 800-60 Vol. 2). https://doi.org/10.6028/NIST.SP.800-60v2r1

National Institute of Standards and Technology. (2020a). Risk management framework for information systems and organizations (SP 800-37 Rev. 2). https://doi.org/10.6028/NIST.SP.800-37r2

National Institute of Standards and Technology. (2020b). Security and privacy controls for information systems and organizations (SP 800-53 Rev. 5). https://doi.org/10.6028/NIST.SP.800-53r5

National Institute of Standards and Technology. (2020c). Assessing security and privacy controls in information systems and organizations (SP 800-53A Rev. 5). https://doi.org/10.6028/NIST.SP.800-53Ar5

U.S. Department of Homeland Security. (2021). Continuous Diagnostics and Mitigation (CDM) Program Overview. https://www.cisa.gov