Network-Attack-Resilient Intrusion-Tolerant SCADA for the Power Grid

Amy Babay*, Thomas Tantillo*, Trevor Aron, Marco Platania, and Yair Amir
Johns Hopkins University — {babay, tantillo, taron1, yairamir}@cs.jhu.edu
AT&T Labs — {platania}@research.att.com
Spread Concepts LLC — {yairamir}@spreadconcepts.com

***Abstract*—As key components of the power grid infrastructure, Supervisory Control and Data Acquisition (SCADA) systems are likely to be targeted by nation-state-level attackers willing to invest considerable resources to disrupt the power grid. We present Spire, the first intrusion-tolerant SCADA system that is resilient to both system-level compromises and sophisticated network-level attacks and compromises. We develop a novel architecture that distributes the SCADA system management across three or more active sites to ensure continuous availability in the presence of simultaneous intrusions and network attacks. A wide-area deployment of Spire, using two control centers and two data centers spanning 250 miles, delivered nearly 99.999% of all SCADA updates initiated over a 30-hour period within 100ms. This demonstrates that Spire can meet the latency requirements of SCADA for the power grid.**

I. Introduction

Supervisory Control and Data Acquisition (SCADA) systems form the monitoring and control backbone of the power grid. SCADA systems allow power grid operators to monitor the status of the grid, detect abnormal conditions, and issue control commands to manage the physical equipment in the power substations. It is critical to ensure that SCADA systems are continuously available and operating correctly: failures and downtime can have severe consequences, including equipment damage and extended blackouts.

As key components of critical infrastructure, SCADA systems are likely to be targeted by nation-state-level attackers willing to invest considerable resources to disrupt the power grid. Moreover, as SCADA systems move to use IP networks to take advantage of their cost benefits and implement smart-grid capabilities, the traditional assumptions that these systems are air-gapped and inaccessible to outside attackers no longer hold. Recent reports show that SCADA systems are increasingly subject to attack [1].

While today's SCADA systems employ fault tolerance to maintain operation when parts of the system fail, they were never designed to withstand malicious attacks. As shown in Figure 1, state-of-the-art SCADA systems typically use primary-backup approaches to provide disaster recovery capabilities. Specifically, a hot backup of the central control server (the SCADA master) can take over immediately if the primary SCADA master fails, and in many SCADA systems, a cold-backup control center can be activated within a couple of hours if the primary control center fails. The SCADA master is responsible for collecting and logging data from *Remote Terminal Units* (RTUs) and *Programmable Logic Controllers* (PLCs), presenting the current status of the infrastructure to a human operator via the *Human-Machine Interface* (HMI), and issuing control commands to the RTUs and PLCs. The RTUs and PLCs connect to the physical equipment in the power substations to translate signals (e.g. current, phase, voltage) into digital data, send status updates to the control center via a wide-area network, and control the physical devices based on supervisory commands from the SCADA master. To provide real-time monitoring and control capabilities, SCADA systems for the power grid must deliver device status updates and supervisory commands within 100-200ms [2], [3].

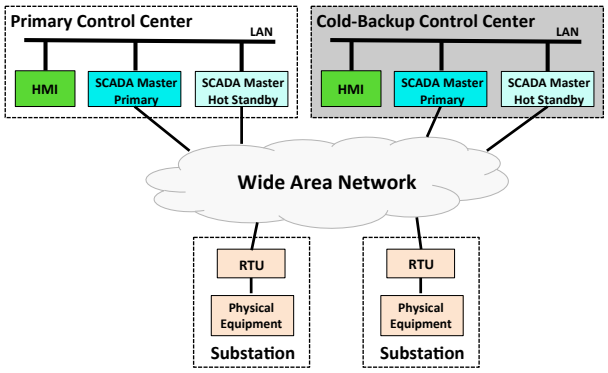

Fig. 1. Modern SCADA architecture using two control centers.

While the current primary-backup architectures provide sufficient resilience to overcome benign failures, they are not adequate to cope with the hostile environments that SCADA systems are now being exposed to. In these environments, SCADA systems will need to overcome both network-level and system-level attacks.

For example, a sophisticated network attack can take the primary control center offline at the time of the attacker's choosing, disrupting its ability to communicate with the field substations and incurring system downtime. Such network attacks cannot be overcome using a primary-backup architecture. A cold-backup approach inherently incurs downtime to bring the backup online. When a control center fails as the result of a benign problem, the downtime incurred while activating the backup is likely to occur at a non-critical time, and therefore is considered acceptable today; however, a malicious attack can be intentionally launched at the worst possible time (e.g. during a major snowstorm or during a coordinated large-scale attack in multiple domains). A hot-backup approach (where the backup control center is always active and ready to take

* Equal Contribution

over) is subject to a "split-brain" problem: if the primary and backup cannot communicate (either due to benign network failures or malicious network attacks), they will both attempt to assume the role of the primary and can issue conflicting control commands, leading to incorrect behavior.

In addition, system-level compromises of the SCADA servers can have devastating consequences. A compromised SCADA master can issue malicious commands to damage physical power grid components and can manipulate monitoring information to prevent operators from correcting or even being able to observe the problem.

We present Spire, the first intrusion-tolerant SCADA system that simultaneously withstands attacks and compromises at both the system level and the network level. To overcome system-level compromises of the SCADA masters, we build on existing work on intrusion-tolerant replication, combined with proactive recovery and diversity, to enable the system to continue to work correctly as long as no more than a certain fraction of the SCADA master replicas are compromised.

However, none of the existing work on intrusion-tolerant replication is resilient to the network attacks we consider. Our recent experience with a red-team attack of the Spire system shows that the network is commonly the first target for attacks: if the system can be disabled by disrupting the communication between its components, there is no need for domain-specific attacks that employ specialized knowledge of the power grid.

We demonstrate that the two-control-center architectures used by power companies today are not sufficient to provide resilience to network attacks: at least three active sites are required. We develop a novel architecture that distributes SCADA master replicas across three or more sites to ensure continuous availability in the presence of simultaneous system intrusions and network attacks. Even if an attacker is able to target and isolate a control center from the rest of the network (as sophisticated network attacks are capable of [4], [5]), the system will continue to operate correctly, as long as the number of compromises in the rest of the system does not exceed the tolerated threshold.

To make our architecture viable for deployment, it must fit the current power company model that budgets for and deploys no more than two control centers that can control physical devices in the substations. Our novel architecture allows the two control centers to be augmented with one or more commodity data centers that do not need to control field devices, providing the same resilience at a feasible cost.

A wide-area deployment of Spire, using two control centers and two data centers, spanning 250 miles (similar to large US power grids), delivered nearly 99.999% of all SCADA updates initiated over a 30-hour period within 100ms. Out of 1.08 million updates, only 13 took over 100ms, and only one of those 13 exceeded 200ms. This demonstrates that Spire can meet the latency requirements of SCADA for the power grid.

The primary contributions of this work are:

- We invent the first intrusion-tolerant SCADA system that simultaneously addresses system compromises and network attacks. To support this threat model, we develop a novel architecture that distributes SCADA master replicas across the required three or more active geographic sites.
- We extend the architecture to leverage commodity data centers (that may not be able to control field devices) to avoid constructing additional power company control centers, reducing costs and making the architecture viable for deployment.
- We deploy and evaluate Spire on a wide-area network with a geographic footprint similar to that of large U.S. power grids. We show that the system can meet the stringent latency requirements of the power grid.

II. Spire Approach Overview

We introduce a new SCADA system architecture that is resilient to simultaneous system compromises and sophisticated network attacks. At the system level, we use a version of the Prime intrusion-tolerant replication engine [6], [7] to overcome compromises of the SCADA master. At the network level, we combine the Spines intrusion-tolerant network [8], [9] with a novel architecture for distributing replicas across multiple geographic sites, such that even if one site is disconnected from the rest of the network, the system is able to continue operating correctly. Our solution protects the system over a long lifetime using proactive recovery and diversity, and provides strict latency and reliability guarantees that meet the demands of SCADA systems for the power grid.

A. Intrusion-Tolerant Replication

Spire uses intrusion-tolerant replication to overcome compromises of the SCADA master. Intrusion-tolerant replication ensures that each correct replica maintains an identical copy of the system state, even when up to a threshold number f of the replicas are compromised and can exhibit Byzantine [10] (arbitrary) behavior. Intrusion-tolerant replication protocols can overcome up to f compromised replicas by using $3f + 1$ total replicas [11].

While all intrusion-tolerant replication protocols guarantee safety (consistency) and liveness (each valid update is eventually executed), only a subset of protocols guarantee performance under attack (e.g. [7], [12], [13], [14], [15]). Spire uses a version of the Prime intrusion-tolerant replication engine [7] because it provides strong latency guarantees for each update. Specifically, Prime guarantees that every update is executed within a bounded delay after it is introduced, making it an excellent fit for the stringent latency requirements of SCADA systems for the power grid. Note, however, that Spire could use any intrusion-tolerant replication protocol that provides the necessary performance (timeliness) guarantees.

B. Diversity

Intrusion-tolerant replication protocols only guarantee correctness as long as the number of compromised replicas does not exceed the tolerated threshold f. However, if all replicas in the system are identical copies of one another, an attacker who successfully exploits one replica can simply reuse the same exploit to compromise all of the replicas in the system.

To prevent an attacker from gaining control of more than f replicas, the system must ensure that the replicas present diverse attack surfaces. Diversity can be achieved using approaches such as N-version programming [16], [17], operating system diversity [18], or software diversification at compilation or run time [19], [20], [21]. Spire uses the MultiCompiler [22], which employs techniques such as stack padding, no-op insertion, equivalent instruction substitution, and function reordering to diversify the code layout of an application. The MultiCompiler uses a 64-bit random seed to generate diversity from a large entropy space, making it unlikely that the same attack on the codebase will successfully compromise any two distinct variants.

C. Proactive Recovery

Even if replicas are sufficiently diverse, given enough time, a dedicated attacker will eventually be able to craft enough distinct attacks to compromise more than f replicas. Therefore, it is necessary to use proactive recovery to ensure survivability over the lifetime of the system (years for SCADA) [11], [23].

In proactive recovery, each replica is periodically brought down and restarted from a known clean state (removing any compromises) with a new diverse variant of the software (and potentially of the entire operating system) that is with high probability different from all past and future variants. This makes the job of the attacker significantly harder, as they now must simultaneously compromise more than f replicas within a limited time window. To maintain availability in the presence of both f intrusions and k simultaneous proactive recoveries, a system with $3f + 1$ replicas (e.g. Prime) must be extended to use $3f + 2k + 1$ total replicas [24].

D. Intrusion-Tolerant Network

While intrusion-tolerant replication (with diversity and proactive recovery) ensures correct operation despite SCADA-master compromises, it does not provide resilience to network attacks. If an attacker disrupts the communication between the control center and the power substations, the SCADA system loses its ability to monitor and control the power grid, even if all the SCADA masters are working correctly. Previous work provides timeliness and quality-of-service guarantees for SCADA networks [25], [26], but these solutions are not resilient to attacks. As we discuss in Section VII, targeting the network is a common strategy for attackers, as it does not require protocol- or domain-specific knowledge. Therefore, a resilient networking foundation is essential for a complete intrusion-tolerant SCADA solution.

Spire uses the Spines overlay messaging framework [8], which provides the ability to deploy an intrusion-tolerant network [9]. Spines uses an overlay approach to overcome attacks and compromises in the underlying network: overlay sites are connected with redundancy, forcing an attacker to successfully attack many links in the underlying networks to disrupt communication to a single site. By using multihoming at each site, Spines can leverage multiple underlying networks (e.g., ISP backbones) to tolerate the complete failure of one or more underlying networks. To overcome compromises of the overlay nodes, intrusion-tolerant protocols authenticate all traffic, employ redundant dissemination, and enforce fairness [9].

E. Remaining Challenges

The intrusion-tolerant network protects against large-scale network disruption, overcomes malicious routing attacks, and substantially increases the effort and resources required to launch a successful denial-of-service attack. However, because SCADA systems are high-value targets, it is likely that dedicated nation-state-level attackers will invest considerable resources to disrupt these systems. With enough resources, it is possible to execute sophisticated denial-of-service attacks that can target a specific site and isolate it from the rest of the network, such as the Coremelt [4] and Crossfire [5] attacks. However, we believe that it is nearly infeasible for an attacker to create the complete simultaneous meltdown of multiple ISP backbones necessary to target and disconnect multiple sites when using the intrusion-tolerant network. As a result, we consider a threat model that includes one disconnected site, in addition to compromises or other failures of the SCADA master replicas. To overcome these sophisticated network attacks, we develop a novel framework for distributing replicas across multiple sites, which we describe in Section IV.

III. System and Threat Model

Our full threat model is very broad, requiring only weak assumptions and has never been considered before. This threat model includes system-level compromises as well as network-level threats. At the system level, we consider compromised (Byzantine) SCADA master replicas that are completely under the control of the attacker and may exhibit arbitrary behavior.

At the network level, we consider network link failures, misconfigurations, and malicious network attacks, including (but not limited to) routing attacks (e.g. BGP hijacking [27]) and sophisticated denial-of-service attacks (e.g. Coremelt [4] and Crossfire [5]) that can isolate a targeted site. As explained in Section II-E, Spire's intrusion-tolerant networking foundation addresses this broad network threat model and allows us to reduce it to a narrower model that assumes only a single site can be disconnected. Hence, in the remainder of the paper, we develop an architecture that simultaneously tolerates the compromise of up to f SCADA master replicas, the disconnection of one system site (possibly a control center), and the unavailability of one replica due to proactive recovery.

Note that our threat model does not cover a compromised HMI or a rogue operator. Such a compromise can have direct system-wide effects and will need to be handled through other mechanisms. Similarly, the threat model does not cover a compromised RTU or PLC. Such a compromise can have direct local effects on the power grid components controlled by that device and may have wider indirect effects. However, Spire's architecture provides protection for all system endpoints, including HMIs, RTUs, and PLCs, making them considerably more difficult to compromise (see Section V).

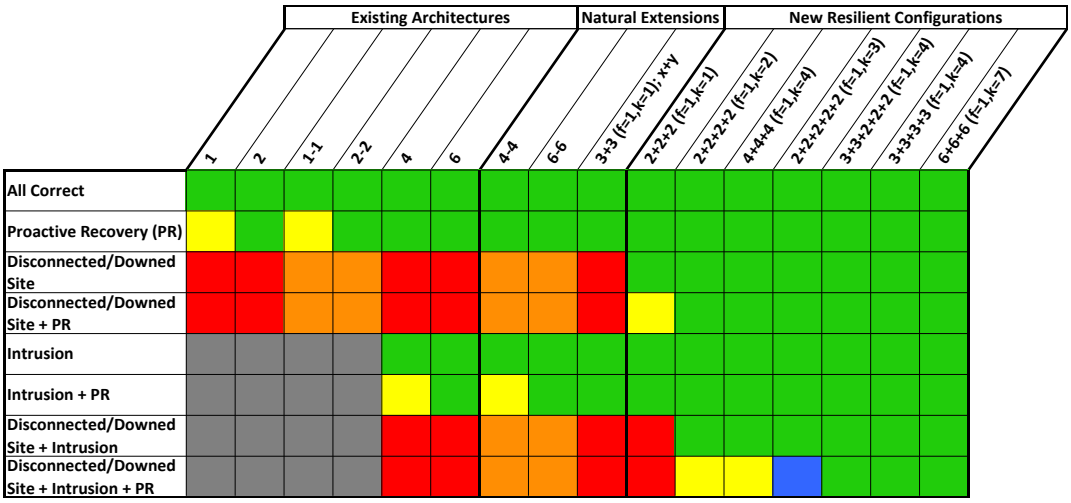

Fig. 2. Illustration of specific SCADA system configurations' ability to support the threat model we consider, including all combinations of a replica being unavailable due to proactive recovery, a site disconnection due to network attack or failure, and an intrusion (SCADA master compromise).

As long as no more than f SCADA master replicas are simultaneously compromised, Spire guarantees consistency of the system state. Specifically, it guarantees *safety*: if two correct replicas execute the i^{th} update, then those updates are identical. As discussed in Section II-C, proactive recovery forces an attacker to compromise more than f replicas within a confined time window (rather than over the entire lifetime of the system) to succeed in violating the system guarantees.

In terms of performance, Spire guarantees *bounded delay*: the latency for an update introduced by an authorized system component to be executed by a correct SCADA master replica is upper bounded. At any time, we assume that at most one site may be disconnected from the rest of the network. To provide bounded delay, Spire requires that all of the correct SCADA master replicas, with the exception of those in the disconnected site, are able to communicate with one another. Moreover, at least one correct SCADA master replica in a control center must be able to communicate with field substations.

Note that due to the network stability requirements of Prime, communication must also meet the *bounded-variance* property of [7], which requires that for each pair of correct servers, the network latency does not vary by more than a factor K_{Lat}. However, since we consider the bounded amount of time required to view change to a correct leader as part of the bounded delay to execute an update, in practice we only require that the latency variation does not exceed K_{Lat} over the short time period required to complete the view change and execute an update in the new view. A fuller discussion of bounded delay across view changes appears in Section V-D.

IV. Network-Attack Resilient Intrusion-Tolerant SCADA Architecture

To develop a network-attack-resilient intrusion-tolerant SCADA architecture that supports the broad threat model we consider, we first analyze existing SCADA architectures (Section IV-B) and their natural extensions (Section IV-C), showing that none completely addresses this threat model. Based on this analysis, we develop a novel architecture that provides continuous system availability under our model. We discuss specific example configurations (Section IV-D), as well as a general framework for network-attack-resilient intrusion-tolerant SCADA architectures (Section IV-E).

A. Analysis Framework

Figure 2 presents example SCADA system configurations and shows each configuration's ability to support the threat model we consider. Each row corresponds to a failure/attack scenario we aim to address. Each column corresponds to a specific SCADA system configuration. The name of each configuration describes how the SCADA master replicas are distributed: a configuration "x" indicates a single control center containing x replicas, "x-y" indicates a primary-backup architecture with x replicas in the primary control center and y replicas in the backup, and "x+y+..." indicates active intrusion-tolerant replication across multiple sites, with x replicas in the first control center, y replicas in a second control center, and so on. Each configuration shown in Figure 2 is discussed in Section IV-B, IV-C, or IV-D.

A **green** cell in Figure 2 represents a fully operational system with performance guarantees under attack. In this case, the system is guaranteed to process any update within the bounded amount of time necessary to support SCADA systems for the power grid (about 100-200ms).

A **gray** cell indicates that the system is not guaranteed to remain safe: an intrusion can compromise the system state.

A **red** cell indicates that the system will remain safe but will not provide any guarantee of progress: progress halts until a network attack ends or a failed site is repaired.

An **orange** cell indicates that the system will remain safe, but will not provide any guarantee of progress until an operator activates a cold-backup control center, which can take a significant amount of time (tens of minutes to hours).

A **yellow** cell is similar to a green cell, except that the performance guarantee is not met when a correct replica

is undergoing proactive recovery. Progress with performance guarantees resumes once the recovery is completed.

The one **blue** cell is similar to a green cell, except that the performance guarantee is not met in a very specific case, where one of the two control centers is disconnected, there is an intrusion in the other control center, and the remaining correct server in that control center is currently undergoing proactive recovery. Once the recovery of that specific server is completed, the performance guarantees will be met again.

B. Existing SCADA Architectures

Figure 2 shows that currently deployed SCADA systems (first four columns) are not sufficient to support the threat model we consider: they cannot even guarantee safety. Configuration "2-2" corresponds to the state-of-the-art SCADA architecture discussed in Section I, where a hot-backup SCADA master takes over if the primary SCADA master fails, and a cold-backup control center can be brought online if the primary control center fails. While configuration "2-2" improves on simpler systems that do not use a hot backup ("1" and "1-1") and on systems that only use a single control center ("1" and "2"), any intrusion can have devastating consequences, violating safety guarantees and causing incorrect behavior. In addition, if the primary control center fails or is disconnected, no progress can be made until the backup is brought online.

Initial efforts to create intrusion-tolerant SCADA used intrusion-tolerant replication within a single control center, using $3f+1$ replicas (4 for $f=1$) to tolerate f intrusions or $3f+2k+1$ replicas (6 for $f=1, k=1$) to simultaneously tolerate f intrusions and k proactive recoveries. As Figure 2 shows, these configurations ("4" and "6") overcome intrusions and maintain safety in all cases (with the "6" also tolerating a proactive recovery), but they cannot tolerate a control center going down or becoming disconnected due to a network attack.

C. Natural Extensions of Existing Architectures

To get the benefits of both existing fault-tolerant SCADA architectures ("2-2") and intrusion-tolerant replication ("4" or "6"), we can combine the two approaches. We can deploy intrusion-tolerant replication with four or six replicas in the primary control center, and if the primary control center fails, we can activate a backup control center with its own self-contained intrusion-tolerant replication deployment (configurations "4-4" and "6-6"). Figure 3 shows configuration "6-6".

This natural extension improves on the previous configurations by making it possible to both tolerate an intrusion and restore operation if a control center is downed or disconnected. However, restoring operation using the backup control center can take a significant amount time (tens of minutes to hours). In a malicious setting, an attacker can launch a network attack to take down the primary control center at the time of their choosing, potentially causing considerable downtime at a critical moment. Furthermore, the attacker can repeatedly launch the same attack, causing downtime to occur frequently.

Recall from Section I that switching from a cold-backup approach to a hot-backup approach, where the backup control center is always active and ready to take over, does not solve the problem: network partitions (due to either benign failures or malicious attacks) can cause a "split-brain" problem in which both control centers believe they are the primary.

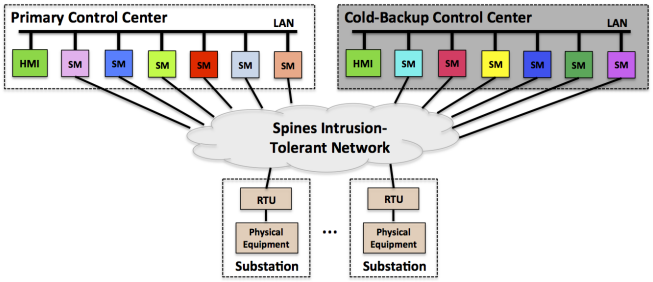

Fig. 3. SCADA Architecture with 6 replicas in primary control center and 6 replicas in cold-backup control center (configuration 6-6).

To avoid the potentially attacker-driven downtime incurred by using a primary-backup approach, we instead use active replication across multiple sites. An initial approach that fits current SCADA architectures using two control centers is to split the six replicas of configuration "6" between two control centers, with all replicas active and running the intrusion-tolerant replication protocol (configuration "3+3").

Unfortunately, splitting the replicas across two control centers does not provide any additional resilience. In fact, this is true regardless of the total number of replicas or their distribution: for any configuration "$x+y$", one of the two control centers must have at least half of the total replicas. If that control center is unavailable, the intrusion-tolerant replication protocol cannot make progress. Specifically, progress requires at least $2f+k+1$ connected correct replicas, which is more than half of the $3f+2k+1$ total replicas.

D. Intrusion-Tolerant SCADA Resilient to Network Attacks

The above analysis of configuration "$x+y$" leads to the key insight that more than two sites are necessary to ensure continuous availability during a network attack that can disconnect a control center. However, it is generally not feasible for power companies to construct additional control centers with full capabilities for controlling RTUs and PLCs in the field due to the high cost of equipment and personnel.

One of the main innovations of this work is the realization that power companies can use additional sites that do not communicate with RTUs or PLCs to deploy an effective and practical solution. These sites can be implemented relatively cheaply using commercial commodity data centers. The data centers connect with the control centers to participate in the intrusion-tolerant replication protocol, but do not communicate with field substations. For configurations with more than two sites in Figure 2, the first two sites are control centers and the remaining sites are data centers.

Knowing that we need more than two sites, we can try to distribute the six replicas needed to tolerate one intrusion and one proactive recovery across three sites (configuration "2+2+2", which is illustrated in Figure 4). Similarly to configuration "6", configuration "2+2+2" successfully provides

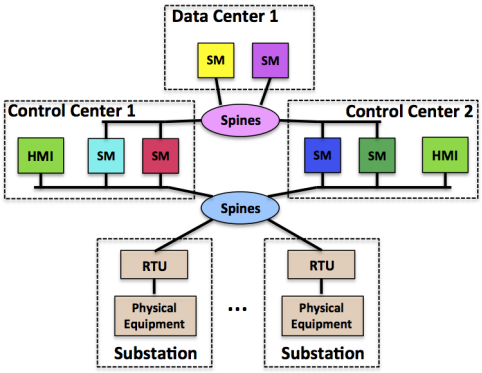

Fig. 4. SCADA Architecture with 2 replicas in each of the two control centers and the single data center (configuration 2+2+2).

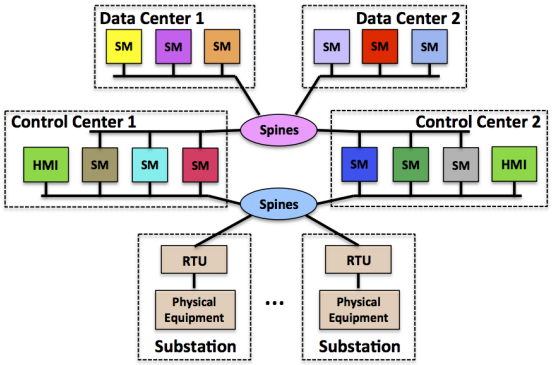

Fig. 5. SCADA Architecture with 3 replicas in each of the two control centers and two data centers (configuration 3+3+3+3).

bounded delay in the presence of one intrusion and one proactive recovery. Moreover, this configuration can now provide bounded delay with a failed or disconnected site. However, if any other issue occurs while a site is down or disconnected, configuration "2+2+2" cannot make progress. In this case, the protocol requires four ($2f + k + 1 = 4$) correct connected replicas to make progress. The disconnection of a site leaves exactly four correct replicas connected, meaning that no additional issues can simultaneously be tolerated.

To simultaneously support a downed or disconnected site and another issue, we can increase the parameter k in the $3f + 2k + 1$ formula. If we set k to the number of replicas in the largest site, the system can provide bounded delay in all cases except when all three issues occur simultaneously: a site is disconnected, a replica is compromised, and a replica is undergoing proactive recovery. Configurations "2+2+2+2" and "4+4+4" provide these availability guarantees. Note that "2+2+2+2" and "4+4+4" are the cheapest configurations (in terms of number of replicas) able to provide these guarantees for four sites and three sites, respectively.

To support the full threat model, maintaining availability even when all issues occur simultaneously (a failed or disconnected site, an intrusion, and a proactive recovery), we can again increase k. If we set k to the number of replicas in the largest site, plus the maximum number of simultaneous proactive recoveries (in this case, one), we can ensure that $2f + k + 1$ correct replicas are connected at all times. This allows the system to provide bounded delay in all cases.

In the case that the largest site contains two replicas, this means that k must be three, so overcoming one intrusion will require $3f + 2k + 1 = 10$ replicas (for $k = 3$, $f = 1$), resulting in configuration "2+2+2+2+2". However, in our SCADA architecture not all replicas are equal. To make the intrusion-tolerant architecture feasible for utility companies to deploy, it only includes two control centers (with the other sites located in commodity data centers), and only replicas in control centers can communicate with field devices. Therefore, our SCADA architecture requires not only that $2f + k + 1$ correct replicas be connected, but also that at least one of those replicas is located in a control center. Configuration "2+2+2+2+2" shows exactly this point. The system provides bounded delay at all times except in the specific case that one control center has failed or been disconnected, there is an intrusion in the other control center, and the remaining replica in that control center is currently undergoing proactive recovery. In that narrow case, progress stops until that particular replica completes its recovery.

Building a third control center will eliminate this issue, but such a solution is not practical in SCADA environments for the foreseeable future. Instead, we can increase the number of replicas to ensure that a correct control center replica is always available under our threat model. Configuration "3+3+2+2+2" adds one replica to each control center and provides bounded delay in the simultaneous presence of an intrusion, proactive recovery, and a failed or disconnected control center.

Configurations "3+3+2+2+2", "3+3+3+3" (illustrated in Figure 5), and "6+6+6" are the first to demonstrate a complete solution that supports the threat model we consider and is viable for power companies to deploy. Using only two control centers, these configurations provide bounded delay even in the simultaneous presence of an intrusion, a failed or disconnected site, and an ongoing proactive recovery. Each of these three configurations uses the minimal number of replicas required to support these guarantees using two control centers and three, two, or one data centers, respectively.

Of the three configurations providing a complete solution, configuration "3+3+3+3" appears to strike the best balance between the number of sites used and the total number of replicas required (and corresponding processing and messaging intensity): configuration "3+3+2+2+2" requires the same number of replicas but uses one additional data center, making it strictly more expensive; configuration "6+6+6" uses one fewer data center, but requires 18 replicas compared with 12. Due to the all-to-all nature of communication in the intrusion-tolerant replication protocol, this makes it considerably more expensive in terms of messaging and processing.

E. General Framework for Network-Attack-Resilient Intrusion-Tolerant SCADA

We can generalize the above examples to design new intrusion-tolerant SCADA system configurations that can use any number of sites S (where $S > 2$) to tolerate any number

	2 control centers + 1 data center	2 control centers + 2 data centers	2 control centers + 3 data centers
$f = 1$	6+6+6	3+3+3+3	3+3+2+2+2
$f = 2$	9+9+9	5+5+5+4	4+4+3+3+3
$f = 3$	12+12+12	6+6+6+6	5+5+4+4+4

TABLE I
SCADA SYSTEM CONFIGURATIONS TOLERATING A PROACTIVE RECOVERY, DISCONNECTED SITE, AND 1, 2, OR 3 INTRUSIONS

of intrusions f, while simultaneously supporting one disconnected site and one replica undergoing proactive recovery.

As stated in Section II-A, the minimal number of replicas needed to tolerate f simultaneous intrusions and k proactively recovering replicas is $n = 3f+2k+1$. As shown in the above discussion of example configurations, the k parameter can be extended to include all non-Byzantine faults in the system. Since our threat model includes an entire site being down or disconnected (potentially due to a network attack), as well as one proactively recovering replica at any given time, k must be at least the number of replicas in the largest site (to account for the disconnection of that site) plus one (to account for the recovering replica). That is, for n replicas evenly distributed across S sites, we require: $k \geq \lceil \frac{n}{S} \rceil + 1 = \left\lceil \frac{3f+2k+1}{S} \right\rceil + 1$. We can show that this requirement is met as long as we have $k \geq \left\lceil \frac{3f+S+1}{S-2} \right\rceil$ (for details see [28]).

After finding the minimal value of k using this formula, the total number of required replicas can simply be calculated from the original formula $n = 3f + 2k + 1$.

For example, to overcome 1 intrusion using 4 total sites ($f = 1$, $S = 4$), this approach gives us $k \geq \left\lceil \frac{3(1)+4+1}{2} \right\rceil = 4$ and $n = 3(1) + 2(4) + 1 = 12$. Distributing these 12 replicas evenly across the 4 sites gives us exactly configuration "3+3+3+3" discussed in Section IV-D.

However, this formula does not account for the constraint discussed in Section IV-D that it is not feasible for power grid operators to construct more than two control centers with full capabilities for controlling field devices. To fix this, we must ensure that each control center has at least $f + 2$ replicas, so that even if one control center is disconnected and the other contains f compromised replicas and one proactively recovering replica, there is still one correct replica that can control the field devices. Since k must be at least one more than the size of the largest site, this means we must have $k \geq f+3$ in all cases. Hence, we adjust our formula for k to:

$$k = \max\left(f + 3, \left\lceil \frac{3f + S + 1}{S - 2} \right\rceil\right)$$

As before, after obtaining a value for k, we calculate the total number of required replicas, based on the requirement $n \geq 3f + 2k + 1$. To distribute the replicas among the sites, $f + 2$ replicas must first be placed in each control center. The remaining replicas must then be distributed such that no single site has more than $k - 1$ replicas, which can be achieved by distributing replicas as evenly as possible across the sites.

Table I presents the minimal number of replicas required to tolerate one, two, or three intrusions while simultaneously supporting a single proactive recovery and a single disconnected site with two control centers and one, two, or three data centers (for a total of three, four, or five sites). In the table, the first two numbers in each cell represent the number of replicas in each of the two control centers, while the remaining numbers represent the number of replicas in each data center.

As Table I shows, configurations with more sites require fewer total replicas, as losing any one site has less impact on the system. This presents a trade-off between the cost of additional sites compared with the cost of additional replicas and their associated processing and messaging increase (due to the all-to-all communication pattern). Configurations using two data centers provide a good balance between these factors.

V. INTRUSION-TOLERANT SCADA SOFTWARE IMPLEMENTATION

The architecture described in Section IV provides a framework for deploying SCADA systems that can overcome both compromises and network attacks. However, this framework requires that all SCADA system components support intrusion tolerance: the SCADA master must be replicated using intrusion-tolerant replication, and the HMI, RTUs, and PLCs must correctly interact with the replicated SCADA master.

Existing SCADA systems were not designed to support intrusion tolerance. Previous work that added intrusion tolerance to an existing SCADA product [29], as well as our initial efforts to add intrusion tolerance to an existing open-source SCADA system, observed important mismatches between the models and performance needs of SCADA systems and those provided by existing intrusion-tolerant technologies. These mismatches made the resulting prototypes complex, difficult to extend, and limited in scalability.

Therefore, Spire is designed from the ground up, with intrusion tolerance as a core design principle: it includes a SCADA master designed from scratch to support intrusion-tolerant replication, RTU/PLC proxies that allow the SCADA master to interact with RTUs and PLCs in an event-driven intrusion-tolerant manner, and an intrusion-tolerant communication library. Spire builds on proven open-source components, using the Prime intrusion-tolerant replication engine [6], a pvbrowser-based HMI [30], the pvbrowser and OpenDNP3 [31] implementations of the Modbus and DNP3 communication protocols (used between RTUs/PLCs and our proxies), and the Spines intrusion-tolerant network [8].

Spire's architecture protects RTUs and PLCs by limiting their insecure communication protocols (e.g. Modbus and DNP3) to only the connection between an RTU or PLC and its proxy. To provide the strongest protection, this connection can be a physical wire (as opposed to a network). Communication between the RTU/PLC proxy and the rest of the system occurs over the secure and intrusion-tolerant Spines network. Note, however, that if an attacker is able to compromise an RTU/PLC proxy, they may be able to use that position to compromise or send incorrect commands to an RTU or PLC, which our threat model does not cover (as noted in Section III). Similarly, Spire's HMI communicates only through Spines and does not accept out-of-band communication.

A. Scalable Event-Driven Architecture

There is a major discrepancy between the server-driven polling model of conventional SCADA systems and the client-driven update model of intrusion-tolerant replication engines. While previous work compensated for this by using complex logical timeout protocols [29], we re-design the SCADA master, offloading its polling functionality to RTU/PLC proxies.

Ideally, an RTU/PLC proxy is placed in each field site (e.g. power substation) and is responsible for polling the RTUs and PLCs in that site. However, if this is not possible, the proxies may be placed anywhere between the SCADA master and the field sites, including in the control center. In fact, for the foreseeable future, many substations are likely to use non-IP communication and will need to communicate via proxies located in the control centers. When the proxy detects a change in the state collected from the RTUs and PLCs, it sends an update to the replicated SCADA master, which is ordered and executed using the intrusion-tolerant replication engine. The proxies also send periodic status updates to the SCADA master even if no change has occurred, but this interval may be relatively long (e.g. on the order of a second or more).

This event-driven approach allows the system to scale to many RTUs and PLCs, as the intrusion-tolerant replication engine does not need to process every poll (which may occur frequently, e.g. every 100ms). Moreover, RTU/PLC proxies can batch status updates from all the RTUs and PLCs in their substations, further reducing the number of updates the SCADA masters must process. To enable batching without requiring RTUs and PLCs to be on the network, a hierarchical approach can be used, in which each RTU or PLC is directly connected (via wire) to a "bump-in-the-wire" proxy, and aggregator proxies batch commands from many such proxies.

B. Intrusion-Tolerant Communication Library

System components that interact with the replicated SCADA master (e.g. the HMI and RTU/PLC proxies) cannot simply send updates to a single replica. Recall that under our threat model, one control center may be disconnected, and the other control center may include up to f compromised replicas and one replica undergoing proactive recovery. Therefore, each update must be sent to at least $f + 2$ replicas in each control center to ensure that at least one correct replica receives the update in a timely manner (the update may initially be sent to fewer replicas and re-sent to more replicas after a timeout only if necessary, but this adds latency).

To ensure that a received message is valid, the HMI or RTU/PLC proxy must verify that at least one correct replica reached agreement on that message. In many intrusion-tolerant replication systems, a client must receive $f+1$ identical copies of the message from different replicas, ensuring that at least one copy was sent by a correct replica. Spire instead uses an $(f + 1, n)$ threshold signature scheme, where at least $f + 1$ out of n total shares are required to create a valid signature, allowing the client to verify that at least $f+1$ replicas agreed on a message by verifying a single RSA signature on that message. This eliminates the need for clients to receive copies of the message from $f+1$ control-center replicas (which may not be possible under all failure scenarios).

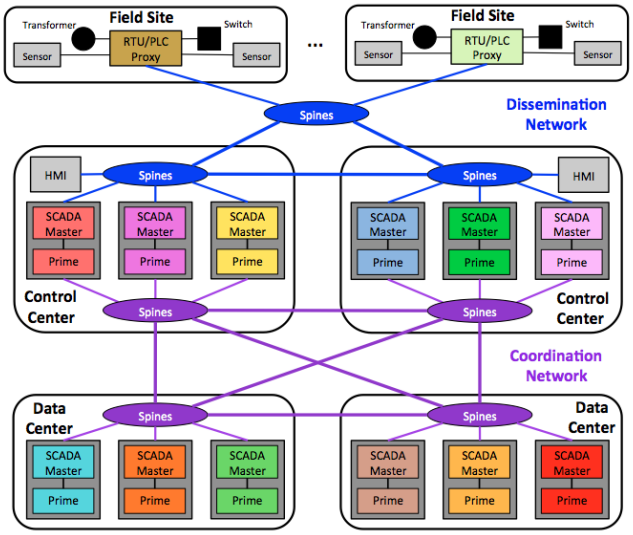

Fig. 6. Spire software architecture for configuration "3+3+3+3"

C. Intrusion-Tolerant SCADA System Architecture

Figure 6 shows the architecture for a complete Spire deployment using configuration "3+3+3+3". The SCADA master is replicated using Prime, and each replica runs a diverse variant of the software (represented as different colors in Figure 6). Periodically, the replicas are rejuvenated one at time, in a round-robin manner, to remove any potentially undetected compromises. Rejuvenating replicas follow the proactive recovery procedure, which includes generating a new diverse variant of the software.

While all replicas participate in the replication protocol, only the control-center replicas communicate with the RTU/PLC proxies in the field sites (substations). Therefore, we deploy two separate intrusion-tolerant Spines networks: a coordination network (purple Spines nodes in Figure 6) connects all of the replicas, and a dissemination network (blue Spines nodes in Figure 6) connects the control-center replicas with the RTU/PLC proxies in field sites (substations).

Normal System Operation. SCADA system updates are generated by the HMI and RTU/PLC proxies. Updates are sent over the dissemination (blue) Spines network to $f + 2$ replicas in each of the two control centers. Received updates are ordered by Prime on the coordination (purple) Spines network and then executed (in order) by the SCADA masters.

If an update triggers a response from the SCADA master replicas, the response is signed using threshold cryptography. Correct replicas (both in data centers and control centers) send their signature shares over the coordination (purple) network to the control center replicas. Once a correct control center replica receives $f + 1$ shares from different replicas for the same response content, it combines the shares to create a complete signature, and sends the signed response to the target clients (HMIs or RTU/PLC proxies) over the dissemination (blue) Spines network.

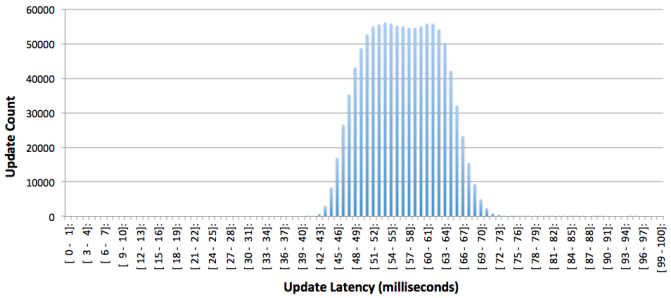

Fig. 7. Update latencies over 30-hour wide-area deployment (13 updates over 100ms not visible)

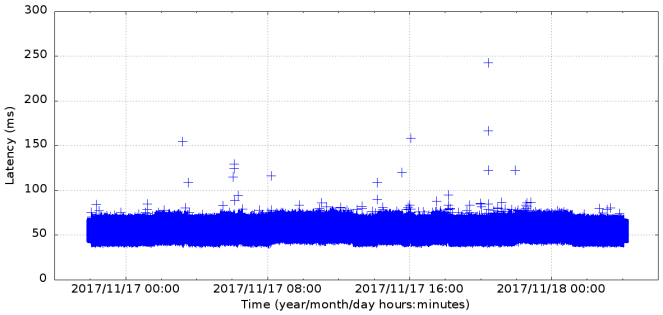

Fig. 8. Update latencies over 30-hour wide-area deployment

D. Bounded Delay in Practice

As discussed in Section III, Spire guarantees bounded delay. However, as noted in [32], the original analysis of Prime's bounded delay guarantee in [7] did not account for proactive recovery. The original analysis relied on the fact that eventually a leader will be elected that will never be suspected; however, when proactive recovery is used, even a correct leader will eventually be taken down for rejuvenation and lose its role as leader. Because Prime's view change protocol completes within a bounded amount of time, Spire can still support bounded delay, as long as we can bound the number of view changes required to settle on a new correct leader.[1]

In the worst case, when a correct leader is taken down for recovery, we may simultaneously have f compromised replicas and one disconnected or failed site. Because Prime chooses leaders in round-robin order, we stripe replicas across sites to prevent repeatedly trying to switch to a new leader in the same failed site. For example, for configuration "3+3+3+3", we place replicas 1, 5, and 9 in site 1, replicas 2, 6, and 10 in site 2, replicas 3, 7, and 11 in site 3, and replicas 4, 8, and 12 in site 4. Therefore, in configuration "3+3+3+3", settling on a correct leader may require executing $f + 2 = 3$ view changes (where the first view change tries to elect a replica from the disconnected site, the second tries to elect the compromised replica, and the third successfully elects a correct replica).

In general, the worst-case number of view changes required is: $f + 2 + \left\lfloor \frac{f+1}{S-1} \right\rfloor$, assuming striping of replicas across sites. This accounts for the f compromises, proactive recovery, and disconnected site, as well as the fact that when the total number of sites S is less than or equal $f + 2$, we can cycle through all S sites and try to elect a leader in the disconnected site multiple times. Note that the worst case occurs when the leader is in the site that becomes disconnected, the next leader is recovering, and the next f leaders are compromised.

VI. Evaluation

We deploy Spire in a real wide-area cloud environment to evaluate its ability to support the timeliness requirements of the power grid. We focus on configuration "3+3+3+3", as it offers a good balance between the number of sites and total number of replicas used to tolerate one intrusion. We then assess the feasibility of a range of system configurations, including support for multiple intrusions, using a local-area environment with emulated latencies between sites.

A. Wide-Area Deployment and Evaluation

We deployed Spire in configuration "3+3+3+3" across four sites on the East Coast of the US, spanning approximately 250 miles. This geographic span is similar to that of large US power grids. The sites included a cloud-provider control center, a development lab (acting as the second control center), and two commercial data centers. In this experiment, Spire monitored and controlled ten emulated power substations that introduced updates to the system via RTU/PLC proxies at a rate of one update per second per substation. The ten substations were emulated in a separate location from the four control- and data-center sites, with 5-7ms latency between the substations and the control centers.

To evaluate Spire's ability to support the requirements of power grid systems during normal operation, we conducted an extended test over a period of 30 hours. Each update submitted by an RTU/PLC proxy was ordered by Prime and delivered to the SCADA masters, which generated a threshold-signed response that was sent back to the proxy. For each update, we calculated the roundtrip latency from the time the update was submitted to the time the response was received. Figure 7 summarizes the observed update latencies. The average and median latencies were both 56.5ms, with 99.8% of the updates between 43.2ms and 71.6ms. The latency for each update is plotted in Figure 8. Out of 1.08 million total updates, nearly 99.999% had latencies below 100ms: only 13 updates exceeded 100ms, and of those 13, only one exceeded 200ms.

To evaluate Spire's performance under attack, we launched targeted attacks designed to test the system's ability to withstand all combinations of faults and attacks illustrated in Figure 2. Spire's performance under all combinations of a proactive recovery and a network attack (rows 1-4 in Figure 2) is shown in Figure 9. Proactive recovery of a non-leader replica (e.g. of replica 2 at 00:30) has no effect on the system's performance. Proactive recovery of the current leader (e.g. of replica 1 at 01:30) leads to a view change, which causes a brief latency spike (with two updates exceeding 100ms in this case). The disconnection of a non-leader site does not cause a

[1] As originally specified, Prime does not guarantee that every correct replica can act as the leader: the f slowest replicas may be suspected and removed from their role. However, when the geographic locations of the replicas and the normal-case latencies between them are known, this is easily fixed by imposing a floor on the acceptable turnaround time, so that the leader is never required to provide a faster turnaround time than the slowest correct replica is capable of (while not subject to a network attack).

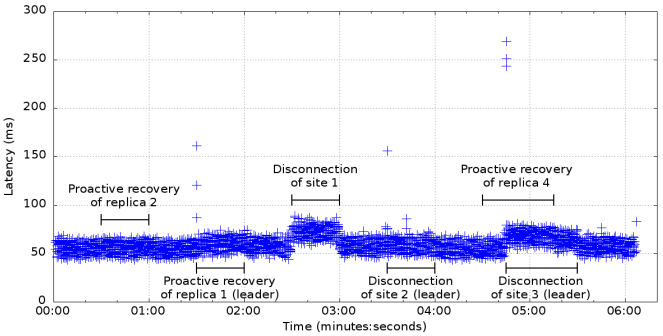

Fig. 9. Latency in the presence of network attacks and proactive recoveries

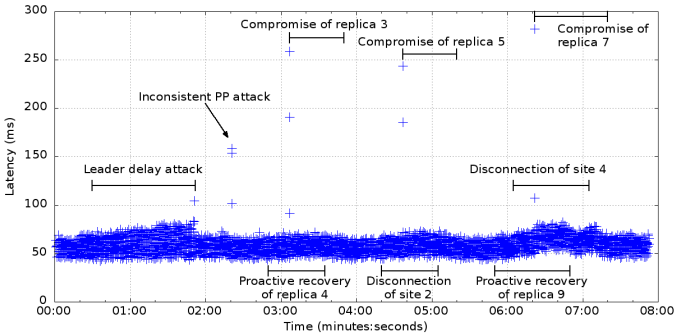

Fig. 10. Latency in the presence of intrusions, network attacks, and proactive recoveries

view change, but may cause an increase in latency, if the fastest (best-connected) quorum of replicas is no longer available (e.g. the disconnection of site 1 at 02:30). The disconnection of the site containing the leader will cause a view change and a corresponding latency spike (e.g. disconnection of site 2 at 03:30). Finally, the worst-case combination of a proactive recovery and site disconnection, where the leader site becomes disconnected while the next leader is undergoing proactive recovery, incurs two view changes, leading to a larger latency spike (e.g. with three updates exceeding 200ms at 04:45).

Spire's performance in the presence of an intrusion (rows 5-8 in Figure 2) is shown in Figure 10. While a compromised replica can perform arbitrary actions, we demonstrate Spire's resilience to two illustrative types of attacks. In the first attack, the leader generally acts correctly, to avoid being suspected and replaced, but attempts to increase the latency of updates. In Figure 10, from 00:30 to 01:50, the leader gradually adds delay, increasing update latencies up to about 80ms; however, when it tries to increase the delay beyond this, it is suspected and a view change occurs. In the second attack, the leader attempts to send a pre-prepare message containing incorrect values to violate the total ordering, but this is immediately detected, and the leader is removed in a view change (02:20). The remaining three attacks show the combination of an intrusion with a proactive recovery and/or network attack. At 03:05, the compromised leader acts maliciously and is detected and removed while the next leader is undergoing proactive recovery, causing two view changes to occur before settling on a correct leader. At 04:35, the compromised leader is removed while the next leader's site is disconnected, again incurring two view changes. Finally, at 06:20, the worst-case situation occurs, where the compromised leader is suspected at exactly the time that the next leader's site is disconnected and the leader after that is undergoing proactive recovery, forcing three view changes to occur before a correct leader is reached.

Overall, this evaluation demonstrates Spire's practicality in supporting the extended threat model in a real-life situation.

B. General Framework Feasibility Evaluation

While we consider configuration "3+3+3+3" the most practical configuration supporting one intrusion, we present a range of configuration options in Section IV and aim to support multiple intrusions. Therefore, we assess the feasibility of other configurations in a local-area environment with emulated latencies between the machines configured to match those observed between the geographic sites in the wide-area evaluation in Section VI-A. These results are shown in Table II.

First, we assess the three configurations supporting one intrusion from Table I in Section IV-E. The emulated results for configuration "3+3+3+3" match the wide-area results well: the average and median latencies are both 54.7ms, compared with 56.5ms, and 99.8% of update latencies are between 43.1ms and 67.1ms, compared with 43.2ms and 71.6ms. The differences between the two environments are explained by differences in the machines' processing power, as well as real-world latency fluctuations that were not captured by the emulation, leading to slightly higher latencies in the wide-area environment. While configuration "6+6+6" has a lower average latency than "3+3+3+3" (51.4ms), the higher communication and processing costs associated with using 18 replicas rather than 12 results in a slight increase in latency for the worst updates (67.1ms to 68.8ms for the 99.9 percentile). Configuration "3+3+2+2+2" shows higher overall latencies than the other configurations due to its additional site.

Since using four sites provides the best trade-off overall, we assess the feasibility of supporting two intrusions using configuration "5+5+5+4" and three intrusions using configuration "6+6+6+6". While configuration "5+5+5+4" shows higher overall latencies than "3+3+3+3" due to the additional replicas, they are still well within the acceptable range: all updates over the one-hour period are delivered within 100ms. This demonstrates that it is feasible to deploy Spire to support two simultaneous intrusions. Configuration "6+6+6+6" begins to reach the limits of the performance the system currently supports. While the average latency is acceptable, the latency of the worst updates increases considerably, and a small fraction of the updates (0.04%) exceed 200ms. We consider this performance to be borderline: more work is required to improve the engineering, replication protocol design, or cryptographic mechanisms to make this configuration deployable.

While it is always better to support more simultaneous intrusions, [32] shows that most of the benefit of proactive recovery can be obtained by supporting two intrusions, rather than one. Supporting more than two intrusions provides additional benefits, but with diminishing returns. Therefore, the ability to support two simultaneous intrusions in our demanding threat model (including network attacks) is a meaningful milestone.

	Avg Latency	% < 100ms	% < 200ms	0.1 percentile	1 percentile	50 percentile	99 percentile	99.9 percentile
6+6+6	51.4 ms	100.00	100.00	39.5 ms	40.6 ms	51.3 ms	63.8 ms	68.8 ms
3+3+3+3	54.7 ms	100.00	100.00	43.1 ms	44.2 ms	54.7 ms	65.4 ms	67.1 ms
3+3+2+2+2	56.4 ms	100.00	100.00	44.5 ms	45.8 ms	56.3 ms	67.3 ms	69.5 ms
5+5+5+4	57.4 ms	100.00	100.00	45.4 ms	46.6 ms	57.4 ms	68.8 ms	71.8 ms
6+6+6+6	64.8 ms	99.9111	99.9667	50.4 ms	52.2 ms	64.5 ms	82.7 ms	97.7 ms

TABLE II
SCADA CONFIGURATION PERFORMANCE ON LAN WITH EMULATED LATENCIES BETWEEN SITES FOR 36000 UPDATES OVER 1 HOUR

VII. SPIRE IN ACTION

In April 2017, we participated in a red-team exercise, in which an experienced hacker team attacked both a commercial SCADA system set up according to best practices and Spire. Within a few hours of attacking the commercial system, the red team completely took over the PLC controlling the mini power grid set up for the exercise by launching a man-in-the-middle attack between the SCADA master and the PLC.

In contrast, over two days of attacking Spire, the red team was not able to cause any disruption, due to the intrusion-tolerant network setup that prevented ARP poisoning attacks, the authentication and encryption of all traffic by Spire's intrusion-tolerant network, and the architecture that enforced that the PLC only communicate with the PLC proxy. On the third day, the red team was given access to a machine running one of Spire's SCADA master replicas and Spines nodes on both the dissemination and coordination networks. From this position inside the system, they launched denial-of-service attacks, but were not successful due to the fairness enforced by the intrusion-tolerant protocols.

During this exercise, the red team largely focused their efforts on network-level attacks, even from a compromised node, reinforcing the need to address the expanded threat model we consider in this work, with simultaneous network attacks and system compromises.

VIII. RELATED WORK

Spire builds on intrusion-tolerant replication to overcome system-level compromises. While Spire uses Prime, there are many other intrusion-tolerant replication protocols. Some, like Prime, guarantee performance under attack (e.g. [12], [13], [14], [15]), while others reduce cost by making stronger assumptions, such as using a trusted component to reduce the number of required replicas (e.g. [33], [34], [14]).

Intrusion-tolerant replication has previously been used to overcome SCADA master compromises. Zhao et al. [35] use PBFT [11] with four replicas to overcome one compromise. Kirsch et al. [29] use Prime to add intrusion tolerance to a Siemens SCADA product in a prototype implementation. However, these works are limited to a single control center, and thus cannot overcome the network attacks we consider.

RAM [36] and EBAWA [14] are intrusion-tolerant replication protocols designed to reduce overhead in wide-area environments. The work in [37] uses a similar approach to replicate a SCADA Master and Distribution Management System across several sites. In these protocols, each replica is placed in its own geographic site, resulting in a threat model that supports a total of f system-level compromises or benign site failures (e.g. natural disasters). However, these protocols do not consider network attacks. The benign site failures that they tolerate are not equivalent to the disconnected sites tolerated in our model: Spire supports a broad network attack model, but reduces the hard problem of overcoming sophisticated network attacks to the simpler one of overcoming a disconnected site using an intrusion-tolerant network. Moreover, using a separate site for each additional replica does not scale well with the number of faults that must be tolerated and may not be feasible for SCADA systems due to cost and latency constraints.

Steward [38] uses a two-level hierarchical replication architecture that, similarly to Spire, includes multiple replicas in each of several geographic sites. In Steward, each site runs its own intrusion-tolerant replication protocol, and representatives from each site participate in a higher-level replication protocol, reducing wide-area messaging costs. Steward's threat model does not support network attacks and limits the of number compromises tolerated per site, while Spire supports f replica compromises anywhere in the system. Moreover, Steward does not provide the bounded-delay guarantees necessary to support the latency requirements of SCADA systems for the power grid, and it is unclear how to do this in a hierarchical model.

An orthogonal approach to protecting critical infrastructure is to use intrusion-tolerant firewalls. For example, CRUTIAL Information Switches use intrusion-tolerant replication, diversity, proactive-reactive recovery, and access control to thwart external attacks [39]. The SieveQ [40] firewall uses two layers of replication to increase the supportable traffic load. Such firewalls are easy to integrate and reduce the attack surface by preventing external threats from reaching critical components (and Spire could benefit from such firewalls). However, if the firewall is breached or an insider attack is present, the Spire approach is needed as a last line of defense.

Another approach is to use domain-specific intrusion detection and response techniques. Such techniques leverage detailed knowledge of the power grid and coordinate information from multiple sources to detect malicious activity (e.g. [41], [42]), and prevent harmful effects from being applied (e.g. [43]) or quickly and automatically respond to attacks to limit their damage (e.g. [44]). While Spire overcomes compromises of the SCADA master, it does not prevent a malicious human operator from issuing destructive commands. Using these detection techniques, Spire could potentially identify and discard such malicious inputs. However, recent work shows that certain types of attacks can evade current detection methods using power grid-specific knowledge [45], further motivating Spire's intrusion-tolerant approach.

IX. Conclusion

We presented Spire, the first intrusion-tolerant SCADA system that simultaneously addresses system compromises and network attacks. Spire uses a novel framework for distributing SCADA master replicas across three or more active sites to ensure continuous availability and bounded delay under attack. Spire supports using commodity data centers with no access to field devices to augment existing power grid control centers. A wide-area evaluation of Spire shows that it meets the requirements of power grid control systems under attack.

X. Acknowledgement

We thank Kevin Jordan for inspiring us to work on intrusion-tolerant SCADA systems for the power grid. This work was supported in part by DARPA grant N660001-1-2-4014 to Johns Hopkins University and by DoD Environmental Security Technology Certification Program (ESTCP) Project EW-201607 to Resurgo LLC. Its contents are solely the responsibility of the authors and do not represent the official view of DARPA or the Department of Defense.

References

[1] C. M. Davis and T. J. Overbye, "Confirmation of a Coordinated Attack on the Ukrainian Power Grid," *SANS Industrial Control Systems Security Blog*, 2016.

[2] "IEEE standard communication delivery time performance requirements for electric power substation automation," *IEEE Std 1646-2004*, pp. 1–24, 2005.

[3] J. Deshpande, A. Locke, and M. Madden, "Smart choices for the smart grid," 2011, Alcatel-Lucent Technolgy White Paper.

[4] A. Studer and A. Perrig, "The coremelt attack," in *14th European Symp. Research in Comput. Security (ESORICS)*, 2009, pp. 37–52.

[5] M. S. Kang, S. B. Lee, and V. Gligor, "The crossfire attack," in *IEEE Symp. Security and Privacy (SP)*, May 2013, pp. 127–141.

[6] "Prime: Byzantine replication under attack," www.dsn.jhu.edu/prime, access: 2018-04-10.

[7] Y. Amir, B. Coan, J. Kirsch, and J. Lane, "Prime: Byzantine replication under attack," *IEEE Trans. Dependable and Secure Computing*, vol. 8, no. 4, pp. 564–577, July 2011.

[8] "The Spines Messaging System," www.spines.org, access: 2018-04-10.

[9] D. Obenshain, T. Tantillo, A. Babay, J. Schultz, A. Newell, M. E. Hoque, Y. Amir, and C. Nita-Rotaru, "Practical intrusion-tolerant networks," in *IEEE Int. Conf. Distrib. Comput. Syst. (ICDCS)*, June 2016, pp. 45–56.

[10] L. Lamport, R. Shostak, and M. Pease, "The Byzantine generals problem," *ACM Trans. Prog. Lang. Syst.*, vol. 4, no. 3, pp. 382–401, 1982.

[11] M. Castro and B. Liskov, "Practical Byzantine fault tolerance and proactive recovery," *ACM Trans. Comput. Syst.*, vol. 20, no. 4, pp. 398–461, Nov. 2002.

[12] A. Clement, E. Wong, L. Alvisi, M. Dahlin, and M. Marchetti, "Making byzantine fault tolerant systems tolerate byzantine faults," in *USENIX Symp. Networked Syst. Design and Implem. (NSDI)*, 2009, pp. 153–168.

[13] G. S. Veronese, M. Correia, A. N. Bessani, and L. C. Lung, "Spin one's wheels? byzantine fault tolerance with a spinning primary," in *IEEE Int. Symp. Reliable Distributed Systems (SRDS)*, Sept 2009, pp. 135–144.

[14] ——, "EBAWA: Efficient byzantine agreement for wide-area networks," in *IEEE Int. Symp. High Assurance Syst. Engineering*, 2010, pp. 10–19.

[15] Z. Milosevic, M. Biely, and A. Schiper, "Bounded delay in byzantine-tolerant state machine replication," in *IEEE Int. Symp. Reliable Distributed Systems (SRDS)*, Sept 2013, pp. 61–70.

[16] A. Avizienis, "The N-version approach to fault-tolerant software," *IEEE Trans. Software Eng.*, vol. SE-11, no. 12, pp. 1491–1501, Dec 1985.

[17] J. C. Knight and N. G. Leveson, "An experimental evaluation of the assumption of independence in multiversion programming," *IEEE Trans. Software Engineering*, vol. SE-12, no. 1, pp. 96–109, Jan 1986.

[18] M. Garcia, A. Bessani, I. Gashi, N. Neves, and R. Obelheiro, "OS diversity for intrusion tolerance: Myth or reality?" in *IEEE/IFIP Int. Conf. Dependable Systems Networks (DSN)*, June 2011, pp. 383–394.

[19] F. B. Cohen, "Operating system protection through program evolution," *Computers & Security*, vol. 12, no. 6, pp. 565–584, 1993.

[20] S. Forrest, A. Somayaji, and D. H. Ackley, "Building diverse computer systems," in *Wkshp Hot Topics in Operating Syst.*, May 1997, pp. 67–72.

[21] V. Pappas, M. Polychronakis, and A. D. Keromytis, "Smashing the gadgets: Hindering return-oriented programming using in-place code randomization," in *IEEE Symp. Sec. and Priv.*, May 2012, pp. 601–615.

[22] A. Homescu, S. Neisius, P. Larsen, S. Brunthaler, and M. Franz, "Profile-guided automated software diversity," in *IEEE/ACM Int. Symp. Code Generation and Optimization (CGO)*, Feb 2013, pp. 1–11.

[23] T. Roeder and F. B. Schneider, "Proactive obfuscation," *ACM Trans. Comput. Syst.*, vol. 28, no. 2, pp. 4:1–4:54, Jul. 2010.

[24] P. Sousa, A. Bessani, M. Correia, N. F. Neves, and P. Verissimo, "Highly available intrusion-tolerant services with proactive-reactive recovery," *IEEE Trans. Parallel Distrib. Syst.*, vol. 21, no. 4, pp. 452–465, 2010.

[25] A. Bessani, N. F. Neves, P. Veríssimo, W. Dantas, A. Fonseca, R. Silva, P. Luz, and M. Correia, "JITeR: Just-in-time application-layer routing," *Computer Networks*, vol. 104, pp. 122 – 136, 2016.

[26] H. Gjermundrod, D. E. Bakken, C. H. Hauser, and A. Bose, "GridStat: A flexible QoS-managed data dissemination framework for the power grid," *IEEE Trans. Power Delivery*, vol. 24, no. 1, pp. 136–143, 2009.

[27] A. Toonk, "Chinese ISP hijacks the Internet," bgpmon.net/blog/?p=282, 2010, access: 2018-04-10.

[28] T. Tantillo, "Intrusion-tolerant SCADA for the power grid," Ph.D. dissertation, Johns Hopkins University, 2018.

[29] J. Kirsch, S. Goose, Y. Amir, D. Wei, and P. Skare, "Survivable SCADA via intrusion-tolerant replication," *IEEE Trans. Smart Grid*, vol. 5, no. 1, pp. 60–70, Jan 2014.

[30] "pvbrowser. Simple process visualization," http://pvbrowser.de, access: 2018-04-10.

[31] "opendnp3," www.automatak.com/opendnp3/, access: 2018-04-10.

[32] M. Platania, D. Obenshain, T. Tantillo, R. Sharma, and Y. Amir, "Towards a practical survivable intrusion tolerant replication system," in *IEEE Int. Symp. Reliable Distrib. Syst. (SRDS)*, 2014, pp. 242–252.

[33] B.-G. Chun, P. Maniatis, S. Shenker, and J. Kubiatowicz, "Attested append-only memory: Making adversaries stick to their word," *SIGOPS Oper. Syst. Rev.*, vol. 41, no. 6, pp. 189–204, Oct. 2007.

[34] G. S. Veronese, M. Correia, A. N. Bessani, L. C. Lung, and P. Verissimo, "Efficient byzantine fault-tolerance," *IEEE Transactions on Computers*, vol. 62, no. 1, pp. 16–30, Jan 2013.

[35] W. Zhao and F. E. Villaseca, "Byzantine fault tolerance for electric power grid monitoring and control," in *Int. Conf. Embedded Software and Systems*, July 2008, pp. 129–135.

[36] Y. Mao, F. P. Junqueira, and K. Marzullo, "Towards low latency state machine replication for uncivil wide-area networks," in *Wkshp Hot Topics in System Dependability*, 2009.

[37] N. A. C. Medeiros, "A fault- and intrusion- tolerant architecture for EDP distribuicao SCADA system," Master's thesis, Univ. of Lisbon, 2011.

[38] Y. Amir, C. Danilov, D. Dolev, J. Kirsch, J. Lane, C. Nita-Rotaru, J. Olsen, and D. Zage, "Steward: Scaling byzantine fault-tolerant replication to wide area networks," *IEEE Trans. Dependable and Secure Computing*, vol. 7, no. 1, pp. 80–93, Jan 2010.

[39] A. N. Bessani, P. Sousa, M. Correia, N. F. Neves, and P. Verissimo, "The crutial way of critical infrastructure protection," *IEEE Security & Privacy*, vol. 6, no. 6, pp. 44–51, Nov 2008.

[40] M. Garcia, N. Neves, and A. Bessani, "SieveQ: A layered bft protection system for critical services," *IEEE Trans. Dependable and Secure Computing*, vol. PP, no. 99, pp. 1–1, 2016.

[41] S. Zonouz, K. M. Rogers, R. Berthier, R. B. Bobba, W. H. Sanders, and T. J. Overbye, "SCPSE: Security-oriented cyber-physical state estimation for power grid critical infrastructures," *IEEE Trans. Smart Grid*, vol. 3, no. 4, pp. 1790–1799, Dec 2012.

[42] A. Bohara, U. Thakore, and W. H. Sanders, "Intrusion detection in enterprise systems by combining and clustering diverse monitor data," in *ACM Symp. and Bootcamp on the Science of Security*, 2016, pp. 7–16.

[43] S. Zonouz, J. Rrushi, and S. McLaughlin, "Detecting industrial control malware using automated plc code analytics," *IEEE Security & Privacy*, vol. 12, no. 6, pp. 40–47, Nov 2014.

[44] S. A. Zonouz, H. Khurana, W. H. Sanders, and T. M. Yardley, "RRE: A game-theoretic intrusion response and recovery engine," *IEEE Trans. Parallel and Distributed Systems*, vol. 25, no. 2, pp. 395–406, Feb 2014.

[45] D. Shelar, P. Sun, S. Amin, and S. Zonouz, "Compromising security of economic dispatch in power system operations," in *IEEE/IFIP Int. Conf. Dependable Systems and Networks (DSN)*, June 2017, pp. 531–542.