# Going Dark: A Retrospective on the North American Blackout of 2038

Prashant Anantharaman
Dartmouth College
Hanover, NH, USA
pa@cs.dartmouth.edu

J. Peter Brady
Dartmouth College
Hanover, NH, USA
jpb@cs.dartmouth.edu

Patrick Flathers
Dartmouth College
Hanover, NH, USA
pflathers@cs.dartmouth.edu

Vijay H. Kothari
Dartmouth College
Hanover, NH, USA
vijayk@cs.dartmouth.edu

Michael C. Millian
Dartmouth College
Hanover, NH, USA
mcm@cs.dartmouth.edu

William G. Nisen
Dartmouth College
Hanover, NH, USA
william.g.nisen@dartmouth.edu

Jason Reeves
Dartmouth College
Hanover, NH, USA
reeves@cs.dartmouth.edu

Nathan Reitinger
Columbia University
New York, NY, USA
nathan.reitinger@columbia.edu

Sean W. Smith
Dartmouth College
Hanover, NH, USA
sws@cs.dartmouth.edu

## ABSTRACT

From March 29, 2038, to April 6, 2038, the world observed the North American Blackout of 2038. The blackout left upwards of 300 million people without power, ravaged the world economy, and devastated the global internet. By many accounts, it was the most devastating blackout ever witnessed. That said, its occurrence should not be surprising. While pundits harp on the technical sophistication of the adversary, debate the merits of a kinetic response, and politicize the blackout, the sad reality is that we have, for years, known we were susceptible to such an event. Moreover, we have had the requisite knowledge and tools to avert the blackout, but failed to use them. Plenty has been written on the wide-reaching societal effects of the blackout; our focus will be on the blackout itself.

The Blackout of 2038 had two major phases. In the first phase, an active adversary exploited a vulnerability in the implementation of the Wireless Access in SCADA Environment (WASE) protocol suite that supports the grid. Grid operators acted swiftly and switched to a fallback system to restore power. Unfortunately, the adversary then subverted the fallback system by exploiting a well-known vulnerability in DNP3, a popular industrial control system protocol. The led to the second blackout. Eventually, a patched implementation of the WASE protocol suite was developed and deployed, which restored power.

In hindsight, this blackout stemmed from two erroneous assumptions. First, immediately following the Texas Brownout of 2020, academics, industry professionals, regulators, and other stakeholders advocated for the adoption of a protocol that was formally verified to protect against race conditions (i.e., the cause of the brownout). However, it was wrong to equate formal verification with perfect security; we should have heeded the adage from Donald Knuth, "[b]eware of bugs in the above code; I have only proved it correct, not tried it." Second, we wrongly assumed a known-to-be-insecure fallback system would be an adequate stopgap until the primary system was back online.

This paper serves as a postmortem to the North American Blackout of 2038. We analyze how the failures came to pass and the assumptions that underlie them. Moreover, we offer a complete and simple solution to prevent these conditions from ever arising again: the adoption of Language-theoretic Security (LangSec) principles. To this end, we provide and evaluate a preliminary implementation of a LangSec parser for the WASE Short Message Protocol format (WSMP). Additionally, we urge lawmakers and regulatory agencies to mandate the verification of fallback protocols.

## CCS CONCEPTS

• **Security and privacy → Network security**; *Intrusion detection systems*;

## KEYWORDS

Language-theoretic Security, Smart Grid Security, Critical Infrastructure Security

**ACM Reference format:**
Prashant Anantharaman, J. Peter Brady, Patrick Flathers, Vijay H. Kothari, Michael C. Millian, William G. Nisen, Jason Reeves, Nathan Reitinger, and Sean W. Smith. 2018. Going Dark: A Retrospective on the North American Blackout of 2038. In *Proceedings of 2018 New Security Paradigms Workshop, Windsor, UK, August 28–31, 2018 (NSPW '18),* 12 pages.
https://doi.org/10.475/123_4


*NSPW '18, August 28–31, 2018, Windsor, UK*

ACM ISBN 123-4567-24-567/08/06. . . $15.00
https://doi.org/10.475/123_4

## 1 INTRODUCTION

In early 2038, the contiguous United States and parts of Canada experienced the most devastating blackout in history: the North American Blackout of 2038. An attacker penetrated the grid network, compromised the devices within it, and shut down the grid for more than a week. Effectively, society came to a complete halt.

The most alarming part of this attack is that it succeeded *despite* efforts spanning several decades to protect the grid. Preceding

the Texas Brownout of 2020, there were four nascent movements to modernize the power grid. These involved: making the power grid more robust and reliable by utilizing formal methods to prove race conditions would not occur; integrating the interconnections; making the power grid "smarter" and reconfigurable by adding an assortment of wireless devices; and adopting an efficient, blockchain-based, wireless protocol for convenient, efficient, and secure communication between new devices and old components of the power grid. Historically, updates to the power grid have been slow due to its reliance on legacy devices that cannot be gracefully replaced at low cost. However, the brownout served as a watershed moment for the power grid, providing both the opportunity and impetus to realize the objectives advocated by these movements [8].

Between 2020 and 2022, academics in collaboration with industry professionals developed the Wireless Access in SCADA Environment (WASE) protocol suite, which was formally verified to not exhibit race conditions. Simultaneously, the allure of more reliability and lower costs led to a tighter coupling of the interconnections covering the contiguous United States and portions of Canada, as well as the deployment of a number of new, wireless devices. Finally, after a rigorous testing phase, North American Electric Reliability Corporation (NERC) and other regulatory bodies mandated the adoption of WASE in 2024 [34]. True enough, a protocol of suits were maintained as backups (e.g., DNP3, Modbus, and IEC 61850), but the transition into WASE was final. Notably, this transition was also swift given the heightened enforcement abilities of NERC, which had, in 2019, gained a serious (and controversial) amount of teeth via a unique "hack-back" provision added to CIP 696-90 [12]. In part, the provision allowed Bulk Electric Systems to launch counterattacks on adversaries if certain preconditions were met—one of which being the satisfaction of all NERC CIP standards. This, combined with incrementally stricter standards set throughout the years, allowed for a healthy and productive relationship between Bulk Electric Systems and NERC.

While we did not observe any major issues with the smart grid between 2024 and 2038, the move to a formally-verified protocol like WASE turned out to be only a partial solution to grid security. Race conditions had been removed from the threat landscape, but many other problems remained, most notably a lack of input sanitization that allowed an attacker to exploit vulnerabilities in WASE via maliciously-crafted input. This oversight led directly to the North American Blackout of 2038.

Beginning on January 19, 2038, an active adversary launched an attack on the power grid, which involved infiltrating the IT and OT networks of a utility company and compromising three-quarters of Supervisory Control and Data Acquisition (SCADA) masters using a zero-day exploit, the Ping of Darkness, to subvert WASE and bring down the smart grid. Immediately following this attack, grid operators initiated the fallback system; however, the adversary immediately exploited vulnerabilities in DNP3 to bring down the fallback system as well. Although grid operators eventually restored power, the blackout highlighted the reality that the power grid was not nearly as secure as we had earlier thought.

Despite the size and scale of the incident, fixing the flaw was surprisingly simple: by verifying the advertised length of a WASE ping message against the actual size of the message, we could have blocked the Ping of Darkness and prevented the attacker from gaining control of the SCADA masters. To mitigate this vulnerability and others like it, we need to examine the specifications and implementations of our SCADA protocols, as well as ensure that any message accepted by a SCADA device is verified for correctness and is properly formatted. This idea, known as *Language-theoretic Security* [46, 54] or *LangSec*, must also be applied retroactively to protocols used within the grid's fallback system to ensure their resiliency.

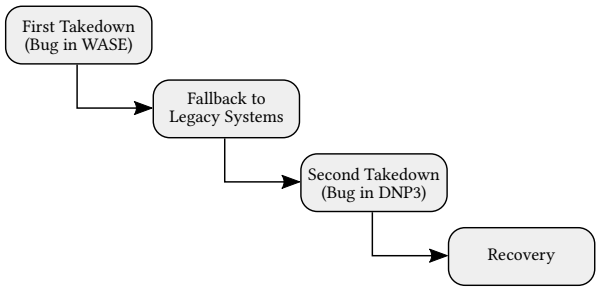


**Figure 1: Major stages of the 2038 blackout. The attacker first attacked the new systems that were in place post-2024. Grid operators reverted back to the legacy systems making use of ICCP and DNP3, while they looked at the blockchain to understand what values had been modified to bring the grid down. DNP3 systems were subjected to denial-of-service attacks by the attacker who still had access to the power grid via VPN.**

This paper provides the following contributions:

- We document previous power failures and explain how they provided the impetus for the development of the modern smart grid.
- We provide a detailed explanation of the Blackout of 2038.
- We retrospect on the blackout and examine the school of thought that led us to wrongly believe we were secure.
- We argue that LangSec and the adoption of reliable fallback systems are core to securing the power grid.
- We present and evaluate a preliminary implementation of a LangSec parser for the WASE Short Message Protocol format (WSMP).
- We make policy recommendations to improve grid resiliency.

This paper is structured as follows: in Section 2 we provide a brief history of the grid over the last two decades, touching on notable power failures and changes within the grid. In Section 3, we give an overview of the blackout itself. In Section 4, we reflect and offer solutions moving forward. Finally, we conclude our analysis in Section 5.

## 2 THE EVOLUTION OF THE MODERN POWER GRID

In this section, we look back on recent examples of power failures and infrastructure changes that led to the modern grid. We show how race conditions, malicious actors, and unverified assumptions have resulted in power failures in the past. We discuss how these

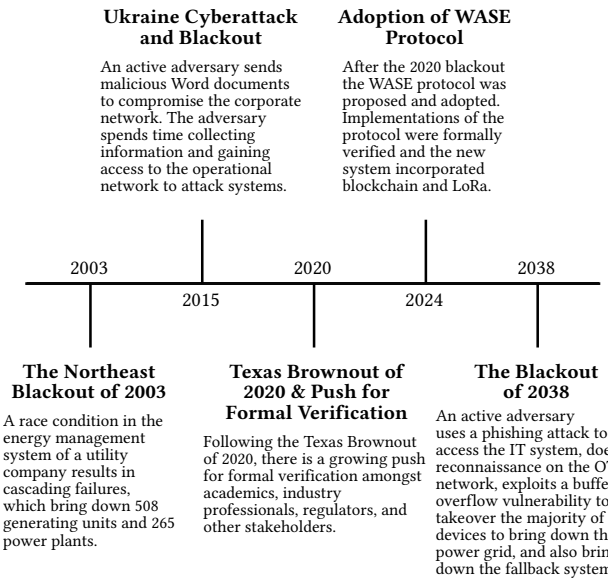


**Figure 2: Major events in the history of the US Power Grid**

events led to a stronger push to adopt formal verification as a means of hardening the grid.

## 2.1 The Northeast Blackout of 2003

The Northeast Blackout of 2003 was the most widespread blackout in North America before 2038. It affected over 50 million people across seven states in the Northeastern and Midwestern parts of the United States, as well as the province of Ontario in Canada [10].

A joint task force between United States and Canada found the blackout to be caused by "deficiencies in specific practices, equipment, and human decisions by various organizations" [28]. It also found that operator training was inadequate, that there was a failure to coordinate bulk power moves across the grid, and that the systems that needed to identify failing and/or emergency conditions needed to be more robust [28, Chapter 10]. These deficiencies caused a cascade of power plants tripping off-line from Thursday, August 14, 2003, at 4:00 PM, until 4:13 PM EDT [27].

Behind these failings, however, was a simple race-condition bug. This particular bug impacted the Energy Management System (EMS) owned by FirstEnergy in Akron, Ohio. The events unfolded as follows.

As the day of the blackout went on, alarms were stalled for over one hour, depriving operators of knowledge that showed some generation facilities and several 345 kV transmission lines had dropped off-line. FirstEnergy's grid was going dangerously under voltage. Additionally, the state estimator at Midcontinent Independent System Operator (MISO) had telemetry issues that morning which triggered a reboot—however, the monitoring portion of the system did not come back online, causing the state estimator to miss the failures happening at FirstEnergy.

Once FirstEnergy's primary EMS crashed, all applications were transferred to the "hot-standby" backup system. This system began processing, but the alarm application remained in a stalled state, causing the backup system to fail thirteen minutes later [28, page 54]. On top of this, even though other utilities were calling FirstEnergy operators directly with possible line overloads, operators at FirstEnergy did not act on these calls because no alarms were signaled by the EMS. As the voltage sag deepened, more operators throughout the grid tried to rebalance or disconnect, causing 508 generating units at 265 power plants to drop offline. When FirstEnergy operators finally realized the EMS issues, the damage had been done. Most of the area was without power for around one day, though though some pockets were without power for three days.

The blackout's primary cause was a voltage collapse in Ohio, but the race condition was the underlying flaw masking these abnormal conditions. Specifically, the GE XA/21 EMS had a subtle bug in its alarm and event processing routines. According to Mike Unum, manager of commercial solutions at GE Energy: "There [were] a couple of processes that were in contention for a common data structure, and through a software coding error in one of the application processes, they were both able to get write access to a data structure at the same time... and that corruption led to the alarm event application getting into an infinite loop and spinning." [50] This failure kept the FirstEnergy operators from being alerted to the loss of the 345 kV transmission lines as the EMS signaled normal conditions.

What is more, the bug would have been incredibly difficult to identify *a priori*. It took weeks for GE to find the fault even after the outage. GE sent patches to every utility with an XA/21. GE's Unum said, "We test exhaustively, we test with third parties, and we had in excess of three million online operational hours in which nothing had ever exercised that bug. I'm not sure that more testing would have revealed that. Unfortunately, that's kind of the nature of software... you may never find the problem. I don't think that's unique to control systems or any particular vendor software." [50]

Some recommendations from the task force involved creating a new set of reliability measurements to hold the utilities accountable. Most of the recommendations, however, involved cybersecurity standards to keep potential attackers out rather than to improve overall software reliability [28, ch. 9-10].

## 2.2 The Ukraine Attacks of 2015 & 2016

The first successful cyberattack on a power grid occurred in Ukraine on December 23, 2015, affecting three power generation companies and about 225,000 customers [60].

The outages began when hackers entered a power distribution company's computer and SCADA systems. Starting at approximately 3:35 PM local time, seven 110 kV and twenty-three 35 kV substations were disconnected for three hours. Local news indicated that the attack impacted additional portions of the distribution grid and forced operators to switch to manual mode as the command and control systems were infected with a virus [24]. Since manual control was available, most people were without power for a maximum of six hours.

In fact, the attack started several months earlier [47] when hackers used spear-fishing emails with BlackEnergy 3 [57] malware

embedded in Microsoft Office documents. The emails were targeted at the grid's corporate network, particularly the command and control IT systems [29]. Once a foothold was established, attackers worked their way through the IT network, collecting VPN credentials and escalating their privileges.

At this point, the attackers were able to identify particular VPNs and other connections to the operational technology (OT) network. From IT into OT, attackers identified types of systems and hardware available as attack vectors.

Upon launching a multi-faceted attack on the OT systems, attackers accomplished the following [48]:

- Take ownership of the SCADA control system and switch substations off remotely by opening and closing the breakers. Grid operators were helpless as attackers overrode all human–machine interface systems.
- Destroy or disable IT infrastructure (e.g., remote terminal units, modems, and Ethernet to serial devices) by overwriting firmware.
- Destroy control system workstations and servers by erasing master boot records with the KillDisk malware.
- Begin a denial-of-service attack on the power company call-centers to prevent customers from getting information or reporting outages.

This attack had much in common with a false data injection attack, where an adversary corrupts the power system by injecting false data into metering operations in a controlled way. Examples would be attacking the grid state estimator [44] or attacking energy distribution by corrupting the data to/from smart meters [43]. These attacks assume the infiltrators understand the power system's features in high- and low-level detail (e.g., network topology and the command and control infrastructure), allowing them to manipulate key measurements and controls.

The Ukraine attackers certainly possessed these abilities. The SANS report confirmed this [21]: "the strongest capability of the attackers was not in their choice of tools or in their expertise, but in their capability to perform long-term reconnaissance operations required to learn the environment and execute a highly-synchronized, multistage, multisite attack."

## 2.3 The Texas Brownout of 2020

In 2009, plans were unveiled for the Tres Amigas Superstation (TAS) [6] to unify the power grid in the continental United States by connecting the Eastern, Western, and Texas Interconnections. The United States had never before managed a grid this large, and the physical constraints of reactive power led to unforeseen consequences. In particular, a brownout reminiscent of the Northeast Blackout of 2003 occurred in 2020. But first, a brief infrastructural background will be necessary.

As electrical requirements on any grid increase, the amount of reactive power needed to maintain proper voltage increases. Real power travels further in lines than reactive power, so the further power moves over transmission lines, the more reactive power is consumed. If reactive power supply cannot meet the need, a voltage drop will occur [35].

As PJM, a regional transmission organization and part of the Eastern Interconnection, discovered several years earlier, increasing the coverage footprint requires implementation of additional Transfer Interfaces to measure power flows across select high voltage lines [17]. This equipment is used to ensure proper levels of reactive power. Local generation is utilized as needed to maintain the necessary voltage levels.

Several months after the TAS was completed, the Texas Interconnection experienced a brownout from 3:42 PM until 4:18 PM Central Time on June 11, 2020. The event was caused by several Transfer Interface processes on the Texas Interconnection side of the TAS writing to a common data structure simultaneously. As a result of the ensuing race condition, the voltage flowing from the Eastern and Western Interconnection was never supplemented by local generation and did not have enough reactive power. All stations down the line in the Texas interconnection suffered voltage collapse.

The event ended after half an hour when local generation spun up to compensate. The bug was discovered and stopgaps were put in place to prevent recurring brownouts while the bug was fixed. This was the second race-condition bug in recent memory to result in a significant power disruption in North America, and it led to a strong push for formal verification.

## 2.4 The Push for Formal Verification

The value of formal methods is not a new concept, as the now-ironic statement from a 2016 NSF workshop report propounds: "Formal methods are the *only* reliable way to achieve security and privacy in computer systems" [9]. Unfortunately, and perhaps unsurprisingly—"It is [surely] axiomatic that the more secure one wants a system to be the more money one must be willing to pay"—it took an event like the Texas Brownout of 2020 to finally push our country in the right direction and utilize these methods.

The process of formal methods involves using mathematical analysis to assess the state of a system. The idea works like this: if we can describe a system in mathematical terms, then we will be able to manipulate those terms to prove or disprove properties of the system [9]. The value here comes from formal verification: if the system is verified, then we have proved it correct, and we will have a guarantee that it resists all possible attacks under the model.

Functionally, formal methods are broken into three parts [62]. First, a formal specification is created which outlines the properties of the system. For example, a non-root user cannot access root-level files. Second, a model of the system is produced. This may be a finite state machine or a syntactic local expression. Finally, the formal specification is proved. Likely, the proof would involve logical methods (e.g., Boolean logic, propositional logic, first-order logic) and is typically solved by an automated computer program. We may either work through all states and check the specification in each one (model checking) or walk through the logical theorem and check the end results (theorem proving) [55]. Once the third step is complete, we will have proven or disproven certain properties of the specification.

But this workflow also points out a notable flaw in formal verification. The verification's accuracy hinges on the specification's description of the system. Stated otherwise, the specification may

be proven correct, but that does not matter if the specification does not accurately describe real-world conditions.

To overcome this hurdle, work on type-safe functional languages has been developed. For a simple example, consider ACL2, a programming language with a logic built in [38]. Imagine a programmer wanted to build a program in ACL2 which multiplies "x" with "y" to produce "z". This could be accomplished with the language itself; importantly, however, ACL2 may also be used to verify that multiplying any two natural numbers will execute correctly—a provably correct statement about an infinite number of test cases [39]. A more complex example comes from Charguéraud and Pottier's work on verifying the Union-Find algorithm with a modular OCaml library [7].

Academics have studied ways formal verification can be used to address and avoid race conditions, as well as deadlocks in code [4, 22], and have pushed for the inclusion of tools that do precisely this in the context of securing critical infrastructure. The code written needs to be annotated with certain keywords, and the resultant binary simulates all possible outcomes of the program, ensuring there are no conditions that violate the assertions annotated in the program. These tools check to see which locks are acquired and which critical sections they protect. Formal verification can also detect whether shared accesses to memory is dangerous and nudge the developer toward solutions.

Prior to the Texas Brownout of 2020, the value of formal methods was well-acknowledged in academia, but received little attention elsewhere. Following the brownout, formal methods received more attention in the smart grid industry. NERC set up a committee to design a standard for a new suite of protocols. The proposed WASE protocol suite would be resilient to race conditions, easier to use, and primarily wireless.

## 2.5 The WASE Protocol Suite

Following the 2020 brownout, NERC pushed for the Wireless Access in SCADA Environment (WASE) protocol suite [30]. The smart grid industry took a cue from the automotive industry, which had embraced Wireless Access in Vehicular Environment (WAVE) protocol for vehicle-to-vehicle communications.

The WASE protocol suite provided a number of technological advances in that it:

- Uses blockchain for distributed consensus among SCADA masters.
- Uses wireless communication between SCADA masters and outstations, as well as consumer devices like cars and unmanned aerial vehicles.
- Uses NERC-sanctioned implementations of WASE within utilities to ensure uniformity. These implementations were formally verified to guarantee the absence of race conditions.

The WASE standard makes use of two different bandwidths for intra-substation and inter-substation communications. Inter-substation communications are long range, usually over hundreds of miles. The techniques eventually adopted in WASE were proposed as early as 2011 and brought into the testing phase in 2022 [19, 42, 58]. WASE Short Message Protocol (WSMP) is one protocol from the WASE protocol suite that provides support for blockchain messages, authentication, and keep-alive operations.

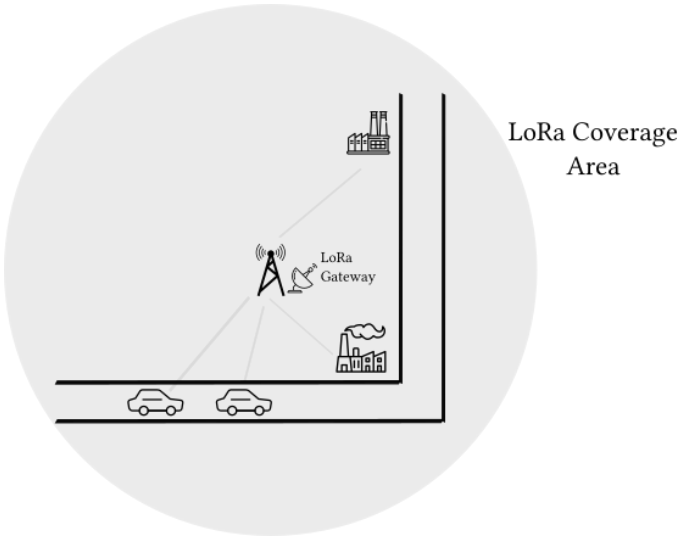


**Figure 3: The WASE protocol suite allows for a LoRa gateway to cover a radius of 20 kilometers. In a particular LoRa region, several smart meters and substations communicate with each other and smart vehicles get authorization to charge.**

**LoRa technology to communicate between substations.** LoRa is a low-power and high-range data communication protocol, used primarily in the physical and data-link layers of the network stack. Gateway transmitter height is 100 meters, and the height of the receivers at all substations is set at 10 meters. This allows a maximum range of 20 kilometers between a gateway transmitter and a substation with a LoRa receiver. Authentication in LoRa relies on the application layer protocols making use of it, in this case the WASE protocol suite. WASE involves end-to-end encryption, authentication, and identification of all endpoints, SCADA masters, and SCADA devices.

**LoRa technology between smart meters and substations.** Smart meters send their reading information to the prescribed substation directly using LoRa. The substation sends back demand-response signals to the smart meter, asking it to turn off specific equipment in houses, depending on the cost of power at that time of the day.

**WAVE technology between electric cars and smart meters.** The WAVE protocol is a vehicle-to-vehicle protocol, which is used to coordinate traffic in self-driving cars. The WASE protocol suite is used for vehicle-to-infrastructure communication, such as coordinating payments, maps, and charging electric vehicles. The WASE protocol suite addresses issues of authentication and identification for vehicles that continuously move and need to authenticate to new networks to continue performing operations seamlessly.

**Adoption of the blockchain to perform state estimation.** DNP3 slaves are connected to actuators and sensors, gathering power system measurements directly. These measurements are then passed on to the DNP3 Master. The Inter Control Center Communications Protocol (ICCP) is used to transmit this data

between the control stations. ICCP is set up as a client-server mechanism, with a control center communicating its state values to another control center via a server. In light of the 2020 brownout, NERC decided to try and avoid any single point of failure, and advocated for the use of blockchain to replace the ICCP protocol as a part of the WASE standard. A control center now broadcasts a message with: a unique transaction identifier, the data portion, the sender ID, the recipient ID, the length field, and a signature script containing the public key and the signature hash. SCADA masters verify and mine these messages. The slaves only validate the recipients and perform the setpoint changes as ordered.

These advances revolutionized the grid, even paving the way for the reconfigurable grid as we know it.

## 2.6 The Reconfigurable Grid

Some consumer "smart grids" were attempted in the early-2000s but had management difficulties as the smart meters, smart appliances, alternative energy generation and electric cars had different or too simple a set of protocols. Utilities and consumers were also looking at conflicting goals: the utilities were looking for ways to better control peak demand, while consumers were looking for ways to control the amount and cost of electricity used. An example of this type of failure was the "Smart Grid City" concept started in 2008 by Xcel Energy in Boulder, Colorado, where cost overruns caused by betting on wired versus wireless technology, incompatible hardware, and a mismatch between utility and customer goals caused the project to be abandoned by 2014 [2, 33].

One early-century trend that continued was the incorporation of renewable generation into the smart grid. Driven by falling hardware costs and a desire to minimize power disruptions, many consumers began installing solar panels, power walls, and other hardware to generate and store power locally for use when the larger grid was unavailable. This push for distributed, renewable generation was so widespread that it allowed North America to reach its goal of drawing 50% of its power from non-carbon sources by 2025 as set by the United States, Canada, and Mexico at the "Three Amigos" Summit [52]. However, the introduction of so many generation sites and the need to coordinate their actions in the event of an outage required the industry to rethink its reliance on wired technology.

Wireless protocols accelerated the rise of the reconfigurable grid, where microgrids can be created in units as small as one house up to large neighborhoods. The push by NERC to have better interoperability standards allowed utilities and homes to tailor power needs based on demand and cost. Taking a cue from late-twentieth-century systems that allowed utilities to disconnect business customers with co-generation facilities from the grid in times of high demand, modern utility energy systems can rapidly interface with individual homes that have solar, wind, battery or even a charged electric car and have them "island" themselves for high demand periods. Areas with co-generation and groups of homes with excess power can communicate their energy needs or excess availability and band together to form larger islands until the demand spikes are over. Paradoxically, this was also the technology that pushed the country towards the Blackout of 2038.

## 2.7 Recap

The movement for formal verification and adoption of the WASE protocol suite has led to a grid that is reconfigurable, more efficient, more resilient, and theoretically more secure. However, the use of formally-verified code was a highly-targeted approach that did not address all of the security issues found within the power grid. Race conditions were eliminated as a threat to grid security, but other problems remained. The most critical problem was improper input handling, which had led to many industrial control system protocol vulnerabilities in the past [14]. This incomplete solution to grid security came back to haunt the United States.

# 3 THE BLACKOUT OF 2038

From March 29, 2038, to April 6, 2038, the world experienced the North American Blackout of 2038. A deluge of prose has been written on the impact of this event. Our effort, however, heeds a call from NERC in their recent whitepaper "The Cyber Kill Chain for the North American Blackout of 2038 [13]". NERC seeks suggestions from the academic community on how to prevent such an attack from ever occurring again. In short, after performing an analysis of the events culminating in the blackout, we suggest a permanent fix to these troublesome bugs: the implementation of LangSec.

Broadly, the Blackout of 2038 can be disaggregated into five steps: intrusion into the IT network; compromise of over 75% of the WASE devices; modification of setpoint values of compromised WASE devices; reversion to fallback legacy systems; and subversion of the legacy systems, resulting in a second nationwide blackout a single hour after power was restored.

## 3.1 The Intrusion

On January 19th, 2038, attackers gained access to the virtual private network of New Hampshire Utilities through a phishing attack. The adversaries spent approximately two months on reconnaissance, discovering which devices were on the OT network and the operational parameters of these devices. The adversaries also observed network traffic to determine which protocols were used, how they functioned, and what features were used.

## 3.2 The Takedown Bug

In part a response to the Texas Brownout of 2020, NERC mandated that all utility companies use the open source implementation of the WASE protocol suite, which was available as an IEEE standard. Building on this publicly-available resource, with additional inspiration taken from the 2014 Heartbleed bug [20], the adversary figured out how to send malformed WASE ping requests to SCADA masters on the network—i.e., the Ping of Darkness.

**Ping of Darkness.** The WASE protocol ping message is an optional feature used by several SCADA devices to keep their connections to the SCADA masters awake when there is no other traffic. The protocol is simple; it involves the SCADA master sending a ping message with a payload to make sure the device is awake. The client sends back a response with the same payload. The connection timeout for the WASE protocol was prescribed to one hour with no activity.

The length field, although accepted, was never validated with the length of the text. For example, a message with the length field

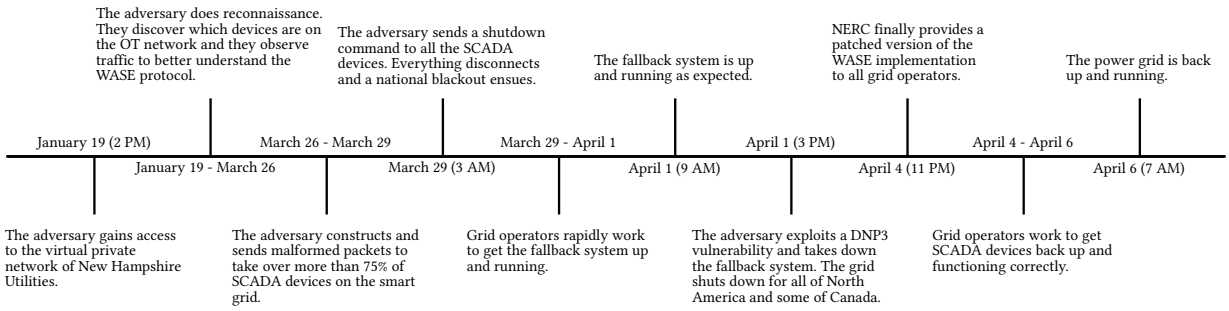


**Figure 4: Blackout of 2038 - Timeline of Key Events**

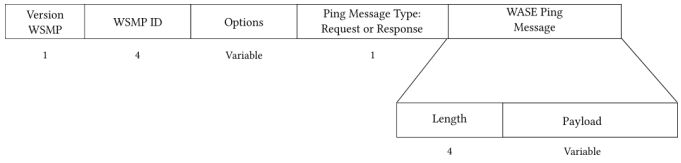


**Figure 5: WSMP packet format with length fields in bytes.**

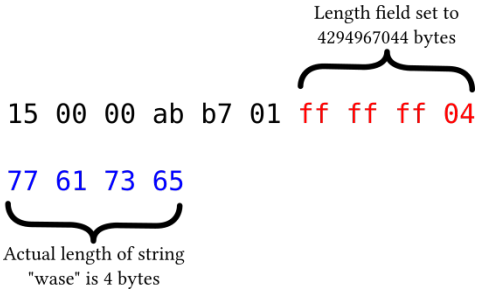


**Figure 6: The Ping of Darkness exploit, showing how an unvalidated length field, can force the receiver to send bytes from the stack. This is an example of a buffer over-read bug.**

set to 4294967044 bytes, and the payload to a four character string, "wase", leads to the SCADA device sending back "wase" followed by bytes from the stack totalling 4294967040. Figure 6 shows an example of this packet.

On March 29, 2038, attackers leveraged the Ping of Darkness to control more than three-quarters of the 11,567 SCADA masters on the smart grid. However, because SCADA masters were also involved in mining and verification of blocks, and the attackers controlled such a large portion of the WASE blockchain network, they are able to suppress successful blocks from other legitimate and uncompromised SCADA masters. The attackers then sent setpoint commands to open all breakers which where SCADA devices. This event caused a cascading failure.

## 3.3 Falling Back to Legacy Systems

An emergency Grid Resiliency Task Force was called into action to help decipher how the attack happened. The group split into two teams, one investigating the cause of the WASE protocol crash, and the other focused on reviving the dormant legacy systems (i.e., those using DNP3, Modbus, and IEC 61850 from prior to the 2024 revamp).

Grid operators traversed blockchain transactions to understand what setpoints were modified in all of the devices. Upon completion, operators learned that each transaction had to be reverted manually before switching to the backup system. Grid operators wanted to ensure that the bug would not recur and were exhaustive in their tests.

Three days after the blackout, grid operators successfully revived the legacy systems. At this point, the grid was operational. NERC promised to deliver a patched version of WASE by April 1, 2038.

## 3.4 Takedown 2.0: Attacks on DNP3 systems

Unfortunately, prior to NERC's patch, the attackers were able to bring to life a demo conducted by Adam Crain and Chris Sistrunk in 2014, showing how several implementations of DNP3 had many input-handling vulnerabilities [14]. Because the attackers already had access to the VPN of New Hampshire Utilities, they could send crafted packets to DNP3 SCADA masters from the control center. This allowed the attackers to perform denial of service attacks, causing another cascading failure throughout the country.

Specifically, SEL-2241, SEL-3505, and SEL-3530 RTAC master devices throughout the network were subject to a particular crafted packet attack [15]. These devices did not validate user data correctly, and when packets were sent with only the DNP3 link-layer headers, the SEL implementations of DNP3 went into an infinite loop expecting more data. This loop crashed most SCADA masters. Scripts left by the attackers continuously sent these payloads, meaning that when grid operators restarted the devices, they immediately crashed again due to this input-handling bug.

# 4 RETROSPECTION: TOWARD A COMPLETE SOLUTION

In this section, we discuss the limitations of formal verification, prescribe an approach to constraining and verifying properties of the input language grounded in LangSec, and highlight the importance of resilient fallback mechanisms.

## 4.1 Formal Verification is Not Sufficient

Formal verification can be a valuable tool in securing the power grid, but it is by no means a panacea. Its value is predicated on the correctness of the underlying assumptions embedded in the formal specification upon which it relies. To provide desirable properties of a real-world implementation it is necessary, but may not be sufficient, to acknowledge and prove the correctness of assumptions, e.g., pre-conditions in the formal specification must hold, post-conditions must truly represent a correct state, the formal specification must map correctly to the real-world implementation, and real-world physical quirks must be acknowledged. This may further be compounded by user error. Indeed, such limitations have been acknowledged in the past. For example, Hall [31] debunked the myth that "formal methods can guarantee that software is perfect." In practice today, there is an unwarranted amount of trust placed in formal verification.

The Ping of Darkness, and the bugs in the fallback DNP3 systems, were both primarily due to a lack of memory safety. In 2017, Bhargavan et al. presented a tool to translate F star (F*) code to a well-behaved subset of C and prove memory safety [51]. F star is a general purpose programming language aimed at program verification. This research also showed that the cryptographic libraries built using these tools were resilient to not only memory attacks, but also some side channel attacks. The adoption and usability of such tools in production has, however, been slow. Most systems making use of verification in the last ten years focused on modeling the system, and verifying properties like race conditions—but properties of the code directly, like in the work by Bhargavan et al., has not reached high adoption [51].

We stress that these limitations are not simply theoretical concerns— misapplication of formal verification happens in practice. For one example, in 2017, Fonseca et al. found buggy implementations of formally verified distributed systems arising from unenforced assumptions [26]. Eight of the sixteen bugs were input-handling bugs, which could have been avoided by implementing a parser that formalizes and codifies the assumptions on the input to be accepted by the program. Twenty one years later, we are running into a similar set of problems.

Even with formal verification becoming a standard part of the software development cycle in many companies, we still have a prevalence of input-handling bugs. In 2025, an in-depth survey of CVEs revealed at least 60% of reported bugs are related to poor input-handling. "This number is assuredly higher, since a substantial number of CVEs still do not provide enough details to classify the causing error" [53].

To improve the security of the power grid, we must acknowledge and understand the limitations of formal verification. This does not mean we should abandon formal verification altogether. Instead, we should responsibly apply formal verification and supplement it with other techniques. Notably, LangSec provides essential tools built upon a sound theoretical foundation to avert vulnerabilities such as the Ping of Darkness.

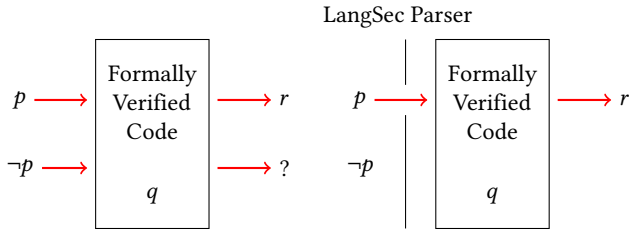


**Figure 7: On the left, we have a representation of formal verification: for input $p$ and formally verified code block $q$, output $r$ is always produced. However, for any input $\neg p$ the behavior of code $q$ is not specified. On the right, we have a representation of formal verification with a LangSec parser. Only input that matches $p$ is sent to code $q$.**

## 4.2 Language-theoretic Security

To mitigate blackout-inducing events like the ones above, we must go beyond formal verification and revisit both the specifications and the implementations of the protocols used for communication in SCADA networks. This idea, known as Language-theoretic Security [46, 54] or LangSec, focuses on identifying a secure subset of a protocol and ensuring that this subset is handled the same way across every input handler in the system. Figure 7 demonstrates how LangSec complements formal verification.

LangSec had been pushed as a solution in the early 2010s, and was shown to mitigate input-handling vulnerabilities in commonly-used protocols such as DNP3 [5], Modbus, and IEC 61850 [3]. However, the technique did not achieve widespread adoption, and the effort was abandoned after the introduction of WASE and the inclusion of formal verification.

Despite its lack of adoption, LangSec has been refined over the years to expand its protection capabilities. In addition to verifying the format of individual packets, LangSec can now examine both the data and metadata of a packet and assign it a "reputation score" to give the program an idea of whether or not the packet is trustworthy [32]. Tools have also been developed to make it easier to build LangSec parsers; for example, Apron [56] provides a simple drag-and-drop interface for constructing a parser with the existing Hammer toolkit [1], while NailGun [40] can automatically generate a parser, unparser, and fuzzer for a protocol based on the provided specification.

Before we discuss a LangSec parser specifically for the WSMP format, it will be helpful to show how LangSec may be applied to any protocol in a three-step process:

(1) **Define a subset of the protocol grammar that is "no more complex than deterministic context-free" [46].** Our goal here is to verify the safety and liveness properties of the protocol [41]; in other words, we want our parser to always come to an accept/reject decision and never accept any input that could compromise our system. While this sort of verification is undecidable in the general case, certain classes

of languages (specifically, regular and non-deterministic context-free ones) are decidable and thus are amenable to proper formal verification [54]. Therefore, to satisfy this step we must identify a subset of our targeted protocol that satisfies this context-free restriction, and then produce a formal grammar that encapsulates this set. Performing this task is a largely manual effort and must be done for each protocol we target, but this work has already been done for legacy SCADA protocols, and only needs to be done once for WASE.

(2) **Develop a formal parser for the secure protocol subset.** Once we define a secure protocol subset, we can turn to a tool such as TRX [37] to produce a formally-correct parser that satisfies the safety and liveness properties.

(3) **Use this parser exclusively wherever this protocol is accepted as input.** To avoid issues arising from "mutually intelligible dialects" [54] of a protocol, we use the formal parser everywhere we handle this specific protocol, ensuring that each instance produces identical output when dealing with the same input. (Failing to do this can lead to malicious inputs being interpreted and displayed as legitimate ones, such as with CN fields in X.509 certificates [36].) This way, we know that malicious and incorrect input will be treated correctly no matter where the input is encountered.

In the case of the 2038 blackout, a proper LangSec parser would have blocked the Ping of Darkness by verifying that the actual length of the packet matched its advertised value. Once it detected a mismatch, the packet would have been immediately rejected, keeping it from reaching the vulnerable section of the program.

## 4.3 A LangSec Parser for the WSMP Format

We implement a preliminary parser for the vulnerable WSMP format making use of the Hammer parser combinator toolkit [1]. We first extract the grammar of the message formats by reading through the specification. The description of the grammar of the message formats need to be in a machine readable format, and not just in verbose text like in the specification of the WASE protocol. ASN.1 syntax was used in the past to describe the X.509 certificate syntax. Parser bugs continued to occur in several popular implementations of cryptographic libraries due to poor implementation of ASN.1 syntax. We propose the usage of a new machine readable syntax making use of the XML syntax. Our sample XML reference and source code for the WASE ping message can be found in Source Code 1. We perform CPU time analysis on our parser, and find that it runs in 27 $\mu$s, and only requires 8 lines of code. Therefore proving that in terms of both human effort and the CPU time, the parser does not add any significant, witness-able overhead to the implementations of WASE. Our experiments are performed on a legacy device - an octa-core Intel i7 CPU with 16 GiB RAM.

**Source Code 1: Our implementation of the WASE Ping parser in python, using the Hammer parser library [1].**

```
<sequence>
  <bits name="version">8</bits>
  <uint32 name="wsmp_id"/>
  <uint32 name="length"/>
  <choice>
    <choiceVal>
      <token>"\x01"</token>
      <token>"\x02"</token>
    </choiceVal>
  </choice>
  <lengthVal>
    <length><uint32/></length>
    <value><ch_range start="a"
    ↪  end="z"/></value>
  </lengthVal>
</sequence>
```

```
def ping_parser():
    version = h._h_bits(8, True)
    wsmp_id = h.uint32()
    length = h.uint32()
    message_type = h.choice(h.token("\x01"),
    ↪  h.token("\x02"))
    payload = h._h_length_value(length,
    ↪  h.ch_range('a', 'z'))
    final = h.sequence(version, wsmp_id, message_type,
    ↪  payload)
    return final
```

## 4.4 Parser Deployment

The requirements of the power grid present numerous challenges to forklifting existing implementations of WASE. Reliance on legacy programs and devices, emphasis on availability, and minimizing downtime force us to think more about how we can manipulate program binaries or modify hardware.

However, readily-available tools such as secure conduits [18] may expedite this process. Secure conduits enable us to patch binaries dynamically by swapping the code for the old parser with a newer parser. Another technique is to push an update to the software switches to filter packets using the parser [45]. This would greatly minimize downtime, and the grid utility can push the update to a large number of these devices with ease.

Before the deployment of the parser, various implementations of the WASE protocol suite must be studied. This can be done by making use of a proxy, that would exhaustively validate packets in the network. This would give us a good sense of how well our parser is performing, and simulating attacks in this environment would give us a chance to understand the pitfalls of using such a parser. This approach has been used previously to identify malformed packets and determine which packets would trigger an attack on the DNP3 protocol endpoints [5].

Deploying LangSec parsers is cost-effective because they can prevent an entire class of input vulnerabilities from becoming dangerous exploits, thus reducing the likelihood of widespread blackouts in the future. The cost of these blackouts are staggering; for example, the North American Blackout is estimated to cost between $80 and $150 billion [25]. In contrast, the deployment cost of LangSec is marginal given the maturity of LangSec and the availability of deployment tools and techniques. This makes the decision to use LangSec parsers an obvious one.

### 4.5 Resiliency of Fallback Mechanisms

One of the fallacious paradigms floating around today's security practices is that effective backup systems need not be rigorously tested. Indeed, the very idea of a backup is temporary—i.e., a stopgap measure only intended to give the primary system enough breathing room to recover. The problem with this thinking is that poorly-secured backup systems can do more harm that good.

Consider the Blackout of 2038: The initial WASE vulnerability, though somewhat sophisticated, did not result in the catastrophic state of affairs following the DNP3 vulnerabilities. The attack was not at its worst when implemented in the first phase; the real damage resulted from the second phase.

What is more, although securing weak backup processes has received some attention in areas like OpenSSL (i.e., coercing a client to use an older version of TLS as opposed to a newer, more secure one) it is less common across the security field in general. Specifically, while NERC standardizes the availability of backup systems (see CIP-009 [11]), securing those systems is given scant attention. For example, CIP-009 rightly sets the standard for regularly practiced recovery plans. Unfortunately, it only demands that operators be able to maintain the system during backup [23, 59], a low bar that says little about how secure the system should be. Contrarily, if drills are executed on systems with known vulnerabilities, it should not be considered a success when a drill runs smoothly [61]. Given the right conditions, a "hidden menu" of vulnerabilities found in backup protocols is just as good as vulnerabilities found in primary protocols.

Here, again, LangSec could provide an answer. Researchers have demonstrated the effectiveness of protecting DNP3 legacy protocols several decades prior to our current state of affairs [3, 5]. Yet, as owed to the dearth of incentives, there is no attention, time, or finance given to these fallback systems.

### 4.6 Our Recommendations

Based on our analyses, we offer the following policy recommendations:

- As part of its CIP requirements, NERC should mandate the use of LangSec-compliant parsers. These parsers should be formally verified for memory safety and used for any primary or fallback protocols within power grid networks. The mandate may include penalties for utilities who remain non-compliant.
- All communication protocols used must include a machine-readable specification format. One possible candidate could be our XML-based specification format demonstrated in Section 4.3.
- To ease the burden on utilities, NERC should build and maintain tools offsetting the cost of ensuring security of backup protocols. For example, NERC could build and maintain a LangSec-compliant parser for the WASE protocol suite, and make it freely available to utilities and grid hardware providers (DNP3, IEC 61850, or others).
- Standards regarding recovery plans should not only require the availability and practice of recovery plans, but should also mandate the securing of backup protocols. Penetration testing is advised.
- Successful completion of a backup process must consider active adversaries during recovery mode. As the current example illustrates, a recovery plan involving DNP3 protocols should have identified DNP3's weaknesses.

Had such recommendations been followed in the past, we believe the Blackout of 2038 would have been averted. Following them henceforth will prevent similar incidents from occurring in the future.

### 4.7 LangSec: The Future

As discussed earlier, LangSec has made significant usability strides in recent years, resulting in Apron and NailGun. Secure parsers are now available for most popular file formats and streaming formats as a part of the DARPA SafeDocs program [16]. One extension to the SafeDocs program could be to design a way of providing specifications for various formats in a machine-readable format. The SafeDocs program required a lot of reverse-engineering of existing proprietary formats to be able to build secure parsers. We expect this to standardize into a format akin to XML reference format like the one we specified for the WSMP format. With increasing adoption of encrypted databases, web applications are now performing simple operations on web applications using homomorphic encryption [49]. Securely parsing input that is encrypted using homomorphic encryption is also an unexplored problem.

We also note LangSec is applicable to a variety of domains for which it is currently not utilized; however, realizing LangSec's potential requires a paradigm shift that begins with education. We must educate people about vulnerabilities arising from accepting bad input and the importance of validating input before processing it, teach them how to design protocols that avoid LangSec pitfalls, and show them how to use the tools LangSec provides to build secure parsers. These topics deserve coverage in computer science and computer security curricula. It would also be fruitful to cover LangSec into employee training for jobs involving protocol design, parser design, or, even coding in general. Additionally, as we have advocated for in the case of the smart grid, perhaps protocols and their implementations should be mandated to follow LangSec best practices.

While the Blackout of 2038 was absolutely devastating, it has provided us the opportunity to reflect. Just as the Texas Brownout of 2020 was instrumental in the adoption of formal verification to prevent race conditions, the Blackout of 2038 may provide the impetus to begin adopting LangSec both in the power grid and more broadly.

## 5 CONCLUSION

In the aftermath of the Blackout of 2038, many are left wondering what went wrong and what can be done to secure the grid. In this paper, we discussed these concerns by recounting recent grid developments, detailing the attack that led to the blackout, and advocating for solutions to secure the grid moving forward. In particular, LangSec provides a usable protocol design and implementation methodology that is key to hardening our nation's infrastructure. LangSec would not only obviate the Ping of Darkness, the primary vulnerability that led to the blackout, but it would also protect against a host of other input vulnerabilities that are prohibitively

costly to discover or address via other means. While we regret that LangSec was not utilized in the design and implementation of WASE, the Blackout of 2038 may provide an impetus to integrate LangSec into the next generation of industrial control system protocols.

## ACKNOWLEDGEMENTS

The authors would like to thank Ira Ray Jenkins and Rebecca (.bx) Shapiro for valuable discussions pertaining to the topic. We would also like to thank Professor Sergey Bratus for his insight and tutelage as a professor, mentor, and one of the founders of Language-theoretic Security, as well as Anna Shubina for insightful discussions pertaining to security and LangSec. Additionally, we would like to thank our shepherds Eireann Leverett and Steven Templeton, as well as anonymous reviewers for their valuable feedback on our paper.

This material is based upon work supported by the United States Air Force and DARPA under Contract No. FA8750-16-C-0179 and Department of Energy under Award Number DE-OE0000780.

Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Air Force, DARPA, United States Government or any agency thereof.

## REFERENCES

[1] Prashant Anantharaman, Michael Locasto, Gabriela F. Ciocarlie, and Ulf Lindqvist. 2017. Building Hardened Internet-of-Things Clients with Language-theoretic Security. In *Proceedings of IEEE Security and Privacy Workshops (SPW)*. IEEE, San Jose, CA, 120–126.

[2] Gretchen Bakke. 2016. *The Grid - The Fraying Wires Between Americans and Our Energy Future* (1 ed.). Bloomsbury USA, London, England. Chapter 6.

[3] Alex Balenson and George Specter. 2019. Securing Industrial Control System Protocols with Language-Theoretic Security. In *Proceedings of IEEE Symposium on Security and Privacy (Oakland)*. IEEE, San Francisco, CA, 42–52. ***(Future)***.

[4] Nicolas Blanc and Daniel Kroening. 2010. Race Analysis for SystemC Using Model Checking. *ACM Transactions Design Automation of Electronic Systems* 15, 3, Article 21 (June 2010), 32 pages. https://doi.org/10.1145/1754405.1754406

[5] Sergey Bratus, Adam J. Crain, Sven M. Hallberg, Daniel P. Hirsch, Meredith L. Patterson, Maxwell Koo, and Sean W. Smith. 2016. Implementing a vertically hardened DNP3 control stack for power applications. In *Proceedings of the 2nd Annual Industrial Control System Security Workshop*. ACM, Los Angeles, CA, 45–53.

[6] Kevin Bullis. 2009. Superconductors to Wire a Smarter Grid. *MIT Technology Review* (12 November 2009). https://www.technologyreview.com/s/416253/superconductors-to-wire-a-smarter-grid/

[7] Arthur Charguéraud and François Pottier. 2015. Machine-Checked Verification of the Correctness and Amortized Complexity of an Efficient Union-Find Implementation. In *International Conference on Interactive Theorem Proving*. Springer, Nanjing, China, 137–153.

[8] Sandra Chen. 2020. Now is the Time to Revolutionize the Grid. *IEEE Power and Energy Magazine* 18, 3 (2020), 17–23. ***(Future)***.

[9] Stephen Chong, Joshua Guttman, Anupam Datta, Andrew Myers, Benjamin Pierce, Patrick Schaumont, Tim Sherwood, and Nickolai Zeldovich. 2016. Report on the NSF Workshop on Formal Methods for Security. http://dl.acm.org/citation.cfm?id=3040225. (August 2016).

[10] Canadian Broadcasting Company. 2003. 2003: The great North America blackout. http://www.cbc.ca/archives/entry/2003-the-great-north-america-blackout. (August 2003).

[11] North American Electric Reliability Corporation. 2016. Critical Infrastructure Protection, CIP-009-6 — Cyber Security — Recovery Plans for BES Cyber Systems. https://www.nerc.com/_layouts/15/PrintStandard.aspx?standardnumber=CIP-009-6&title=CyberSecurity-RecoveryPlansforBESCyberSystems&jurisdiction=null. (2016).

[12] North American Electric Reliability Corporation. 2022. CIP 696-90. https://www.nerc.com/pa/Stand/Reliability0Standards/CIP-696-90.pdf. (2022). ***(Future)***.

[13] North American Electric Reliability Corporation. 2038. Cyber Kill Chain and the Ping of Darkness. https://www.nerc.com/ping-of-darkness. (2038). ***(Future)***.

[14] Adam Crain and Chris Sistrunk. 2014. Project Robus, Master Serial Killer. (January 2014).

[15] CVE-2013-2792 2013. CVE-2013-2792. Available from MITRE, CVE-ID CVE-2013-2792.. (3 December 2013). http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2013-2792

[16] DARPA. 2018. Restoring Trust in Electronic Documents. https://www.darpa.mil/news-events/2018-08-09. (2018).

[17] PJM State & Member Training Dept. 2015. Reactive Reserves and Generator D-Curves. (2015). https://www.pjm.com/-/media/training/nerc-certifications/gen-exam-materials/gof/20160104-reactive-reserves-and-d-curve.ashx?la=en

[18] Ross Deschamps and Thomas Avelar. 2020. Using Secure Conduits To Secure Legacy Programs. In *The 29th USENIX Security Symposium*. USENIX, Santa Clara, CA, 17–25. ***(Future)***.

[19] Ali Dorri, Salil S. Kanhere, Raja Jurdak, and Praveen Gauravaram. 2017. Blockchain for IoT security and privacy: The case study of a smart home. In *Proceedings of IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. IEEE, "Kona, HI", 618–623.

[20] Zakir Durumeric, James Kasten, David Adrian, J. Alex Halderman, Michael Bailey, Frank Li, Nicolas Weaver, Johanna Amann, Jethro Beekman, Mathias Payer, and Vern Paxson. 2014. The Matter of Heartbleed. In *Proceedings of Conference on Internet Measurement Conference*. ACM, Vancouver, BC, 475–488.

[21] E-ISAC and SANS. 2016. Analysis of the cyber attack on the Ukrainian power grid: Defense use case. https://ics.sans.org/media/E-ISAC_SANS_Ukraine_DUC_5.pdf. (March 2016).

[22] Dawson Engler and Ken Ashcraft. 2003. RacerX: effective, static detection of race conditions and deadlocks. In *Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles (SOSP*, Vol. 37. ACM, Bolton Landing, NY, 237–252.

[23] Division of Audits FERC, Office of Enforcement. 2013. Reliability Audit of Salt River Project Agricultural Improvement and Power District. https://www.ferc.gov/industries/electric/indus-act/reliability/reliability-orders/PA12-11-000.pdf. (July 2013).

[24] finance.ua. 2015. Хакери атакували «Прикарпаттяобленерго», знеструмивши половину регіону на 6 годин. http://news.finance.ua/ua/news/-/366136/hakery-atakuvaly-prykarpattyaoblenergo-znestrumyvshy-polovynu-regionu-\na-6-godyn. (December 2015).

[25] Patrick J. Flathers and William Nisen. 2038. 2038 Blackout Costs Projected to Fall Between $80 and $150 billion. *Nisen Economic Group* (1 April 2038). ***(Future)***.

[26] Pedro Fonseca, Kaiyuan Zhang, Xi Wang, and Arvind Krishnamurthy. 2017. An empirical study on the correctness of formally verified distributed systems. In *Proceedings of the Twelfth European Conference on Computer Systems*. ACM, Belgrade, Serbia, 328–343.

[27] U.S.-Canada Power System Outage Task Force. 2003. August 14, 2003 Outage Sequence of Events. https://www.ferc.gov/industries/electric/indus-act/reliability/blackout/09-12-03-blackout-sum.pdf. (September 2003).

[28] U.S.-Canada Power System Outage Task Force. 2004. Final Report on the August 14, 2003 Blackout in the United States and Canada: Causes and Recommendations. https://www.energy.gov/sites/prod/files/oeprod/DocumentsandMedia/BlackoutFinal-Web.pdf. (April 2004).

[29] GReAT. 2016. BlackEnergy APT Attacks in Ukraine employ spearphishing with Word documents. https://securelist.com/blackenergy-apt. (January 2016).

[30] Jason Grimm, Triple H, and Susan Smith. 2022. IEEE D17.215.2-2022 - IEEE Standard for the WASE Protocol Suite. *IEEE communications magazine* 60, 7 (2022), 85–96. ***(Future)***.

[31] Anthony Hall. 1990. Seven myths of formal methods. *IEEE software* 7, 5 (1990), 11–19.

[32] John Hawkins and Lien Susan. 2035. Assigning Trust Scores to Packets. In *Proceedings of the Thirtieth European Conference on Computer Systems*. ACM, Belgrade, Serbia, 328–343. ***(Future)***.

[33] Marisa Helms. 2013. The lessons of smart grid test in Boulder. https://finance-commerce.com/2013/04/the-lessons-of-smart-grid-test-in-boulder. (April 2013).

[34] Hermes Hogan, Richard Savage, and Luke Stevenson. 2022. NERC Mandates Adoption of WASE Protocol Suite. *IEEE spectrum* 59, 2 (2022), 13–19. ***(Future)***.

[35] PJM Interconnection. 2017. Reactive Power. https://www.pjm.com/-/media/about-pjm/newsroom/fact-sheets/reactive-power-fact-sheet.ashx. (April 2017).

[36] Dan Kaminsky, Meredith L. Patterson, and Len Sassaman. 2010. PKI Layer Cake: New Collision Attacks Against the Global X.509 Infrastructure. In *Proceedings of the 14th International Conference on Financial Cryptography and Data Security*. Springer, Tenerife, Canary Islands, Spain, 289–303.

[37] Adam Kaprowski and Henri Binsztok. 2010. TRX: A Formally Verified Parser Interpreter. In *Proceedings of the 19th European conference on Programming (ESOP)*. Springer, Paphos, Cyprus, 345–365.

[38] Matt Kaufmann, Panagiotis Manolios, and J Strother Moore. 2013. *Computer-aided reasoning: ACL2 case studies*. Vol. 4. Springer Science & Business Media.

[39] Matt Kaufmann and J. Strother Moore. 2017. A Flying Tour of ACL2. http://www.cs.utexas.edu/users/moore/acl2/v8-1/combined-manual/index.html?topic=ACL2____A_02Flying_02Tour_02of_02ACL2. (2017).

[40] Cruz Kyle and Anderson Timothy. 2023. NailGun: Automatic Generation of Hammer Parser combinators. In *Proceedings of IEEE Security and Privacy Workshops (SPW)*. IEEE, San Jose, CA, 89–100. ***(Future)***.

[41] Leslie Lamport. 1977. Proving the Correctness of Multiprocess Programs. *IEEE Transactions on Software Engineering* 3, 2 (1977), 125–143.

[42] Jun Lin, Zhiqi Shen, and Chunyan Miao. 2017. Using Blockchain Technology to Build Trust in Sharing LoRaWAN IoT. In *Proceedings of the 2nd International Conference on Crowd Science and Engineering*. ACM, Beijing, China, 38–43.

[43] X. Liu, P. Zhu, Y. Zhang, and K. Chen. 2015. A Collaborative Intrusion Detection Mechanism Against False Data Injection Attack in Advanced Metering Infrastructure. *IEEE Transactions on Smart Grid* 6, 5 (September 2015), 2435–2443. https://doi.org/10.1109/TSG.2015.2418280

[44] Yao Liu, Peng Ning, and Michael K. Reiter. 2011. False Data Injection Attacks Against State Estimation in Electric Power Grids. *ACM Transactions on Information and Systems Security* 14, 1, Article 13 (June 2011), 33 pages. https://doi.org/10.1145/1952982.1952995

[45] Nuno Lopes, Nikolaj Bjørner, Nick McKeown, Andrey Rybalchenko, Dan Talayco, and George Varghese. 2016. *Automatically verifying reachability and well-formedness in P4 Networks.* Technical Report. Technical Report.

[46] Falcon Darkstar Momot, Sergey Bratus, Sven M. Hallberg, and Meredith L. Patterson. 2016. The Seven Turrets of Babel: A Taxonomy of LangSec Errors and How to Expunge Them. In *Proceedings of IEEE Cybersecurity Development Conference*. IEEE, Boston, MA, 45–52.

[47] Ukraine Ministry of Energy. 2016. Міненерговугілля має намір утворити групу за участю представників усіх енергетичних компаній, що входять до сфери управління Міністерства, для вивчення можливостей щодо запобігання несанкціонованому втручанню в роботу енергомереж. http://mpe.kmu.gov.ua/minugol/control/uk/publish/article?art_id=245086886&cat_id=35109. (February 2016).

[48] Department of Homeland Security. 2016. Cyber-Attack Against Ukrainian Critical Infrastructure— Alert (IR-ALERT-H-16-056-01). https://ics-cert.us-cert.gov/alerts/IR-ALERT-H-16-056-01. (February 2016).

[49] Raluca Ada Popa, Emily Stark, Jonas Helfer, Steven Valdez, Nickolai Zeldovich, M Frans Kaashoek, and Hari Balakrishnan. 2014. Building Web Applications on Top of Encrypted Data Using Mylar.. In *NSDI*. 157–172.

[50] Kevin Poulsen. 2004. Tracking the blackout bug. https://www.securityfocus.com/news/8412. (April 2004).

[51] Jonathan Protzenko, Jean-Karim Zinzindohoué, Aseem Rastogi, Tahina Ramananandro, Peng Wang, Santiago Zanella-Béguelin, Antoine Delignat-Lavaud, Cătălin Hriţcu, Karthikeyan Bhargavan, Cédric Fournet, et al. 2017. Verified low-level programming embedded in F. *Proceedings of the ACM on Programming Languages* 1, ICFP (2017), 17.

[52] Roberta Rampton and David Ljunggren. 2016. 'Three Amigos' aim for more clean power by 2025: White House. https://www.reuters.com/article/us-usa-canada-mexico-energy-idUSKCN0ZD2L0. (27 June 2016).

[53] Karen Robben and John Stevens. 2025. Creating a Taxonomy of Bugs in Modern CVEs. In *Proceedings of IEEE Security and Privacy Workshops (SPW)*. IEEE, San Jose, CA, 117–127. ***(Future)***.

[54] Len Sassaman, Meredith L. Patterson, Sergey Bratus, Michael E. Locasto, and Anna Shubina. 2011. *Security Applications of Formal Language Theory.* Technical Report. Dartmouth College, Hanover, NH.

[55] Sean Smith and John Marchesini. 2007. *The Craft of System Security.* Pearson Education, Boston, MA. 391–409 pages.

[56] Donna Stones and Arya Wiens. 2021. Apron: A simple frontend for Hammer parsers. https://www.cs.dartmouth.edu/~apron. (May 2021). ***(Future)***.

[57] Adam Vincent. 2018. BlackEnergy Malware: How Hackers May Tackle our Infrastructure. https://www.infosecurity-magazine.com/opinions/blackenergy-malware-infrastructure. (February 2018).

[58] Alexey Vinel. 2012. 3GPP LTE versus IEEE 802.11 p/WAVE: Which technology is able to support cooperative vehicular safety applications? *IEEE Wireless Communications Letters* 1 (2012), 125–128.

[59] LLP Vinson & Elkins. 2014. Summary of CIP Version 5 Standards. https://www.velaw.com/uploadedfiles/vesite/resources/summarycipversion5standards2014.pdf. (2014).

[60] Dustin Volz. 2016. U.S. government concludes cyber attack caused Ukraine power outage. https://www.reuters.com/article/us-ukraine-cybersecurity-idUSKCN0VY30K. (25 February 2016).

[61] LLP White Case. 2011. NERC Case Notes: Reliability Standard CIP-009-1. https://www.whitecase.com/nerc-cip-009-1. (2011).

[62] Jeannette M. Wing. 1998. A symbiotic relationship between formal methods and security. In *Proceedings of IEEE Conference on Computer Security, Dependability and Assurance: From Needs to Solutions*. IEEE, Williamsburg, VA, 26–38.