The Case for Robust Adaptation: Autonomic Resource Management is a Vulnerability

Joseph Noor†1, Ahmed Ali-Eldin‡2, Luis Garcia‡3, Chirag Rao‡4, Venkat R. Dasari‡5, Deepak Ganesan‡6, Brian Jalaian‡7, Prashant Shenoy‡8, and Mani Srivastava9

†University of California, Los Angeles; ‡U.S. Army Research Laboratory; ‡University of Massachusetts, Amherst

Abstract—Autonomic resource management for distributed edge computing systems provides an effective means of enabling dynamic placement and adaptation in the face of network changes, load dynamics, and failures. However, adaptation in-and-of-itself offers a side channel by which malicious entities can extract valuable information. An attacker can take advantage of autonomic resource management techniques to fool a system into misallocating resources and crippling applications. Using a few scenarios, we outline how attacks can be launched using partial knowledge of the resource management substrate – with as little as a single compromised node. We argue that any system that provides adaptation must consider resource management as an attack surface. As such, we propose ADAPT2, a framework that incorporates concepts taken from Moving-Target Defense and state estimation techniques to ensure correctness and obfuscate resource management, thereby protecting valuable system and application information from leaking.

Index Terms—autonomic computing, side-channel attacks, Internet-of-Things, distributed systems, edge security

I. INTRODUCTION

The expansion and evolution of distributed edge computing has enabled powerful applications spanning large-scale centralized clouds, mobile edge clouds, IoT and mobile devices, with emerging workloads ranging from agriculture to city planning and military surveillance. Due to the highly dynamic and heterogeneous nature of these environment domains, it is necessary for an underlying system to provide resilient and reactive application management [27]. Adaptive resource allocation and scheduling, particularly autonomic resource management, provides a means to implicitly maintain application semantics while adjusting to changes in workload dynamics, network topology, link quality, and node capabilities. The increasing abundance of edge computing and IoT system necessitates an increased emphasis on both the security of these devices, as well as security mechanisms to guard against the harmful implications of devices being compromised [21], [10]. Maintaining a secure network remains a fundamental requirement for these systems to reach their full potential. In this paper, we argue that adaptation in autonomic resource management for applications spanning distributed edge-cloud networks exposes an attack surface with critical vulnerabilities.

Robust algorithm and model development is well studied in applications where attackers target the semantic properties of autonomous systems, such as the physical model of a cyber-physical system [7] or the decision boundaries of deep learning as in adversarial machine learning [11]. However, resource management has yet to be cast in this context. As autonomous resource management systems become more pervasive, the importance for robust adaptation increases. Compared to centralized clouds, the challenges posed by the heterogeneity, mobility, scale, and resource constraints of mobile and IoT devices offloading computations to edge-cloud systems exacerbate the difficulty in designing proper management solutions.

The essence of autonomic resource management in computer systems is to adapt resource allocation based on trusted load measurement data reported from the physical devices. We argue that one largely overlooked concern is the opportunity for an attacker to use information about adaptation triggers to launch an attack. These vulnerabilities in autonomic resource management enable complex attacks that leak cyber-physical aspects of the system, propagates malicious or exploitative code, and/or launch side-channel attacks against other co-located users. We enumerate cases where an attacker can report false utilization statistics to invoke adaptive network behaviors in a targeted fashion, e.g., over-reporting utilization to request additional resources from the resource manager.

Even without controlling a device in the system, a malicious entity can infer sensitive spatio-temporal information by merely observing the current state of network activation. The vast inter-connectivity of edge networks increase both the visibility and accessibility of network nodes. This inference of activity based on adaptation has significant implications in safety-critical applications, such as those in smart cities, industrial IoT, and the Internet-of-Battlefield-Things [24]. Failing to obfuscate location information of these devices can be crippling in mission-critical contexts.

In this paper, we use three attack scenarios derived from the aforementioned vulnerabilities to propose the ADAPT2 framework for secure and robust resource management. Using techniques derived from Moving Target Defense [12], we suggest state estimation of computational models to identify,
having a full view of the distributed computation of workloads. Finally, computing resources are shared between the software components using light weight virtualization such as Micro-VMs, thus providing multi-tenancy on the network’s sensing, computation, and actuation resources [17], [28].

B. Adversary Model

The adversary’s goal is to manipulate resource allocation and infer sensitive application information. We assume that the adversary wants to maintain stealthiness, so as not to raise suspicion. Stealthy attacks such as Stuxnet [7] have proven that stealthiness enables longer and more impactful attacks, especially in a cyber-physical context. We focus on attacks that specifically target the resource adaptation mechanism.

Attack vectors. In order to manipulate resource allocation, an attacker must compromise at least a single device. However, an attacker can still derive system information without compromising a device. In summary, we consider the following attack vectors:

1) An attacker has gained access to at least one physical device. Gaining physical access to a device is significantly easier in an edge network [25]. This allows an attacker to mount a physical attack by either attaching malicious hardware or manipulating the device platform [9].

2) In IoT networks that allow ad-hoc admission of IoT devices, an attacker uses a malicious device to join the network, and is accepted by a Resource Manager via the legitimate admission interface.

3) The attacker has not gained access to a physical device, but has gained access to a software component. The attacker may exploit a software or network vulnerability to gain root or user access. While this type of attack is not specific to the IoT edge domain, IoT and mobile software are often less robust, thereby offering more exploits when compared to typical machines [25].

4) The attacker has not gained device access, but can obtain meta-information regarding device activation. This information can be obtained via side-channel attacks, such as remotely detecting changes in signal output, generated heat, or electromagnetic leakage [16].

C. Assumptions

These four attack vectors, illustrated in Figure 1, lead to an assumption of an untrusted edge network. The attacker has managed to circumvent existing security mechanisms implemented on the system to gain access to devices. This model also assumes that we cannot attest the integrity of either the software or hardware for some devices\(^1\). Similarly, we assume that any trusted execution environments can be circumvented by attackers [26].

Number of compromised devices. We assume the attacker can only compromise a subset of the system, as a distributed

\(^1\)Although remote attestation can be implemented, this typically requires a hardware root-of-trust. If a device doesn’t support such hardware, software-based attestation can be used but typically requires very strong assumptions regarding the timing and authenticity guarantees of the communication channels between the device and the external verifier [23]
attack spanning all heterogeneous devices located in different physical regions would be infeasible.

**Trusted resource management.** The resource manager is assumed to be a trusted entity that has much stronger security guarantees than other devices, particularly those residing at the edge. We assume this entity has more stringent cyber and physical security mechanisms.

**Computing power.** We assume an attacker has access to an adversarial pool of external resources that can be used for processing and offloading of computations. While not necessary, this effectively provides an attacker infinite compute resources by which to fool any source of compute validation necessary, this effectively provides an attacker infinite compute for processing and offloading of computations. While not necessary, this effectively provides an attacker infinite compute resources by which to fool any source of compute validation checker.

**Applications.** We assume the applications of interest are deployed and managed by an autonomic resource manager over a distributed edge network. For the purposes of our discussion, we will look to applications that span a smart city environment to aid law enforcement and first responders.

### III. ATTACKS ON ADAPTATION

In this section, we enumerate three attacks on autonomic resource managers in edge computing systems. The first two focus on using the utilization statistics reported by a device to the monitoring entity to attack the system. The latter attack focuses on making inferences about spatio-temporal network characteristics given a network’s resource allocation and system meta-information. There are many more attacks that can be launched from a set of compromised devices in a autonomic computing system; we focus on those that target the resource management substrate.

**A. Falsely reporting low utilization**

Resource managers often implement a dynamic scheduling policy that considers current device utilization or computational capacity (e.g., CPU cores, memory, etc). Tasks are more likely to be assigned to devices experiencing low utilization as opposed to those experiencing high utilization. Reported utilization and/or capacity is often accepted at face-value without verification from the resource manager; however, in an ad-hoc network it may be important to treat device-reported statistics with scrutiny. A compromised device can fool a resource manager by providing false utilization information.

In this attack, a compromised device reports underutilization and/or claims to have very large capacity. The resource manager may then choose to map more tasks to the underutilized device, allowing the compromised node to gain access to more application semantics. In order to maintain stealthiness, the compromised node can offload some assigned computations to an adversarial external cluster.

The attacker can use the placement of multiple application components by the resource manager on the compromised node to stage other attacks such as data-poisoning. In the worst case, the attacker will have access to all applications’ semantics, as the resource manager will blindly place software components onto the device with the lowest resource utilization in the network.

**Example attack.** An application deployed over a smart city is likely to span a wide range of devices. Exfiltrating semantic behavior to untrusted entities can ultimately cripple application objectives. We present this attack against the first-fit placement algorithm, a simple but efficient placement algorithm [5]. Given a software component, e.g., a virtual machine, first-fit assigns to the first available physical machine with sufficient capacity to meet QoS requirements. For our example, we define a scenario with 5 homogeneous physical machines (PMs) capable of supporting 3 equivalent virtual machine (VM) workloads each. The placement algorithm fits these 10 VM workloads onto the PMs at each time epoch, with each workload requiring 2 time epochs to complete. Figure 2 presents this scenario under normal operation. Figure 3 illustrates an attacker exfiltrating workloads to an external entity via a compromised PM. At each time epoch, the compromised PM reports underutilization, thus gaining access to 3 VM workloads at every epoch.

**B. Falsely reporting high utilization**

One of the key tasks of a resource manager is to maintain Quality-of-Service (QoS). Variations in device and application
performance are common, typically a result of workload burstiness, energy dynamics, and device mobility. A resource manager may initiate an adaptation to remap computation to resources and maintain QoS with respect to a pre-defined system metric (e.g. latency, energy, utilization). While important for all applications, maintaining strict QoS is essential for mission-critical and real-time applications, such as those in search-and-rescue and military applications [2]. When faced with an over-utilized device or unacceptable QoS, a resource manager will typically assign more resources to the application, either by horizontally scaling, i.e., adding replicas of a software component to additional devices, or by vertically scaling, i.e., increasing the memory or CPU allocation to a particular software component.

In this attack, the compromised software component reports that it has experienced an increase in utilization and violated QoS. If vertically scaled, the resource manager will allocate more resources to the compromised software component, allowing the malicious component to control more resources that now cannot be used by legitimate applications.

Figure 4 illustrates how a compromised VM workload is horizontally replicated by the resource manager at each time epoch if it is consistently reporting that it requires an additional PM. The resource manager scales the malicious component creating more replicas of the malicious software, thereby allowing an attacker to infect more devices and mount serious attacks, such as cross-VM side channel attacks [29]. By replicating malicious software, the total available resources will be effectively reduced, leading to a Denial-of-Service launched by the resource manager itself. This style of DoS attack is quite common in the IoT and edge computing domains, including those incorporating robust autonomic resource management systems [13].

C. Inferring action from adaptation

Resource Managers often optimize computation placement based on a particular heuristic, for example end-to-end latency. Applications with a spatio-temporal aspect may leak sensitive location information as a direct result of a resource manager adapting computation placement or sensor/actuator activation.

In this attack, we consider an application that computes data over a dynamic region. For example, consider a smart city environment where a set of law enforcement vehicles are moving through an IoT and infrastructure-rich environment. An application is deployed to identify suspicious activity and potential threats surrounding the vehicles. As these vehicles traverse the territory, edge devices are active depending on whether a vehicle is located nearby. An attacker can observe the currently active set of devices to infer information on vehicle locations. In this way, adaptation can leak sensitive spatio-temporal information to an attacker, who can construct a location heat-map based on device activity.

IV. ADAPTING ADAPTATION

To counter the aforementioned attacks on adaptation, we present how existing characteristics of adaptive models can be instrumented to detect false reports of device utilization statistics and obfuscate leaked spatio-temporal information.

A. Detecting False Device Utilization Statistics

Due to potential attacks on a resource manager’s placement algorithm, a robust system must scrutinize reported device utilization statistics. An obvious solution to detect if a device has been compromised is to verify the integrity of the device’s software state via device-fingerprinting [8] or standard trusted computing techniques [19]. However, in dynamic distributed IoT systems where new devices are being recruited onto the network, it is difficult both to build deterministic fingerprints for devices as well as to ensure that these devices are enabled with trusted platform modules to perform remote attestation. As such, we opt for a solution that depends on the state of the network’s resource consumption model.

1) State estimation for resource consumption: The first component is analogous to state estimation for detection of false data injection attacks [20]. We propose generating probabilistic models of expected resource consumption based on device characteristics. For a given task and device, a predicted target is compared to reported statistics. This enables
a challenge-response against each device to verify that a reported resource utilization is within certain bounds of the model. An attacker may have the ability to spoof responses given a particular workload challenge, but can no longer report drastically inaccurate utilization statistics, thus preventing a crippling attack leveraging reported resource utilization.

2) Moving target defense for more robust state estimation: If an attacker can learn a probabilistic model of resource consumption for a given application workload, then the attacker may be able to provide spoofed data to fool the challenger. To counter this spoof, a resource manager can rely on Moving Target Defense (MTD) techniques, where resource mappings are constantly changed to randomize computation assignment to devices [12]. As a device’s workload changes, accurately predicting expected resource consumption becomes more difficult, and an increased opportunity arises for identifying malicious devices. To spur workload changes, we can adapt MTD by randomly initiating an unprovoked “adaptation” to remap computation to devices and initiate a new round of challenge-response. The resource manager can improve initial state-estimation models by collecting utilization and resource consumption data from each round of MTD for all the components and devices. This presents a clear security-performance trade-off dependent on overhead in adaptation; furthermore, meeting QoS requirements under network constraints may restrict the space of possible MTD configurations.

3) Isolating suspicious devices: Suspicious devices that are marked as deviating significantly from an expected model can be isolated and further evaluated. This group can be tested with either latency-tolerant applications or dummy workloads. Dummy workloads can be explicitly designed to intentionally generate resource consumption outside a given model’s expectations. In doing so, it becomes easier to determine whether a device is intentionally misreporting utilization statistics.

B. Obfuscation of Leaked Spatio-temporal Characteristics

Although the state estimation model with induced randomness can provide a means of verifying a device’s resource utilization characteristics, it does not obfuscate the side channels that are exposed by adaptation, e.g., identifying movement by detecting computational workload patterns in different locations. As such, we propose instrumenting the aforementioned redundant workloads in a way that randomizes the activation of different nodes. Dummy workloads, initially spanning the set of suspicious devices, can additionally span devices that would otherwise be relatively inactive. In this way, application-specific location data can be obscured such that attackers can no longer generate a valid activity heatmap. Furthermore, for applications that are not latency sensitive, a resource manager can move these applications to idle regions in a manner that is cognizant of the resource allocation’s entropy. In this way, we can trade-off latency for security.

C. ADAPT² framework

To enhance resource managers with the capabilities required to shield against the above attacks, we propose ADAPT², a model framework for resource manager extensions that include a state estimator, MTD, and spatio-temporal obfuscation. Figure 6 illustrates the ADAPT² extension to resource management, which is comprised of three main system components. The first is a state estimator that is used to construct a probabilistic model that maps device characteristics and application tasks to a predicting range of resource consumption. The second incorporates MTD by monitoring adaptation history and application mapping in order to determine appropriate times to initiate a randomization. It tracks suspicious devices based on challenge-response information generated from the state estimator to determine appropriate workloads to deploy onto suspicious devices. Suspicious devices can additionally be isolated to enable in-depth testing. Finally, the location obfuscation component uses current resource mappings to identify latent idle regions and have (potentially dummy) workloads scheduled in order to minimize spatio-temporal location leakage as a result of localized activity.

Implementing the ADAPT² model framework as an extension of an existing resource manager prevents the attacks on resource usage discussed in Section III. In doing so, one can reduce the attack surface exposed by the Resource Manager in distributed edge-cloud networks.

V. RELATED WORK

IoT devices have been the target of a number of attacks. DDoS-as-a-service attacks such as the Mirai botnet have shown how the vast quantity of these deployed devices can be exploited as a cyber weapon [13]. As the Internet-of-Things has encompassed more safety critical applications, we have also seen cyber-physical attacks on autonomous vehicles [18].

Securing edge-cloud systems and their applications has received considerable attention [10], [22], [24]. Similarly, the problem of securing IoT networks has also received significant attention in the research community [2], [6], [21]. Fundamentally, the work on IoT and edge network security can be broadly divided into two main subcategories: work illustrating novel attacks, and work introducing mitigation
techniques. To the best of our knowledge, we are unaware of any other work that focuses on exploiting or securing the resource management substrate.

Among the plethora of work considering side channel attacks exploiting IoT and mobile devices, Chen et al. discuss a number of attacks on real-time IoT systems [2] focused on four main attack classes, namely, integrity violations due to malicious code injection, side-channel attacks, attacks on the communication channels, and Denial-of-Service attacks. Chen et al. [3] discuss how IoT devices at home can reveal sensitive information about the users. Mitigation techniques include systems like CellPot [14], a novel honeypot for cellular networks to detect threats and provide defense against malicious mobile devices in the network. Naveed et al. [1] introduce the idea of using Physical Unclonable Functions (PUFs) to provide secure authentication protocols for IoT devices in an ad-hoc network. Other techniques such as Post-Quantum public cryptography systems to secure edge devices [15]. Chen and Xu design a collaborative based edge cloud system where social trust networks are built for managing security risks among the cloud edges due to collaboration [4].

VI. CONCLUSION

In this paper, we enumerated and characterized the side-channel vulnerabilities in distributed IoT systems that arise as a result of network adaptation. We showed how an attacker can falsify utilization statistics for the purpose of manipulating a network’s resource allocation. Furthermore, we illustrated how adaptation can leak a system’s valuable meta-information about an IoT node. We then describe our work-in-progress system, ADAPT 2, a framework for distributed IoT systems that can attest device utilization statistics and obfuscate system meta-information. To detect false utilization statistics, ADAPT 2 uses a state estimation model of the computation workloads for each device to determine if there is a discrepancy in the device’s reported statistics versus the device’s expected statistics. ADAPT 2 can use dummy workloads to isolate suspicious devices and perform the attestation procedure. To obfuscate meta-information, ADAPT 2 can either issue dummy workloads or workloads that are not latency-sensitive onto idle nodes to obfuscate the node-activation characteristics of the distributed IoT system. Future work includes the implementation and evaluation of ADAPT 2.

REFERENCES