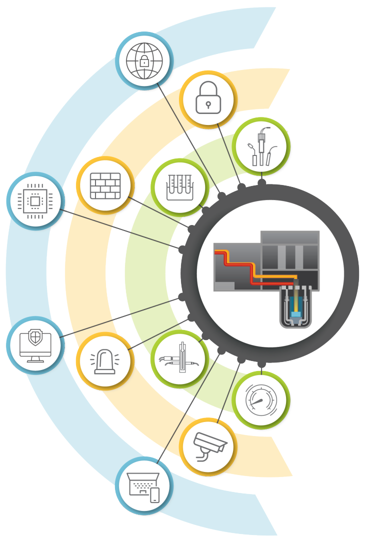

ARCADE
Analysis Methods & Validation Pathway

Prepared for
US Department of Energy

Andrew S. Hahn, Adam Beauchaine, Titus Gray,
Noel Belcourt, Fraser Dougall

Sandia National Laboratories

September 2025
SAND2025-12807R

Prepared by Sandia National Laboratories, Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of energy's National Nuclear Security Administration under contract DE-NA0003525.

Sandia National Laboratories

ABSTRACT

The Advanced Reactor Cyber Analysis and Development Environment (ARCADE) provides an automated analysis system which supports risk-informed performance based (RIPB) evaluations of nuclear control systems. Every possible cyber threat which could lead to consequence is identified by simulating the unsafe control action sequences which transform digital harm into physical harm. Eliminating the simulation of complex digital cyber attack chains cuts out unnecessary computational overhead and focuses directly on the physics of cyber-physical attacks. This focus enables designers to make informed decisions which can entirely eliminate categories of cyber threats against advanced reactors through the physical nature of the plant design.

This narrowing of cyber threat against nuclear power plants through the physics of the system is intended to make any remaining threat management and cost efficient. This is the goal of the Tiered Cyber Analysis (TCA) outlined in NRC Draft Regulation Guide (RG) 5.96, which provides a RIPB cybersecurity approach for new reactors. ARCADE has been custom developed to meet the demands of the rigorous analysis required in Tier 1 of the TCA, which forms the foundation of the TCA process. Currently, ARCADE is still under development, but has made significant leaps in capability. A pilot analysis on the opensource Asherah simulator was performed which demonstrated key functionality goals. The next stage of ARCADE development involves improvements to the applications which support the analysis system, and enabling the analysis system to utilize the full suite of unsafe control action simulations. Since the analysis method's core functions are complete, validation of the analysis method will be started concurrent to the next development stages.

The automated analysis ARCADE will provide can radically change the cybersecurity design process for advanced reactors, reducing the cost of security implementation while enhancing cyber resilience. The pathway for ARCADE's development to this goal has become much clearer. The majority of technical hurdles have been cleared, and the remaining development needs have been solidified. ARCADE is now capable of assisting the advanced reactor design process and directly support advanced reactor industry RIPB practices.

This page intentionally left blank.

ACKNOWLEDGEMENT

The authors would like to acknowledge the DOE Office of Nuclear Energy Advanced Reactor Safeguards and Security (ARSS) program for funding this critical work providing the nuclear industry secure-by-design tools.

This page intentionally left blank.

CONTENTS

This page intentionally left blank.

LIST OF FIGURES

This page intentionally left blank.

LIST OF TABLES

This page intentionally left blank.

ACRONYMS & DEFINITIONS

AR	Advanced Reactor
ARCADE	Advanced Reactor Cyber Analysis and Development Environment
CDA	Critical Digital Asset
DCS	Distributed Control System
DCSA	Defensive Cyber Security Architecture
ETE	Equipment Test Environment
HiL	Hardware in the Loop
ICS	Industrial Control System
I/O	Input/Output
JSON	JavaScript Object Notation
NPP	Nuclear Power Plant
NRC	Nuclear Regulatory Commission
OT	Operational Technology
PLC	Programmable Logic Controller
PROMISE	Power Reactors on Minimega In Service of Education
RG	Regulation Guide
RIPB	Risk-Informed Performance Based
SeBD	Secure By Design
SLDA	System-Level Design Analysis
STAMP	System-Theoretic Accident Model and Processes
STPA	Systems Theoretic Process Analysis
TANGO	Twinned Analysis of Nuclear Reactor and Grid Operations
TCA	Tiered Cyber Analysis
UCA	Unsafe Control Action

This page intentionally left blank.

1. INTRODUCTION

As cybersecurity threats against critical infrastructure increase, it is imperative that nuclear power plants are designed to be inherently resilient to cyber attacks. This robustness cannot come at the expense of greater construction or operation costs as this risks the economic feasibility of nuclear power. Ideally, designing in this cyber resilience would not create additional engineering costs. Through conventional cybersecurity design methods these requirements are impossible. Utilizing the Advanced Reactor Cyber Analysis and Development Environment (ARCADE) to perform a Tiered Cyber Analysis (TCA) can produce risk-informed, performance-based (RIPB) cyber secure designs that are economically sound. The previous publication on ARCADE, [4] details how ARCADE can be used to dramatically increase cyber resilience while reducing cybersecurity implementation costs in construction and for the lifetime of a nuclear power plant via the TCA method.

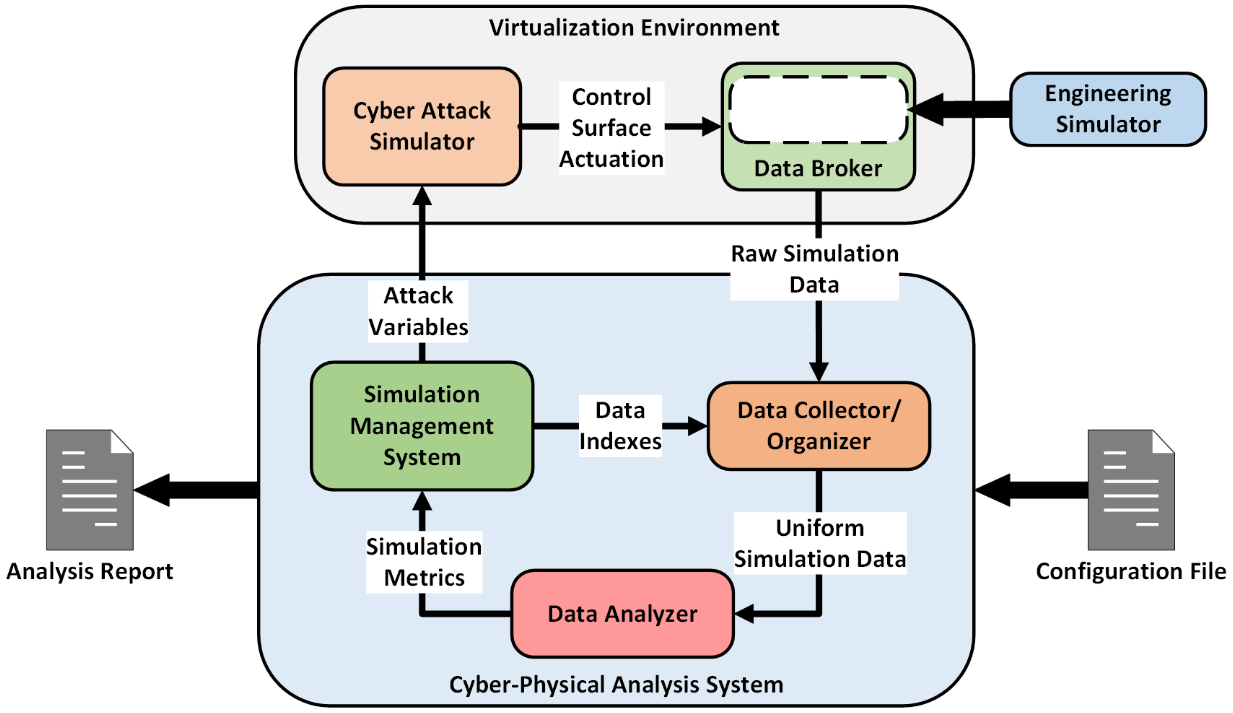

Figure 1-1. ARCADE Overview

ARCADE is still being developed, but has recently made significant progress towards enabling a full TCA Tier 1 analysis, which has been out of reach with conventional analysis tools. Utilizing the Asherah nuclear power plant (NPP) simulator as an test subject [7], a limited Tier 1 analysis was performed which will be discussed in section 1.1. This represents a significant leap in ARCADE's capabilities and opens the door to begin limited fault analysis on real control system designs. To complete a full Tier 1 analysis ARCADE still needs to complete some development steps and validate its analysis method. This report will detail how ARCADE is to perform its analysis, the pathway to validate the analysis method, and finally what components in ARCADE must be developed to become production ready.

1.1. ARCADE Pilot Analysis

Asherah was selected as a test subject because its data is non-sensitive and can be readily published. Through it is a PWR simulation, it is representative of the same simulators used in the industry to design advanced reactor control systems. The analysis presented can be done on any advanced reactor engineering simulator, but for demonstration purposes this opensource simulation was necessary.

First, Asherah NPP simulator was integrated with the Data Broker and encapsulated in a virtual machine which could be managed by the simulation management system and interfaced with by the cyber attack simulator (Figure 1-1). An analysis system configuration file was developed which defined the mechanical limits of each actuator and the safety boundaries of the NPP. A small 32 core server performed simulations for approximately 48 hours to process all the 2 UCA combinations for the "Provided Not Needed" U1 category. A snippet of the raw results out of ARCADE are shown below.

```
combined attacks (46, 69) SG2_Press value 14456702.180489(Pa) >= safe upper
    limit 8974000.0(Pa) occurred at 768.0(s) (CE_Pump1OnOffCmd U1-U) (
    TB_BypassValveCmd U1-U)
combined attacks (46, 69) CD_Press value 7768.263704(Pa) >= safe upper limit
    6760.0(Pa) occurred at 720.3(s) (CE_Pump1OnOffCmd U1-U) (TB_BypassValveCmd
     U1-U)
combined attacks (46, 69) SG1_Level value 8.198232(m) <= safe lower limit
    12.0(m) occurred at 823.6(s) (CE_Pump1OnOffCmd U1-U) (TB_BypassValveCmd U1
    -U)
combined attacks (46, 69) RX_Tavg value 680.2422730000001(K) >= safe upper
    limit 580.0(K) occurred at 717.0(s) (CE_Pump1OnOffCmd U1-U) (
    TB_BypassValveCmd U1-U)
combined attacks (46, 70) CD_Level value 1.606579(m) >= safe upper limit 1.5(m
    ) occurred at 765.7(s) (CE_Pump1OnOffCmd U1-U) (INT_RXPowerSetpoint U1-U)
combined attacks (46, 70) RX_DT value 76.91692(K) >= safe upper limit 40.0(K)
    occurred at 709.2(s) (CE_Pump1OnOffCmd U1-U) (INT_RXPowerSetpoint U1-U)
```

Listing 1.1 ARCADE Raw Output

These results can be graphically represented by tallying how many times a control surface was part of a UCA sequence which resulted in a consequence. Figure 1-2 shows a heat map of all the control surfaces in Asherah, with the coincidence of each included in a UCA sequence with consequences. It becomes apparent which sections of the plant are most significant from a safety perspective. These are the components which an adversary is most likely to attack and thus are the most important to consider for cybersecurity. Now an engineer has a new data perspective to make decisions about their design. They could for example, investigate the Turbine Speed Control Valve (TB-SpeedCtrlValveCmd) and redesign it to make it impossible for the adversary to use it in an attack. Hypothetically if the valve was always closed in an attack sequence to cause a high pressure event, a pressure relief mechanism would prevent that attack pathway entirely.

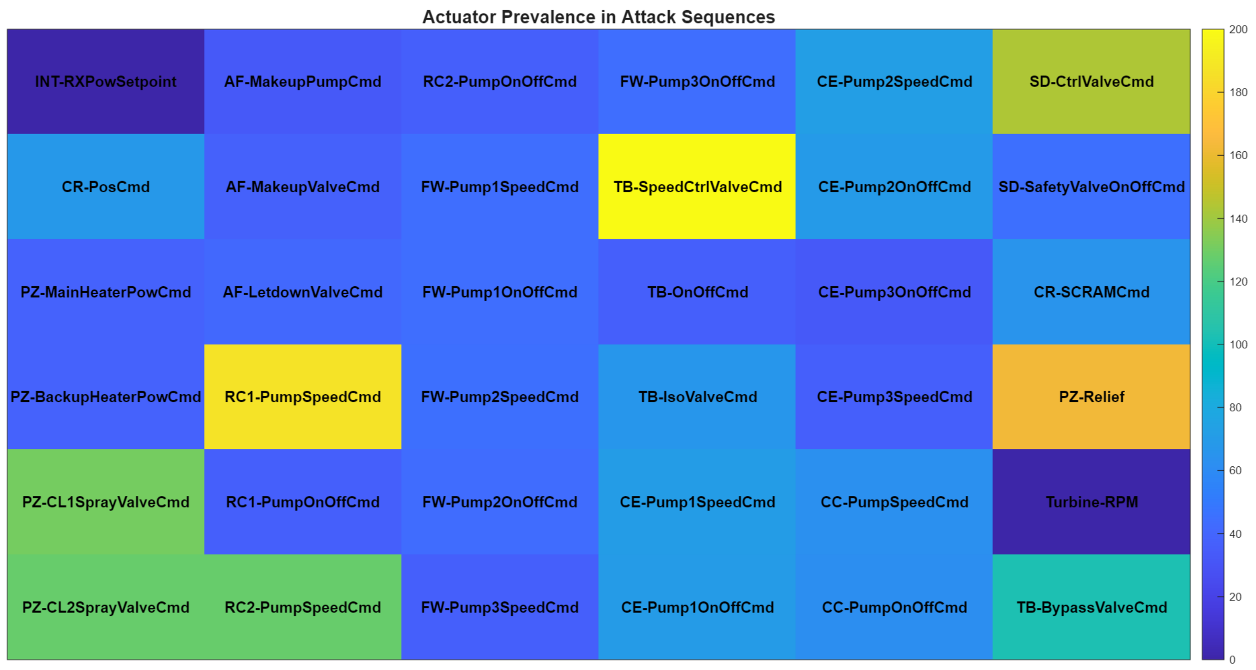

Figure 1-2. ARCADE Asherah analysis results, all failure modes.

This is only the surface level information that can be gleaned from the ARCADE analysis, as Figure 1-2 is counting all the potential failures in the plant. A regulator might only care about fuel failures, and this information can be extracted from the ARCADE data. Figure 1-3 shows only the reactor failures, corresponding to fuel temperature exceeding safe limits. Now the Turbine Speed Control Valve no longer seems as significant to Asherah's safety design, and it becomes clear which components are most important to this PWR's safety. Rather expectedly, the reactor coolant pumps and the pressurizer are the most critical to the safety of this PWR. Perhaps more important is what is shown to be not critical to safety, pressurizer heaters don't seem to affect plant safety all that much. Effectively nothing on the tertiary side like the condensers has any effect, and the secondary side has very limited effect to the primary side safety.

Understanding what doesn't contribute to cyber risk is just as important as understanding what does. Those areas of the graph which do not significantly contribute to risk do not require the same degree of cybersecurity focus. This is the key cost saving proposal of ARCADE. If it can be definitively determined that a given component has no links to severe safety threats, it's security level or profile can be reduced or eliminated. This risk grading can accurately focus cybersecurity efforts to only those areas of the control system which have consequence and save costs on areas which do not.

ARCADE's impact does not end at quantifying the risk from a regulatory perspective, asset owners may also be concerned about continuity of operations. The flexibility allows ARCADE to investigate threats which could endanger the plant economically. A condenser system failure may not pose a significant radiological threat, but it could stop a plant from producing power for a significant time

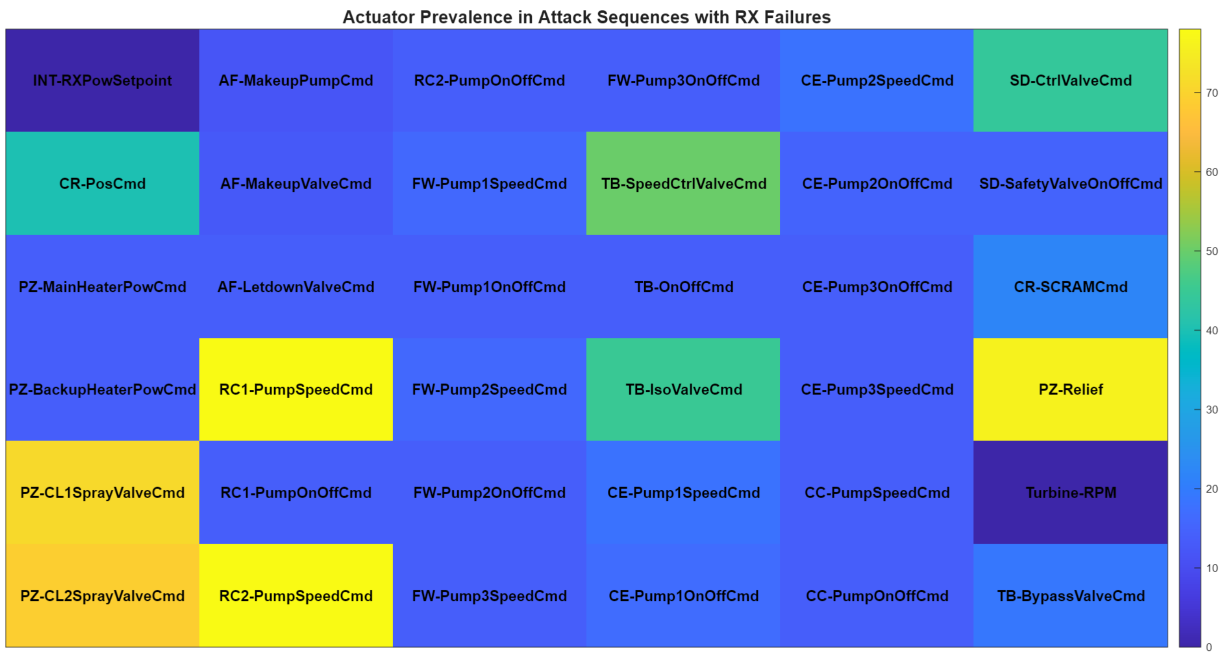

Figure 1-3. ARCADE Asherah analysis results, reactor failure modes only.

depending on the damage caused. The analysis mode in Figure 1-4 has been switched to focus on only condenser failures. This risk analysis highlights areas of concern which are unique to the condenser and can be addressed at the discretion of the asset owner. This new visibility into the cyber threat profile of the NPP gives designers and owners more decision making power managing cyber risk.

These figures display only one dimension of the information produced by ARCADE, there is a wealth of information which automated data processing pipelines are being built. The severity of the consequence for each UCA sequence has not been processed into graphical representations, nor has the time between initiation and consequence. The reaction time plant staff would have to mitigate a severe UCA sequence consequence is a critical risk dimension which ARCADE can determine. As ARCADE advances, the quantification of these risk dimensions will become easier to visualize and assist the designers of advanced reactors build in inherent cyber resilience.

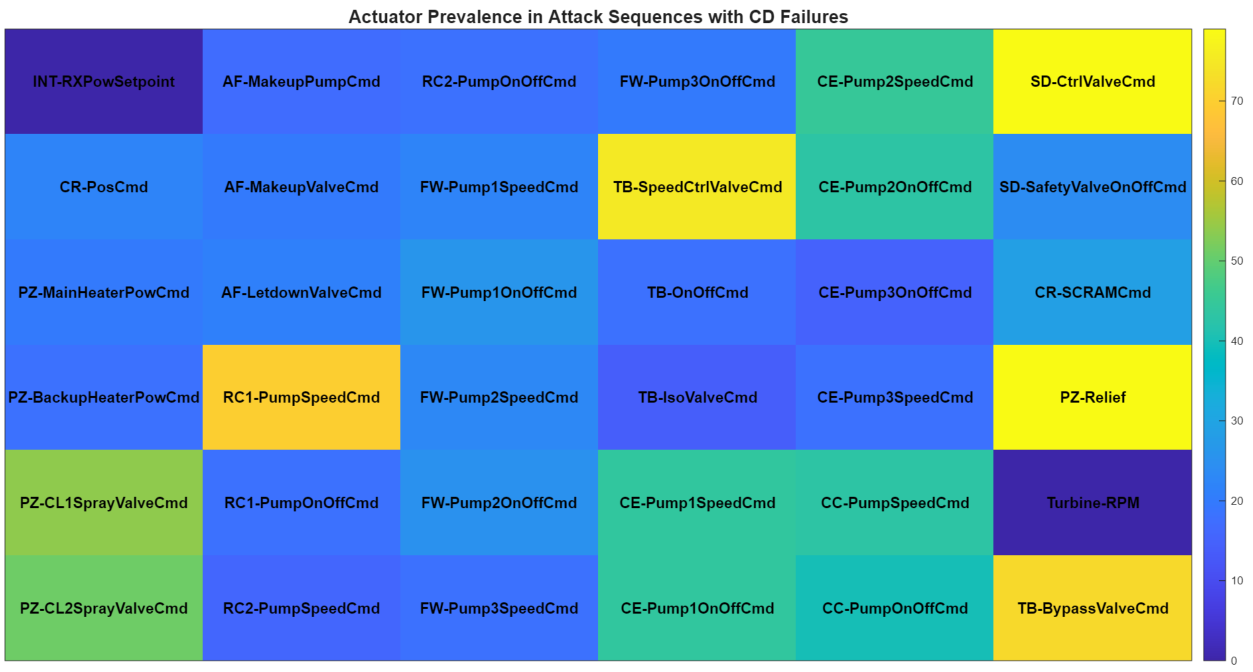

Figure 1-4. ARCADE Asherah analysis results, condenser failure modes only.

This page intentionally left blank.

2. ANALYSIS METHOD

The ARCADE analysis method required the removal of the human element of conventional fault analysis to automate the entire process. This limits human bias to the input values which define the analysis parameters and eliminates the thousands of engineer hours required to develop a fault analysis. When considering sequences of UCAs, the automated approach can reach combination depths which are economically unfeasible with conventional methods. This section documents the analysis concept and the technical method utilized to analyze the control system designs of advanced reactors and produce a report of all the sequences of UCAs which cause consequence.

2.1. Analysis Concept

Figure 2-1 provides an overview of the main loop of the analysis engine. The analysis system sweeps through all the categories of UCAs beginning with the U1 category which provides the initial transient conditions necessary to evaluate the rest of the UCA categories listed below.

- [U1] Command provided but not needed
- [U2] Command needed but not provided
- [U3] Command provided too early, too late, out of order
- [U4] Command stopped too soon, applied too long

The analysis runs adds these initial simulations to the processing queue, $/mathcalQ$, which are processed when computation availability opens. The UCA scenarios are simulated, represented by the transform $\text{Pop}(s, pi)$. The data from the simulations is evaluated to determine if they have exceeded any accident conditions which were set by the user. If the UCA scenario results in an accident condition, its information is recorded and it is dropped as a seed for subsequent scenarios. If a scenario results in accident conditions, it is not worth the computation to continue to add UCAs to a scenario that already results in a consequence.

If we consider the first round of U1 UCAs to be round 0, then round 1 would be the first opportunity to evaluate U2-U4 UCAs which required that initial transient. The system will be selective for which U2-U4 UCAs it will test, it will only evaluate these on actuators which change their position based on the previous transient. If the position of the actuator or control surface did not change, then a UCA like "Needed Not Provided" doesn't have an opportunity to effect the process. Any identified potential combinations of merit will be added to the simulation queue.

Following the first complete sweep of all the categories of UCAs, the system will start alternating between building combinations based on predictions and combinations of U2-U4 UCAs based on control surface movement. This allows complex UCA sequences to be rapidly evaluated, while

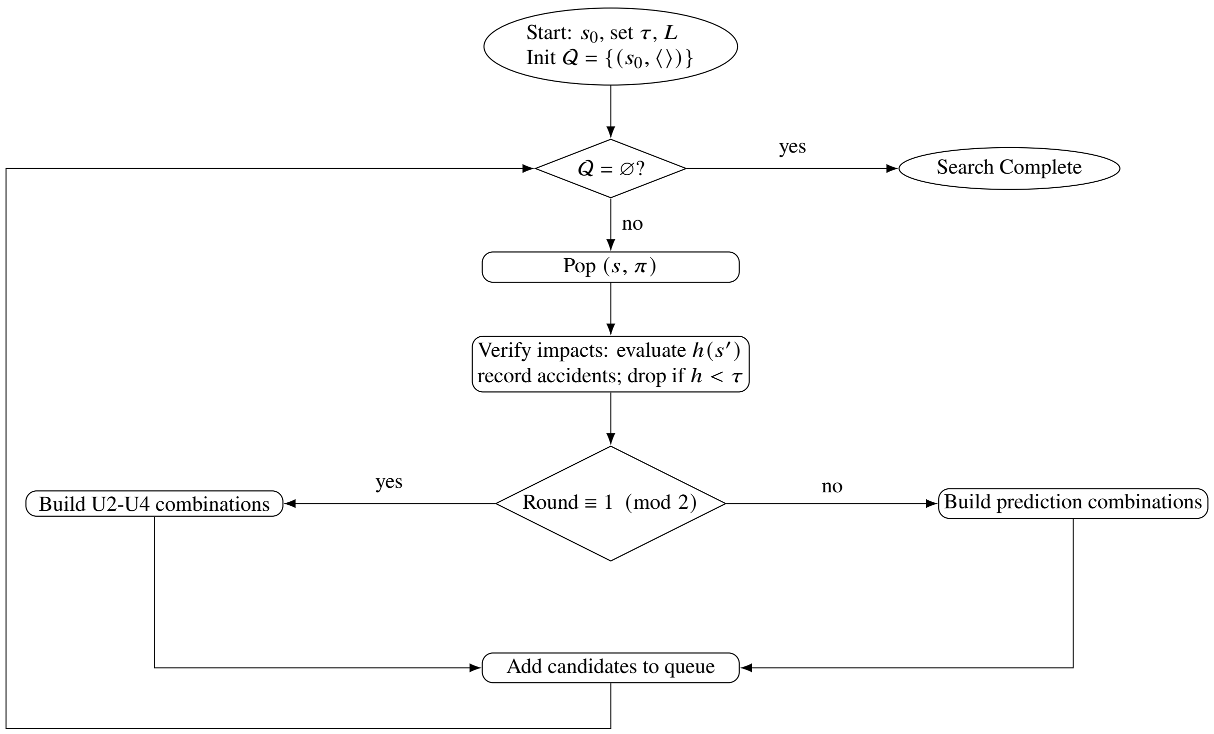
Start: s_0, set τ, L
Init $Q = \{(s_0, \langle\rangle)\}$
$Q = \varnothing$?
yes
Search Complete
no
Pop (s, π)
Verify impacts: evaluate $h(s')$
record accidents; drop if $h < \tau$
Round $\equiv 1 \pmod 2$
yes
Build U2-U4 combinations
no
Build prediction combinations
Add candidates to queue

Figure 2-1. Flowchart of the ARCADE Analysis Process

also ensuring that any new actuator perturbations are maximally exploited. This will ensure that the analysis can attain reasonable completeness. This process continues until the user requested search depth is reached.

2.2. Technical Implementation

Implementation of the analysis concept is limited to the use of the U1 category, but the core structure for achieving the complete analysis can be still be constructed and exercised. When the other UCA category simulations are complete, this implementation will accommodate inclusion of these new threat search vectors. This section details the technical implementation of the analysis concept.

The ARCADE data analysis methodology can be partitioned into four main steps, 0) generate and run initial UCAs, 1) analyze existing UCA simulation results to check for consequences, 2) generate new UCA combinations for the next search depth, and 3) predict whether new UCA combinations will result in consequence for the next search depth. The input deck has a description of both the control surfaces and accident conditions which defines the safe operational envelop and the dimensions of the control space to be searched.

The number of initial generated UCAs to run is calculated from the input file as follows:

```
num_attacks = num_control_surfaces * 2 + 1
attack_ids = list from [0, num_attacks)
```

Listing 2.1 ARCADE inital UCAs

Two UCA simulations are generated for each control surface, one for U1-L and one for U1-U. The additional simulation is for the nominal base case. For brevity in the code, these UCA sequences are refereed to as attacks, since they are simulations of the physical effects of cyber attacks.

Its assumed that the data to be analyzed is stored in a rank 3 array in C memory layout dimensioned by num_accident_conditions, num_attacks, num_timesteps. The rank 3 data array is as follows:

```
data[num_accident_conditions, num_attacks, num_timesteps]
```

Listing 2.2 ARCADE Data Array

The results are loaded into this data array when the runs have all completed. The first step in the ARCADE analysis is to analyze the existing results. The pseudo code in Listing 2.3 describes the operation of this analysis. Each user defined accident condition is loaded and the time series for each run are compared to the conditions defined. If the conditions are met, the system marks that run as having exceeded an accident condition and the simulation time and value of the data point are recorded.

```
for each accident condition:
  op := get comparison operator for this accident condition
  safe_limit := get the limiting value for this accident condition
  for each attack:
    if op is '<=':
      # find minimum value over all time steps
      val := minval(data[accident_condition,attack,:])
      if val <= safe_limit:
        this attack caused an accident for this accident condition
        timestep := get time step where minimum value occurred
        append attack number to list of unsafe attacks
    else if op is '>=':
      # find maximum value over all time steps
      val := maxval(data[accident_condition,attack,:])
      if safe_limit <= val:
        this attack caused an accident for this accident condition
        timestep := get time step where maximum value occurred
        append attack number to list of unsafe attacks
```

Listing 2.3 ARCADE Analyze Results

When the list of unsafe attacks is generated, there can be duplicates in the list since a single attack could fail for multiple accident conditions. These duplicates are recorded for reporting to the user, but processing the next generation of attacks requires the list to be sorted and duplicates pruned. The code below shows a standard Python idiom for achieving this cleanup. A set is a unique associative container whose key and value are the same. A sorted list is then created from the results.

```
unsafe_attacks = sorted(list(set(unsafe_attacks)))
```

Listing 2.4 Remove Duplicate Attacks

Next the unsafe attacks are removed from the list of all attacks. The remaining list of safe attacks is used to generate new combinations of attacks to test in the next search.

```
num_attacks -= num_unsafe_attacks
for each unsafe_attack:
  remove unsafe attack from list of attacks
```

Listing 2.5 Remove Unsafe Attacks

Next the new attack combinations from the remaining valid attacks in the current search depth are generated. For creating the second round of simulations, a simple quadratic algorithm is used to create the attacks. The function same_control_surface reports true if i and j refer to the same control surface.

```
new_combinations = []
for each i in range(num_attacks):
  for each j in range(i+1, num_attacks):
    if not same_control_surface(i, j):
      tuple = (i,) + (j,)
      new_combinations.append(tuple)
```

Listing 2.6 Generating New Attacks

The new combinations array is a list of pairs of attack ids. This list is a complete set of new combinations meaning no attack pairs are skipped unless one of the attacks in the pair caused an accident. This generates all possible pairs except those that caused accidents.

The prediction algorithm for generating new attack scenarios uses the results for each individual attack generated in search depth 1. The analysis system computes the perturbation from nominal for each attack in each combination, sums the perturbations and nominal values, then, as above, computes the min or max and checks if the new attack combination could potentially result in accident conditions.

```
for each accident condition:
  # get nominal (basecase) data for all timesteps
  nominal := data[accident_condition, 0, :]
  op := get comparison operator for this accident condition
  safe_limit := get the limiting value for this accident condition
  for each pair in new_combinations:
    perturbations = []
    # compute perturbation from nominal over all timesteps and sum
    for each attack in pair:
      perturbations += data[accident_condition,attack,:] - nominal
    # compute resulting total perturbation
    result := perturbations + nominal
    if op is '<=':
      # get minimum value from perturbation result
      val := minval(result)
      if val <= safe_limit:
        this attack predicted to cause an accident for this accident condition
        timestep := get time step where minimum perturbation occurred
        diagnostic indicating we predict this attack pair may fail when run
    else if op is '>=':
      # get maximum value from perturbation result
      val := maxval(result)
      if safe_limit <= val:
        this attack predicted to cause an accident for this accident condition
        timestep := get time step where maximum perturbation occurred
        diagnostic indicating we predict this attack pair may fail when run
```

Listing 2.7 Pertubation Generation

The attack combinations which result in potential accident conditions are selected to be simulated to confirm their consequence. This process repeats until the search depth is reached. For the current implementation which can only evaluate U1 UCAs, there is no alternation to U2-U3 category searches for every other round of UCA combination builds.

2.3. Initial Asherah Evaluation

Utilizing the limited U1 analysis system, ARCADE evaluated Asherah to a search depth of 2 seeking out UCA combinations on the PWR simulator. Asherah has 35 control surfaces which were evaluated, which results in a round 1 search of 70 U1 UCA's consisting of lower and upper bound U1 evaluations. 17 of these UCAs resulted in consequence which would indicate they would be of the highest concern. The second generation of resulted in 1358 possible 2 U1 UCA combinations, when evaluated 493 were found to cause consequence. This second round also informs the division of the control system network, as separating the components in these 493 UCA combinations would passively inhibit threats.

Table 2-1. Asherah UCA Evaluations

	UCAs Searched	Consequential UCAs
Round 1	70	17
Round 2	1,358	493

A total of 510 UCAs and UCA combinations out of a total of 1428 were identified out of in these 2 rounds of the search. This reduced the threat scenarios worth considering by nearly 1/3 which can still be considered an excessive number of consequential UCA scenarios. Asherah being an older conventional PWR makes it far less resilient, and the reduced complexity of the model implies that the majority of modeled control surfaces will be essential to operating the plant safely. These factors contribute to the model having a high ratio of consequential to unremarkable UCAs.

An advanced reactor would be expected to have a smaller ratio of consequential UCAs as they designed with more inherently and passively safe features. For example, if the reactor coolant pump in Asherah is shut down completely at full reactor power, this results in a significant fuel temperature rise and fuel damage risk. If an HTGR with TRISO fuel has its Helium blower shutdown at full power operation, its reasonable to assume that the negative temperature reactivity coefficient will shut down the reactor well before the fuel integrity is compromised. For these inherently safer advanced reactors, it is expected that the analysis will identify less numerous consequential UCAs, but produce the evidence necessary to reduce cybersecurity requirements for systems which do not impact safety.

This page intentionally left blank.

3. VALIDATION PATHWAY

This section details the formal modeling of ARCADE as a discrete-time simulation system, and discusses the simulation approach with respect to existing system models. It details the heuristic search technique employed by ARCADE and contrasts the approach with existing work in the control systems analysis space. Finally, it provides a measure of bounded correctness for the analysis approach with respect to a NPP system.

3.1. Problem Formalization

ARCADE formalizes the simulation environment as a synchronous, discrete-time system achieved through the simulation of all physical plant components. Given the challenges of performing analysis on real-time systems that include non-trivial physical simulations, ARCADE simplifies the problem search space by overlaying discrete time elements on a plant simulation environment. When interfacing with physical simulation engines, which model continuous time in most cases, ARCADE discretizes continuous time $t \in \mathbf{R}_{\geq 0}$ into timesteps of length $\triangle t$, on which all observation and actuation events occur. In some cases, the simulation engine may already operate on a similar discrete time model. In such cases, ARCADE may define $\triangle t$ as the time unit leveraged by the simulation engine. In all cases, ARCADE interfaces with the physical simulation via simulated logic controller data. Any physical simulation elements are fully abstracted from the view of the simulation controller.

Within this model, ARCADE may interface with a system using the standard view of Lee et al. [5]. In this model, a system Σ is a mapping from a starting state X_0 and input signals $U \in U^T$ to and output signal Y^T. The input of the system analyzed by ARCADE is equivalent to the set of all control registers corresponding to plant actuation elements. This input may be selected to be a maximally inclusive as possible or streamlined depending on analysis goals. For each register $\mathbf{r}_1, ..., \mathbf{r}_n$, the system must define a finite input domain $\mathbf{D}$. At each timestep, the system issues an actuation vector $a \in \mathbf{A} := \mathbf{D}_1 ... \mathbf{D}_n$ corresponding to each control register. The attack simulations in ARCADE are based on the perturbation of this actuation vector, and the subsequent impact surface on the observable output signal Y^T. In a similar manner to the input domain, ARCADE defines this output domain as the set of all observable sensor values after a given input timestep $o \in \mathbf{O} := \mathbf{D}_1 ... \mathbf{D}_n$.

All simulation runs of ARCADE are conducted over a set time interval N, which may be customized based on the needs of the system under test. In the case of systems that may not show the impacts of a perturbation immediately, N may be increased to allow for the results of a perturbation to be recorded successfully. Regardless of simulation duration, ARCADE output is defined as the distance of the run to the system accident space. The accident space is defined as a subset

$A \in \mathbb{R}^o$, where $\mathbb{R}^o$ is the output space produced by the simulator. The accident space A is of the same dimensionality of the output vector, and encodes all observation patterns corresponding to accidents (e.g. under/overpressure hazards, actuation failure hazards, temperature-based hazards.)

The formal structure and accident space of ARCADE simulations allow for detailed understanding of perturbation impacts and serve as the grounding for the spacial exploration strategy leveraged by ARCADE. This strategy is subsequently detailed in section 3.2.

3.2. Heuristic Search Approach

To search for accident conditions, ARCADE leverages a greedy heuristic-based search algorithm, that combines the mechanics of STPA and UCA categories to search a system for common hazards. Through an informed trajectory search of the system, ARCADE is capable of discovering novel accident sequences significantly faster than existing techniques such as Monte Carlo and Ant Colony Optimization.

Prior work in this vein has sought to perform trajectory analysis leveraging similar statistical methods. Liu et al. [1] deploy stochastic optimization techniques across MATLAB® simulations to discover minimal robustness paths across a system. Donzé et al. [3] perform investigation of large continuous space trajectory data using differential equation modeling for parameter optimization. ARCADE differs in both the scale and techniques leveraged in prior work. It expands the scale of simulation data to a significantly higher amount than previous work, as well as incorporates STPA concepts such as UCAs into the search algorithm. A simplified example of this approach is viewable below in figure 4-2

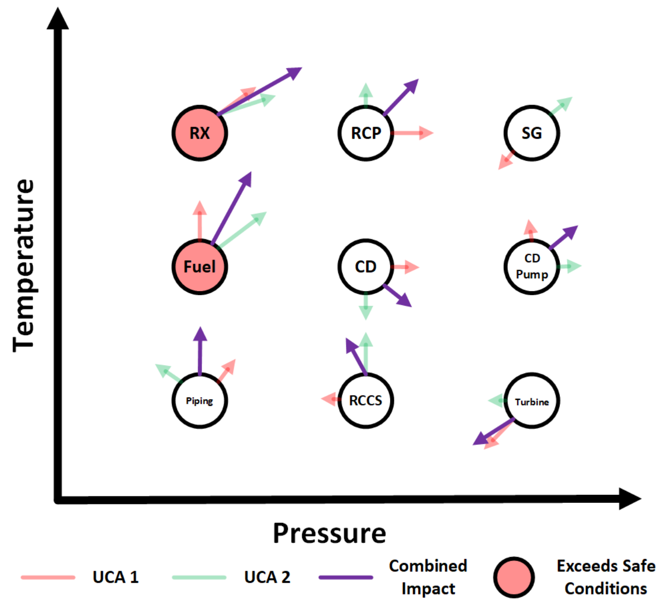

Figure 3-1. Search vectors with respect to the output observations in ARCADE

ARCADE conducts a search across several rounds of simulation on a given system. A single round consists of minimum U runs of simulations, where U is the number of the input domain, or the number of editable registers associated with an environment. Each round establishes a heuristic value associated with a run, which ARCADE defines as the magnitude of perturbation measured with respect towards the accident space. Subsequent runs are constructed based on the observed heuristic values of the previous runs. While this technique bears similarities to classical heuristic search algorithms, ARCADE leverages UCA categorization to conduct a more efficient search procedure than classical approaches.

Algorithm 1 Round-based UCA search with a simple threshold

```
Input: Simulator step $\delta(s,i)$ (enabled if guard $G$ holds),
accident check Accident($s$) over observations,
impact score $h(s)$ (larger = more disturbed),
threshold $\tau$, max depth $L$
Output: Accident traces found

Initialize queue $Q \leftarrow \{(s_0, \langle\rangle)\}$; Round $\leftarrow$ 1.
while $Q \neq \emptyset$ do
    Pop $(s,\pi)$ from $Q$ (by depth / FIFO).  if $|\pi| \geq L$ then
        continue
    if Accident($s$) then
        record $\pi$ and continue
    if Round = 1 then
        // Round 1:  singleton U1 sweep over each input tag
        Build candidates $\mathcal{I}$ by “provide” (U1)
          on one coordinate at a time (all tags once).
    else
        // Round 2+:  combine & refine, guided by prior impacts
        Build candidates $\mathcal{I}$ that combine actions (U2 withhold, U3 ordering, U4 duration).
          Construct candidates based on the impact of prior perturbation.
          Prioritize combinatorics of high-impact singleton UCAs, then temporal modification of
          existing combinatorics, then value modification of existing UCAs.
    foreach $i \in \mathcal{I}$ do
        if $G(s,i)$ then
            $s' \leftarrow \delta(s,i)$;   $m \leftarrow h(s')$.  if $m \geq \tau$ then
                push $(s', \pi \| i)$ into $Q$
    if Round = 1 and all tags swept then
        Round $\leftarrow$ 2
    if Round $\geq$ 2 and no new pushes this iteration then
        break
```

Within STPA, the identification of UCAs with respect to a given system is critical to understanding accident trajectories. ARCADE exploits this developed formalism for systems design in its algorithmic approach. STPA defines 4 categories of UCAs, which are re-iterated here and subsequently referred to as U(1-4) in this section.

- [U1] Command provided but not needed
- [U2] Command needed but not provided
- [U3] Command provided too early, too late, out of order
- [U4] Command stopped too soon, applied too long

ARCADE begins with a round comprising of the "singleton" UCAs, U1 and U2. These UCAs are grouped into a single round, as they represent the same underlying atomic action with respect to value modification, simply in opposite directions. Time is not a considered dimension with respect to these UCAs, so obtaining results for all associated U1 and U2 UCAs is possible in a single round of evaluation. The associated results of the first round are leveraged significantly in the subsequent round. All subsequent rounds are conducted based on the heuristics gathered from the previous round. After round 1, ARCADE switches to the combinatorics-based UCAs (U3 and U4). Time values are subsequently adjusted based on impacts of perturbation in rounds after 1, and attack values after the search reaches diminishing returns. The search concludes after no paths remain to be explored that exceed a heuristic impact threshold. This approach is detailed in Algorithm 1. While combinations of UCA categories could be explored in the future, ARCADE currently leverages single categories per simulation run.

3.3. Completeness Evaluation

In the future, ARCADE will conduct rigorous validation of the completeness of generated solutions in comparison with alternative approaches. While time-to-trajectory measurements are key in establishing the efficiency of ARCADE in comparison with traditional approaches, ARCADE will be subject to several bounded experiments to prove completeness on finite horizon problems. Further, ARCADE provides a platform for deeper exploration into algorithmic methods for determining accident scenarios. To provide initial success in these areas, ARCADE has conducted a robust analysis of the simplified IAEA Asherah NPP simulation engine, and uncovered all accident pathways in a reasonable amount of search time.

ARCADE leverages perturbation impact as a search heuristic in a similar vein to robustness methods such as the work of Annpureddy et al. [1]. While the approach of ARCADE is comparatively simpler than existing robustness formulas, results indicate such an approach is performant for the class of problems ARCADE is deployed for. The combination and subsequent exploration of high impact UCAs often leads to further accident scenarios. Due to the time bound imposed on problems, heuristic search techniques are guaranteed to explore all possible scenarios with respect to UCA combinations. While greater combinations of UCAs could expand this search space, such combinations are further bound by included plant hardware.

Even when analyzing results at a shallow search depth, U1 and U2 UCAs may provide immediately actionable feedback for system designers. As part of an initial exploration of completeness, ARCADE conducted an analysis of the Asherah simulator. Due to the simplified component structure of Asherah, ARCADE quickly identified all UCA sequences leading to accident conditions within the given time frame. A complete analysis was conducted in only two rounds of running, and results were encoded to demonstrate input registers associated with system component accident conditions. A display of such registers is color coded below in figure 3-2 in reference to the turbine system component. Due to the simplified system architecture, ARCADE is able to provide a complete list of accident scenarios within the Asherah system with respect to a given time domain.

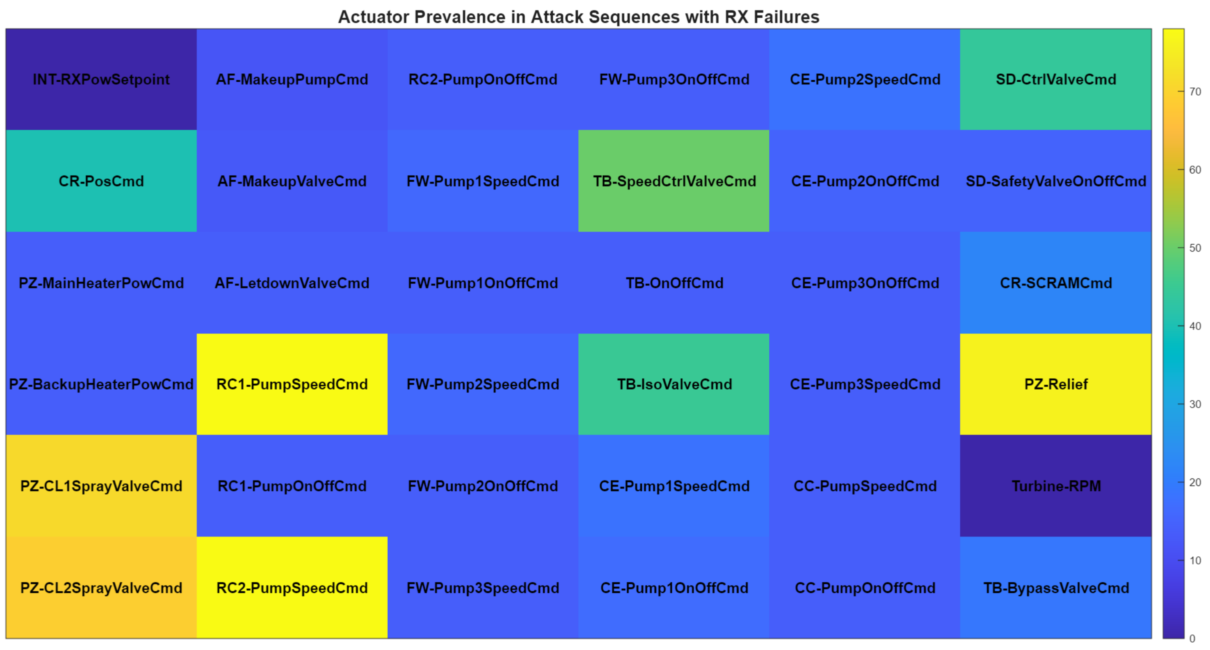

Figure 3-2. Identified vulnerable input registers after UCA actuation with respect to the simulated turbine component, color coded by impact severity.

Even in scenarios in which the time or space search domains may be significantly larger than Asherah, the search conducted by ARCADE is probabilistically complete with respect to system accident space. In all cases, searching beyond the initial structured sweep of U1 actions, every subsequent round retains a non-zero probability of sampling perturbations uniformly across the enabled input and timing space. This ensures that no region of the accident space is permanently excluded by the heuristic filter: given unbounded iterations, every neighborhood of a hazard-inducing sequence will eventually be explored with probability one. In practice, the heuristic impact score directs most expansions toward high-disturbance areas for efficiency, while the persistent heuristic horizon exploration guarantees that the probability of missing a reachable accident decays to zero as the number of samples grows.

This leads to the question of how coarse the search can be and still reach the most of the critical threat sequences. To validate the assumptions regarding the completeness of the ARCADE search method and the effects of search granularity on completeness, experiments utilizing multiple types of advanced reactor will complete an exhaustive search of all UCA combinations to a given depth. The ARCADE analysis will be compared to this exhaustive search to confirm that all possible threats were identified. The analysis system search granularity will also be tuned to ensure completeness with the greatest computational efficiency. Since no other system exists to execute and manage vast amounts of UCA simulations, ARCADE will have to be modified to undertake the exhaustive search. This will require significant computational resources and the development of ARCADE's clustering capabilities. Building the technical capability to perform such a large scale validation effort is part of the critical development path for ARCADE.

4. DEVELOPMENT PATHWAY

ARCADE supports numerous OT cybersecurity research and development areas which fall into the cyber-physical domain. There are three major areas where ARCADE is utilized which drive development: cyber-physical analysis, the equipment test environment, and training and education support. Figure 4-1 illustrates the underlying components of ARCADE which support these applications. Layer 0 constitutes the lowest level communication and operating system dependencies ARCADE relies on. The core utilities of Layer 1 provide the essential functionality of ARCADE's virtual environment and simulation interfaces. Layer 2 is comprised of the applications which directly support functionality for users and automated systems. Between Layers 1 and 2 are the Application Programming Interfaces (APIs) which enable the applications of Layer 2 to interface and control the lower level systems in Layer 1. At the top is a the management layer, which combine the applications to enable specific R&D and educational activities.

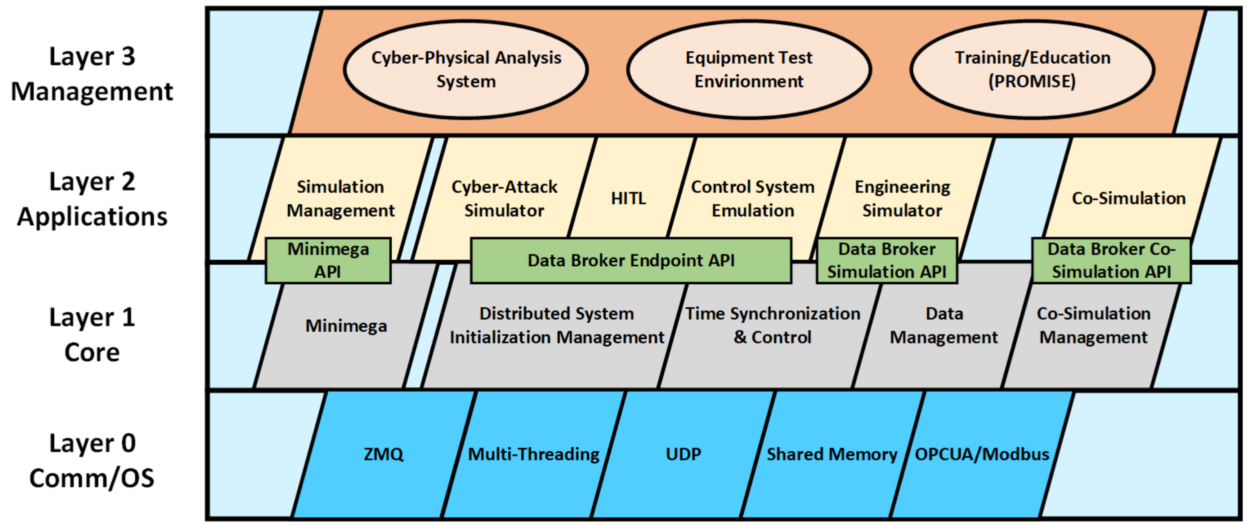

Figure 4-1. ARCADE system layers

The majority of ARCADE's development is focused in the top two layers and centered around the needs of each use case. Layers 0 and 1 development is complete, with the exception of co-simulation management which is expected in the next major open source release. Discussing the development pathway of ARCADE is best discussed in the context of each management layer use case.

4.1. Cyber-Physical Analysis

The cyber-physical analysis system is the core ARCADE use case, which fulfills the requirements of the TCA Tier 1 analysis. The current cyber-physical analysis system has enabled a proof of concept automated analysis of the notional Asherah system. This is a significant leap from the previous year which demonstrated manually driven analysis of a small number of scenarios. The analysis system will be a major focus of the next development stages, it must be developed into a production quality system. Improved system automation, better analysis methods, and user interface development will be critical next steps in its progress. To rapidly deliver on the goals of a complete and automated Tier 1 analysis, the underlying applications and the analysis system itself must be completed.

Three applications are relied upon for the operation of the analysis system: the simulation management, cyber-attack simulator, and engineering simulator applications. The analysis system uses the simulation management application to spawn many virtual environments which will run engineering simulators that are interfaced with the cyber-attack simulator. The analysis system itself is responsible for decision making regarding UCA combinations to evaluate and analysis completeness, as well as processing and organizing data. Figure 1-1 depicts the general relationship between these components. The development pathway for each of these components is detailed in this section.

4.1.1. Simulation Management

Minimega is controlled with the simulation management application in order to spawn many virtual environments which will host UCA simulations. This application also manages the injection of UCA simulation configurations, listens for completion of the simulation (or kills the simulation if it runs too long), and captures the raw data from the UCA simulation once complete. If ARCADE is intended to be run on many servers, the simulation management system distributes the computational load to each server. Initially Minimega meshing was investigated to manage multi-server applications, but this proved to have stability issues under the particular VM management needs of ARCADE.

The current simulation management system is sufficient to prove out the analysis concepts, however for production it must develop or improve the following:

- **Simulation Status Signaling:** Simulations may range in solution time, especially when computing complex threat sequences. It is unknown exactly how long each simulation will take, and therefore there must be some manner for the Data Broker within the virtual environment to signal to the simulation manager that it has completed, or failed the simulation. If a simulation failed, the simulation manager must capture data logs which can diagnose the cause of the failure. This seemly simple communication problem is made extraordinarily complex due to the isolated nature of Minimega virtual environments.
- **Clustering Capabilities:** To further reduce the computational time of the analysis, more parallel computational power is needed. The simulation manager must be made capable of orchestrating many servers in parallel while reporting data back to a central server. This must be scalable to thousands of compute nodes, especially for the validation efforts of full brute force analysis is to be feasible. When considering that each simulation may produce more than 1Gb of data, managing data on a large cluster with hundreds of thousands of simulations becomes non-trivial, and may invoke the need for distributed data pre-processing.

The expected development path is that simulation status signaling will be completed first. Clustering capabilities will scale as needed for validation efforts.

4.1.2. Cyber-Attack Simulator

Simulating cyber attacks is done with a modified Data Broker Endpoint which injects Unsafe Control Actions (UCAs) into the physics simulator (Figure 4-2). Control surface commands are rerouted automatically from the simulators internal controls, and unsafe control surface commands are injected in their place. Currently only 2 categories of UCAs are implemented in the simulator, being U1 and U2. Of those categories, each only has two UCAs developed. U1 can inject low and high values for the control surface, and U2 can inject static overwrites and replay normal control surface actuation data. For future development, not only do U3 and U4 UCAs need to be developed, but U1 and U2 should have improvements made to cover a greater variety of UCAs.

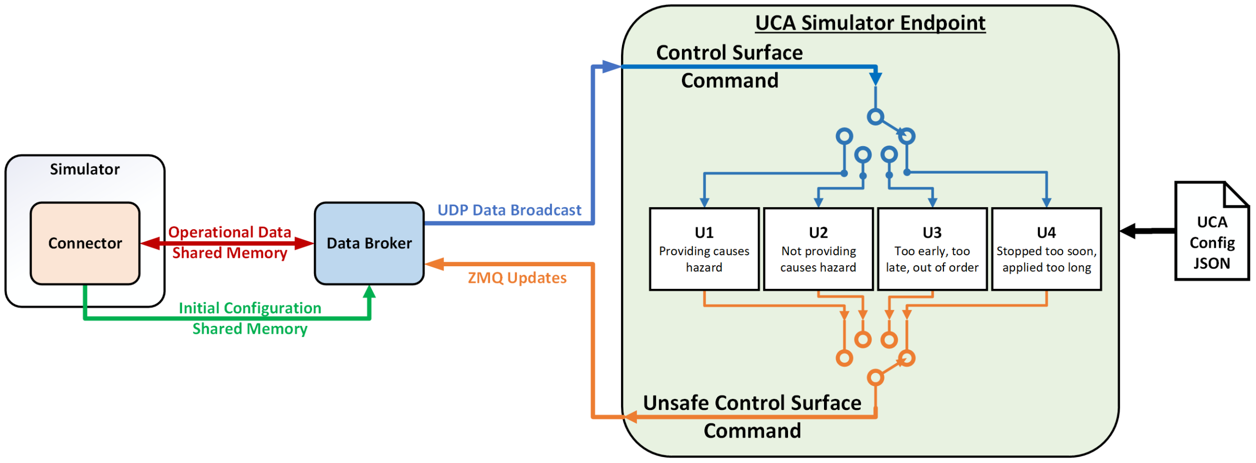
UCA Simulator Endpoint
Control Surface Command
Simulator
Connector
Operational Data Shared Memory
Data Broker
UDP Data Broadcast
ZMQ Updates
Initial Configuration Shared Memory
U1 Providing causes hazard
U2 Not providing causes hazard
U3 Too early, too late, out of order
U4 Stopped too soon, applied too long
UCA Config JSON
Unsafe Control Surface Command

Figure 4-2. A high level overview of UCA simulation within ARCADE

For binary control surface signals, all the UCA categories are simple to simulate, but most of the critical control signal in a plant are analog. For example: the position of a control rod is a continuous range from full insertion to full extraction. This makes UCA simulation much more difficult, as the concept of a UCA, such as 'needed not provided,' might imply a 55% insertion rate versus a 47% insertion rate at 575 seconds of operation. To accommodate the simulation of analog and temporal UCAs requires specific advancements to fully cover the spectrum of possible scenarios. These advancements are detailed as they relate to each UCA below.

- **U1 - Provided Not Needed:** The baseline functionality for U1 is complete, but waveform injection or oscillation injections can enable system harmonics investigation. Technically U1 can inject any value into the control surface, but it is the cyber-physical analysis system which is limited to the maximum and minimum travels of the control surface. Updating the analysis system to accommodate sweeps of the control surface space may illuminate more potential threats, but intuitively it seems unlikely. Further investigation, potentially after 2026, is warranted to determine whether such functionality would be worth the exponential increase in computation.

- **U2 - Needed Not Provided:** This UCA category has been developed, but its functionality has not been introduced into the analysis engine yet. While the basic functionality exists, there are still questions surrounding complex scenario use of this capability which arise when replaying normal control surface data. If the scenario U2 is deployed in already has 2 previous UCA's activated, which control surface data is correct to replay? Should the baseline data with no UCAs be replayed, or is it the data from the 1st UCA? Also, if the sequence of UCAs is long enough, there may not be enough data to replay from. The system should then signal this to the analysis system and new longer baseline be run. These features must still be developed in the analysis system for U2 to fully be utilized.
- **U3 - Provided Too Early/Late/Out of Order:** The first of the temporal UCA categories, U3 requires the replay of actuation data which is time shifted. This is rather simple if the signal is to be simulated as "Too Early" as the time series of replay data is just shift to the left of T_0. The "Too Late" UCA is rather more difficult as it requires shifting this time series to the right of T_0 leaving a gap of data which does not exist. If it can be assumed that the beginning of the simulation is at steady state, then it should be simple to just replay static data at the beginning of a simulation. However, it has been observed that simulations undergo perturbations when restarted from a paused steady state with some simulators. This creates an issue of when exactly to fill in data for the delay without altering initial conditions. These issues must be solved before the U3 UCA category can be completely simulated.
- **U4 - Stopped Too Soon/Applied Too Long:** Conceptually this is a simple UCA category to simulate, however, like U3 UCAs this category requires time compression and expansion. Additionally, the concept of stopped and applied also requires interpretation for analog signals. If a signal can be considered to be "applied" when above 60% but over the course of being applied the signal drifts, simulating a "stopped too soon" may require interpolation to bridge differences between the beginning and end of the truncated signal. Alternatively, the PID variables within the simulations control algorithm could be altered to cause U4 scenarios, but the varied nature of the simulation software and PID structures could make it difficult to manage and produce undesirable artifacts. These problems relate mainly to "stopped too soon", as "applied too long" simply requires slowing signal replay time on the rising edge of signal application.

The problems and advancements needed in the UCA simulator are not intractable and solutions are already being devised. These are primarily engineering problems which are seeking the most computationally efficient pathway. It is expected that U1-U3 core functionality will be completed in 2026, with U4 and advanced UCA simulations (such as waveform injection) will be follow on work.

4.1.3. Engineering Simulator Plugins

The engineering simulator is not a part of ARCADE, it is provided by the reactor designer who is utilizing ARCADE for tier 1 analysis. Advanced reactor designers have incorporated simulation tools to assist the development and qualification of their designs. It is standard practice now to create what is essentially a digital twin of the AR control system to properly design control system logic and develop control system specifications. These are high fidelity physics models of the plant and its control system, often they also operate in real-time. Perfect for ARCADE to leverage for analysis without incurring any extra labor on behalf of the designer. The catch is that these simulators run on a variety of simulation platforms which have a variety of code bases, API's, and integration pathways. This makes integrating with each require up front development labor on ARCADE's behalf.

Flownex® and MATLAB® Simulink® are currently the two simulators which have been integrated with ARCADE to provide nearly plug-and-play support. Developers only need to copy and paste the Data Broker connection code into their simulation and connect the I/O that ARCADE will use for the analysis. The intention is to allow this plug-and-play support for all the major Engineering simulators in use across the U.S. domestic nuclear industry. This requires significant development effort and will be prioritized with industry interest in ARCADE.

4.2. Equipment Test Environment, TANGO, & PROMISE

The other applications for ARCADE are supporting system of systems simulation, hardware in the loop, and training and education. These have potential applications for the nuclear industry, but are not directly supported by the main ARCADE project. Each of these applications and their influence on the development of ARCADE is described in this section.

Equipment Test Environment or (ETE) is an infrastructure as a service project. The ETE allows users to check out OT hardware such as Programmable Logic Controllers (PLCs) and connect these to large scale network and physics simulations to perform research across the cyber-physical space. The ETE was born out of the need to perform experiments with OT systems that could not be emulated, such as FPGA based systems. A high-speed, high-fidelity analog signal regeneration and capture system was developed to integrate these OT systems in such a way as allow the OT devices to be operated in a manner indistinguishable from the real plant. This has been identified as having industry use in the development and testing of hardware and logic of the control system. The ETE was developed to serve remote users, another significant advantage for a much more remote engineering workforce in the industry.

At the heart of the hardware in the loop capabilities of the ETE is ARCADE, which provides a link from physics simulations, out of large scale virtual networks, and directly into PLCs and analog signal regeneration and capture devices. The Data Broker and Endpoints were originally developed to interface with hardware in the loop and emulated PLCs. The Data Broker provides the deep connection with the physics model's control surfaces, and an Endpoint outside of the virtual network is able reach out to external PLCs and the signal regeneration and capture equipment. This application has lead ARCADE to support Modbus, OPCUA, and ZMQ communication protocols and support for direct analog I/O writing and reading. These hardware capabilities will likely expand as more R&D project move to ETE.

TANGO or Twinned Analysis of Nuclear reactor and Grid Operations is a project which seeks to develop a system of systems model of nuclear reactors attached to the grid. It seeks to emulate the networks of these systems to research various cybersecurity questions and as a development space

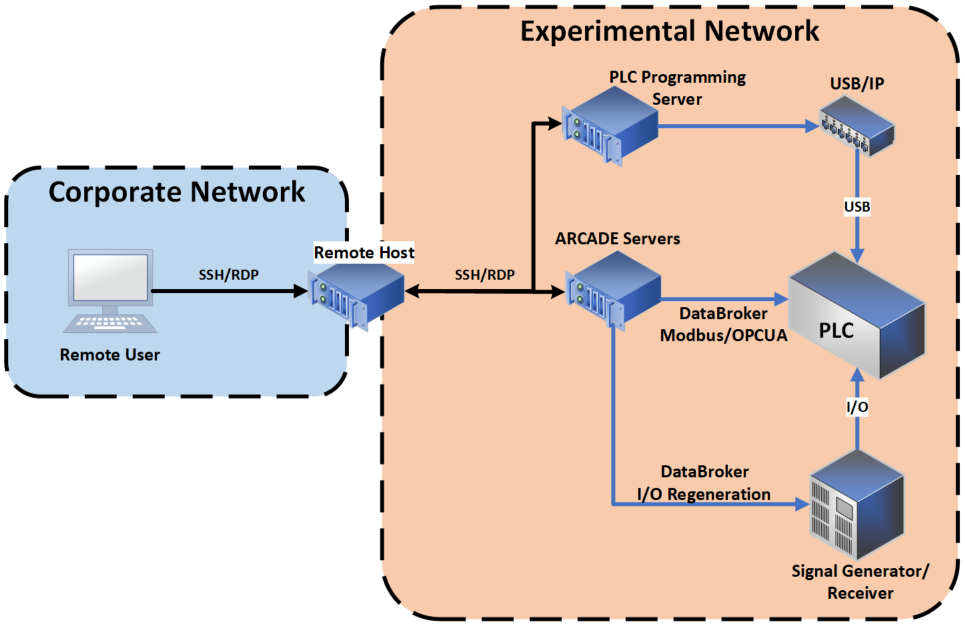

Figure 4-3. High level diagram of the Sandia Equipment Test Environment (ETE).

and testbed for advanced OT cyber network analysis systems. The physics of the grid and nuclear plants are modeled to determine cyber threat consequence and coupled system risks.

ARCADE supports the nuclear portion of the TANGO system, and has been modified to integrate with multiple other physics simulations. This has spurred the development of the ARCADE co-simulation application which allows ARCADE to connect with any other simulator and exchange physics data. A co-simulation time control mechanism was created to allow ARCADE to become the central simulation time keeper or for it to allow another system to control time steps for ARCADE. This capability allows large scale holistic simulations to be evaluated with ARCADE's tier 1 analysis system. The co-simulation system is expected to continue to be refined as needed by the TANGO project, but presents interesting opportunities for studying the risks around large scale problems like load flowing for advanced reactors.

PROMISE stands for Power Reactors on Minimega In Service of Education which is the educational environment that is supported by ARCADE components. PROMISE is an emulated control system and network for the Asherah NPP simulation which is often referred to as ARCADE lite. It allows students to interact with a representative nuclear control system, design PLC logic and SCADA control panels, then attack the network to observe the consequences. Using educational hacking tools, the students become the hacker to understand at a deep level, how to protect control systems inherently from the principals of the TCA. Students also become familiar with ARCADE as a tool for cyber analysis, making it a key bridge for training future ARCADE users.

PROMISE has been deployed for various training and education missions over the last four years of its existence. It was central to an international OT cybersecurity course and a semester long university course has been developed around it. Direct support for PROMISE ended this year, however its development support is continued by the universities which have adopted it. The

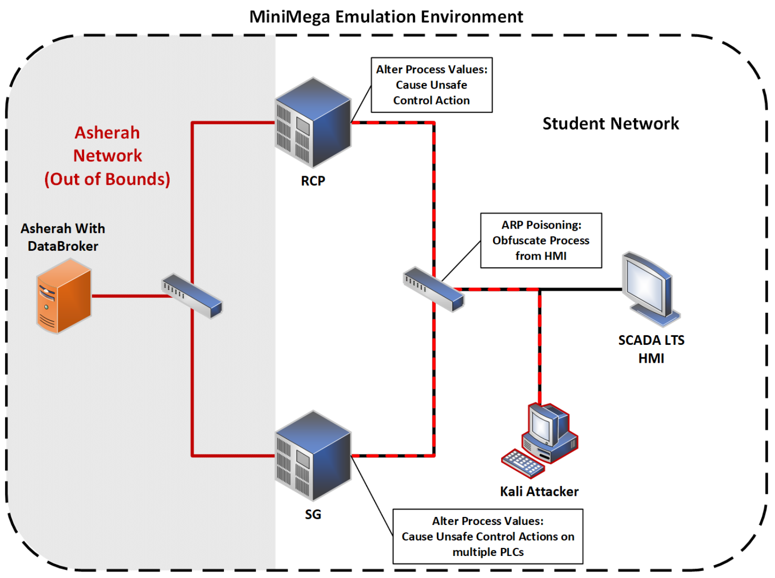
MiniMega Emulation Environment
Asherah Network (Out of Bounds)
Asherah With DataBroker
Alter Process Values: Cause Unsafe Control Action
RCP
Student Network
ARP Poisoning: Obfuscate Process from HMI
SCADA LTS HMI
Kali Attacker
SG
Alter Process Values: Cause Unsafe Control Actions on multiple PLCs

Figure 4-4. PROMISE student environment with 1st attack example.

development for PROMISE as ARCADE lite is expected to be restarted once ARCADE reaches a point where greater education is needed.

This page intentionally left blank.

5. CONCLUSION

ARCADE has made significant progress in its ability to automatically search through UCA sequences and provide valuable cyber risk information on nuclear power plants. Even with only the capability to simulate "Provided Not Needed" UCAs the ARCADE system can significantly reduce the effort needed to perform fault tree analysis. As ARCADE's capabilities mature, it will enable risk analysis which is not possible with conventional human driven processes. The example analysis of the Asherah simulator demonstrates that the technical pathway for ARCADE's analysis is feasible. There still remains a long development pathway to bring ARCADE to a production level, but direct impact to the nuclear industry is achievable within the next year.

Providing the foundation of an analysis using the TCA as outlined in NRC RG 5.96 [8] is within reach for ARCADE. If this is achieved, the cost of implementing defense-in-depth cybersecurity for advanced reactors will be significantly reduced [6]. As ARCADE becomes capable of this analysis, the final major hurdle will be validation of its methods. As described in this report, this validation pathway is not insurmountable and is already being planned for to expedite the process. Once ARCADE's analysis is validated, it should be a suitable tool for the advanced reactor industry to generate cyber risk data which is presentable to regulators.

This page intentionally left blank.

REFERENCES

[1] Yashwanth Annpureddy, Che Liu, Georgios Fainekos, and Sriram Sankaranarayanan. S-taliro: A tool for temporal logic falsification for hybrid systems. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 254–257. Springer, 2011.

[2] Abhishek Chaudhary, Junseo Han, Seongah Kim, Aram Kim, and Sunoh Choi. Anomaly detection and analysis in nuclear power plants. *Electronics*, 13(22), 2024.

[3] Alexandre Donzé. Breach, a toolbox for verification and parameter synthesis of hybrid systems. In *International Conference on Computer Aided Verification*, pages 167–170. Springer, 2010.

[4] Andrew S. Hahn, Lee T. Maccarone, Titus Gray, Adam J. Beauchaine, Michael T. Rowland, Fraser Dougall, and John Connor Grady. ARCADE Technical Pathway and Industry Impact, February 2025. SAND2025-02425R.

[5] Edward A Lee and Pravin Varaiya. Signals and systems, 2003.

[6] Lee T. Maccarone and Michael T. Rowland. The Sliding Scale of Cybersecurity Applied to the Cybersecurity Analysis of Advanced Reactors. In *American Nuclear Society 13th Nuclear Plant Instrumentation, Control, and Human-Machine Interface Technologies*, pages 215–223, Knoxville, TN, 2023.

[7] RABE Silva, Koroush Shirvan, José Roberto Castilho Piqueira, Ricardo Paulino Marques, et al. Development of the asherah nuclear power plant simulator for cyber security assessment. In *Proceedings of the international conference on nuclear security, vienna, austria*, pages 10–14, 2020.

[8] U.S. Nuclear Regulatory Commission. NRC Draft Regulatory Guide DG-5075: Establishing Cybersecurity Programs for Commercial Nuclear Plants Licensed Under 10 CFR Part 53, June 2024. Available online.

This page intentionally left blank.

APPENDIX A. ARCADE Technical Information

A.1. ARCADE Input File

ARCADE uses standard JSON input file syntax which is block structured with lines composed of key value pairs. A block begins with the ’’ character, ends with a ’’ character, and all lines in the block are terminated with a comma. Each line has one or more keys, which is typically a string, and is separated from the value by a colon. If there’s multiple inputs on a line, each key value pair is separated by a comma before the next key value pair.

```
{
  "attack_time" : 605,
  "max_active_simulations" : 32,
  "comment_minimega_home" : "/opt/minimega",
  "port" : 9136,
  "search_depth" : 2,
  "simulation_runtime" : 900,
  "wallclock_timeout" : 900
}
```

Listing A.1 ARCADE Input File

Each physics code system is defined uniquely in the physics input file. Each physics has a unique set of accident conditions and control surfaces that are required inputs (the accident conditions are required to perform data analysis), the control surfaces are required to run the physics, when not analyzing existing results. The Attack Directory is the top-level output directory where ARCADE writes the search depth analysis results, the attack JSON and YAML files, and other information. Specifying the attack directory runs the Asherah physics with the specified control surfaces, and uses the specified accident conditions during analysis. In order to analyze existing results, comment out the Attack Directory and enable the Job Results by setting the job results to the file for a search depth attack_uids.txt file.

```
  "asherah" : [
    { "commentjobresults" : "</path/to/attack_uids.txt>" },
    { "attackdirectory" : "</path/to/attack/directory>" },
    { "accidentcondition" : "CDLevel", "operator" : ">=", "value" : 1.5, "unit
   " : "m" },
    { "accidentcondition" : "CDPress", "operator" : ">=", "value" : 6760, "
   unit" : "Pa" },
    { "accidentcondition" : "RXDT", "operator" : ">=", "value" : 40, "unit" :
   "K" },
    { "accidentcondition" : "RXTavg", "operator" : ">=", "value" : 580, "unit"
    : "K" },
    { "accidentcondition" : "SG1Level", "operator" : "<=", "value" : 12, "unit
   " : "m" },
 { "accidentcondition" : "SG1Press", "operator" : ">=", "value" : 8974000,
"unit" : "Pa" },
 { "accidentcondition" : "SG2Level", "operator" : "<=", "value" : 12, "unit
" : "m"  },
 { "accidentcondition" : "SG2Press", "operator" : ">=", "value" : 8974000,
"unit" : "Pa" },
 { "controlsurface" : "TBIsoValveCmd", "min" : 0.0, "max" : 1.0, "value" :
"Value1" },
 { "controlsurface" : "TBSpeedCtrlValveCmd", "min" : 0.0, "max" : 100.0, "
value" : "Value2" },
 { "controlsurface" : "PZMainHeaterPowCmd", "min" : 0.0, "max" : 100.0, "
value" : "Value25" },
 { "controlsurface" : "PZBackupHeaterPowCmd", "min" : 0.0, "max" : 1.0, "
value" : "Value26" },
 { "controlsurface" : "PZCL1SprayValveCmd", "min" : 0.0, "max" : 100.0, "
value" : "Value27" },

]
```

Listing A.2 ARCADE Physics Input File

A.2. Accident Conditions

Accident conditions describe under what scenario the physics results are considered to have caused an accident. There can be multiple accident conditions, up to as many as there are output variables. Note that some accident conditions are not present in the output file, for example, RX_DT and RX_Tavg. These two quantities are computed from other output variables. RX_DT is computed as the absolute difference between the outlet and inlet coolant temperatures while RX_Tavg is the average coolant temperature in the reactor.

$$RX_{DT} = |out\_cool\_t - in\_cool\_t| \tag{A.1}$$

$$RX_{Tavg} = \frac{in\_cool\_t + out\_cool\_t}{2} \tag{A.2}$$

All accident conditions should provide a constraint operator, a value, and unit so that it is known under what conditions an output is considered to have caused an accident. The current Asherah model has 8 accident conditions (6 output variables and 2 computed from outputs).

Our Asherah accident conditions come from Table 1 of the Chaudhary et. al. [2]. A brief description of the accident condition variables is in Table A-1.

Variable	Description
CD_Level	Hotwell level, which automatically transfers the condensate to or from the storage system.
CD_Press	Pressure of the condenser, which is monitored to verify condenser operation.
RX_DT	Reactor temperature difference across the simulation timestep.
RX_Tavg	Average reactor temperature of the inlet and outlet coolant operating temperatures.
SG1_Level	Water level in steam generator 1.
SG2_Level	Water level in steam generator 2.
SG1_Press	Pressure inside steam generator 1.
SG2_Press	Pressure inside steam generator 2.

Table A-1. Description of Accident Condition Variables

A.3. Control Surfaces

Each control surface describes an input to the physics model along with the valid range of values for that input. The value tag is an internal identifier that will be removed in a future version of the code. Each control surface has a name, a minimum value (or lower bound) and a maximum value (or upper bound). Our current Asherah model has 35 control surface inputs.

A.4. IO Broker

The IO Broker component is designed to orchestrate both IO processing (i.e. reading the JSON input file), generating the initial set of attacks based on the input, submitting the jobs to the ARCADE Server to be run, waiting for the server response to confirm that the job(s) have completed, and launching the data analysis once all the attacks in the current search depth have completed. If the search depth is greater than one, the IO Broker will take the results of the first analysis and use that to generate, run, and analyse another set of attacks.

A.5. Data Analyzer

The data analysis component is responsible for determining if any accidents occurred in the current search depth results. Because the physics output can be large, typically a run is performed with a 0.1s time step for an hour, resulting in 36000 data points per output variable. There are typically dozens or more output variables than there are accident conditions so the output files can be quite large to copy, load into memory, and analyze. For this reason, multi-threaded data analysis is leveraged to utilize each available core during analysis. This should help the analysis to scale up as runs consist of larger problems with more attacks.

Data is extracted from the pub_data.csv file generated by the data collector when the physics terminates. The data is in the form of a CSV (Comma Separated Values) file with each line of output consisting of the output variable name, the variable value, and the time step. Because the physics runs with a fixed time step of 0.1s, an hour long run produces 36000 output lines for each output variable. Since the pub_data.csv output file can be quite large, and to avoid problems copying the file from the VM back to the host, the file is compressed with gzip. Compression reduces the file size by roughly an order of magnitude, and the output file is then transferred back to the host in compressed format.

When ARCADE notifies the IO Broker that all physics have completed, the compressed output files from /tmp/minimega are moved into the local search depth directory. This is done so that the physics output from each search depth pass can be separately tracked. Once the files are copied out of /tmp/minimega, and the file that provides a mapping from the UID (Universal Identifier) to the attack number (attack_uids.txt) is retrieved, the pub_data.csv files are decompressed. Python is much faster reading the uncompressed files than using the Python compression library to read the files. With the output files uncompressed, the data is loaded into a large rank 3 array dimensioned by accident condition, attack id, and time step output values. For the default analysis presented (in search depth 1), the array is dimensioned [8, 71, 36000] where 8 is the number of accident conditions, 71 = 2 * 35 + 1, 35 is the number of input variables (i.e. control surfaces), 2 is U1-L and U1-U for each control surface, and the additional 1 is for the base case which runs with nominal values. For later searches, the array size grows significantly as the number of attacks increases. Note that sometimes Asherah runs faster or slower for some attacks so we retain only the maximum number of timesteps present for each output variable. Once the data has been loaded into the array by the IO Broker, the IO Broker calls the Data Analyzer to perform the analysis.

The basic steps in the data analysis are to 1) partition the analysis across the number of available cores running one thread per core 2) set the thread processor affinity so that threads aren't swapped to different cores during the analysis for enhanced performance 3) retrieve accident condition data (name, safe_limit, operator, and unit) and attack data by accident condition. Each thread analyzes one accident condition so a thread team (group of threads) partition all the attacks for a single accident condition across multiple threads. In this way, the threads are partitioned along two axes, one is a thread group for each accident condition, the other axis partitions the attacks in a single accident condition across multiple threads.

In order to determine if an accident has occurred, each attack accident condition is checked against the safe limit that was in the JSON input file. There are currently two possible accident conditions, one where a value is less than or equal to a lower limit, and one where the value is greater than or equal to an upper limit. If the CD_Level accident condition stipulates that any value greater than or equal to 1.5 meters is an accident, then each output variable for each attack is compared, to see if any output value is greater than or equal to 1.5 meters. If a CD_Level value is found that exceeds the safe operational envelope, then an error diagnostic is reported, this attack causes an accident, and this attack is removed from future search depths when new attacks are generated.

The following output can be found in the attack_directory analysis_results.txt file. The output details: the label of the attack as the name of the JSON input file (attack_13.json), the accident condition variable (SG1_Level), the failed condition (minimum value of SG1_Level <= 12m), when the accident condition occurred (663.3s), and the input variable(s) and UCA(s) that caused this accident.

```
attack (13) SG1\_Level min value 11.723754(m) <= safe lower limit 12.0(m)
   occurred at 663.3(s) (FW\_Pump1SpeedCmd U1-U)
```

Listing A.3 Example Single Attack Output

In later search depths, the attacks are grouped as tuples (pairs, etc..) so there's a bit more complexity in the accident condition output. For example, here attack_22.json combines two original atta‘cks, 2 and 6, and it lists the two input variables that were combined in this accident attack. Note that attack 22 could cause accidents due to multiple accident conditions. Here attack 22 fails because the SG1_Level value causes an accident, but there's 7 other possible accident conditions that could also cause this attack to fail.

```
attack (22) combines (2, 6) SG1\_Level min value 9.23(m) <= safe lower limit
   12.0(m) occurred at 615.3(s) (TB\_SpeedCtrlValveCmd U1-L, RC2\
   _PumpSpeedCmd U1-L)
```

Listing A.4 Example Search Depth 2 (Two Attack) Output

The original (search depth 1) attack ids are listed in parentheses following the word "combines" in order to link them to the attacks that are being combined from the initial sweep. The attack id is the name of the JSON attack file in the search_depth_2 directory. Because of how the attacks are combined, each search depth yields attacks with that number of attacks. For example, search depth 2 will always combine pairs of original attacks, search depth 3 will always have combine original attacks into triplets, and so forth.

DISTRIBUTION

Email—Internal

Name	Org.	Sandia Email Address
Andrew S. Hahn	8851	ashahn@sandia.gov
Technical Library	1911	sanddocs@sandia.gov