Energy Attack on Server Systems

Zhenyu Wu, Mengjun Xie[*] *and Haining Wang*
The College of William and Mary, Williamsburg, VA 23187, USA
{*adamwu, mjxie, hnw*}*@cs.wm.edu*

Abstract

Power management has become increasingly important for server systems. Numerous techniques have been proposed and developed to optimize server power consumption and achieve energy proportional computing. However, the security perspective of server power management has not yet been studied. In this paper, we investigate energy attacks, a new type of malicious exploits on server systems. Targeted solely at abusing server power consumption, energy attacks exhibit very different attacking behaviors and cause very different victim symptoms from conventional cyberspace attacks. First, we unveil that today's server systems with improved power saving technologies are more vulnerable to energy attacks. Then, we demonstrate a realistic energy attack on a standalone server system in three steps: (1) by profiling energy cost of an open Web service under different operation conditions, we identify the vulnerabilities that subject a server to energy attacks; (2) exploiting the discovered attack vectors, we design an energy attack that can be launched anonymously from remote; and (3) we execute the attack and measure the extent of its damage in a systematic manner. Finally, we highlight the challenges in defending against energy attacks.

1 Introduction

Power management is one of the critical issues for server systems nowadays. To date energy cost has become a major factor in the total cost of ownership (TCO) of large-scale server clusters [3, 13]. According to [21], more than 100 billion kilowatt hours, representing a $7.4 billion annual cost, will be consumed by servers and data centers in U.S. by 2011. As the price of hardware keeps dropping while its performance continuously improves, the proportion of energy cost in overall expense of server systems tends to grow even larger [3, 13].

Previous researches on server system power management mainly focus on reducing power consumption while maintaining acceptable quality of service. Numerous techniques have been proposed to improve energy efficiency in a variety of aspects, from low-level hardware features such as processor Dynamic Voltage and Frequency Scaling (DVFS) [10, 14] and hard disk spin-down [7, 12], to high-level system-wise management schemes such as cluster load provisioning [8, 19] and virtual machine consolidation [17]. While these power management advancements have significantly improved power savings [1], they have also opened up spaces for energy misuse. However, the security aspect of server system power management has not yet been paid attention to.

In this paper, we investigate energy attacks, a new type of malicious exploits on server systems. Energy attacks are remotely launched, stealthy attacks that attempt to increase the energy consumption of the victim system non-proportionally to its effective workload. A successfully launched energy attack can cause the victim system to waste a large amount of energy, which in turn becomes waste heat, resulting in significantly increased power and cooling expense, shortened hardware component lifespan, reduced reliability, and sometimes even permanent hardware failure. Current power management and security mechanisms provide virtually no defense against energy attacks.

Energy attacks are distinct from conventional cyberspace attacks in three interrelated aspects: objectives, attacking behaviors, and victim symptoms. First, an energy attack aims solely at abusing server power consumption. It does not attempt to disrupt a victim server's normal services or operations, nor to acquire sensitive information from the victim. Second, an energy attack is mounted in a stealthy manner, because the damage is delivered over a relatively long period of time. The network flow of an attacker is similar to that of a normal client, and there is no high-profile traffic patterns or data

[*] This author is currently affiliated with the Department of Computer Science at University of Arkansas at Little Rock and can be reached at mxxie@ualr.edu.

[1] For example, our study shows that a mainstream server in idleness consumes less than half of the energy consumed in full utilization.

fingerprints left by the energy attack. Third, the victim server would only experience increased power consumptions due to energy attacks, and observe no other anomalies such as tangible performance degradation.

To demonstrate the feasibility of launching an energy attack, we perform a step-by-step design and execution of a realistic energy attack on a Wikipedia mirror server. First, we profile the power consumption of the victim Web server under different page serving conditions, and identify a condition that incurs high energy consumption as a viable attack vector. We then proceed to design an energy attack, achieving stealthiness by leveraging knowledge of human Web browsing behaviors. And finally, we evaluate our design by executing the attack on the victim server and systematically measure the power consumption increases under different load conditions. We observe that the damage of the energy attack is dependent on the workload of the server system. For a victim server under typical workloads, our attack is able to increase its power consumption by 21.7% to 42.3%.

Finally, we argue that fine-grained power measurement is a critical component for differentiating energy attacker from benign users, and the lack of support of which in current server systems makes building effective general purpose defense system against energy attacks quite challenging.

The remainder of this paper is structured as follows. Section 2 presents the background on server system energy saving and the security implication. Section 3 details the design of energy attacks. Section 4 evaluates the threat of the proposed energy attack. Section 5 discusses other attack vectors, attack applicability and defense challenges. Finally, Section 6 concludes the paper.

2 Background

In this section, we first discuss the impact of energy proportional computing on a server system and present power measurements on our own server systems. Then, we describe the threat of energy attacks exposed on today's server systems.

2.1 Energy Proportionality

Energy proportional computing [4] is an important concept in today's server systems. It aims to address the increasing energy concern and demand for power saving by making servers consume energy proportional to its workload. This goal is normally achieved by conditionally trading off component performance for power savings.

Processors are the primary targets for power optimization, because of their high maximum power consumption (hundreds of watts per unit). Nowadays, the majority of server-class CPUs have employed power saving techniques that are already used in desktop and mobile processors, such as DVFS, multiple power states with reduced performance, and even turning off idle cores. Motherboard and chipset feature the shutdown of unused circuitry, and memory chips also have several standby states with reduced power for no read/write cycles. Hard drives can only save a small portion of energy at idleness, due to their power demanding internal mechanical parts (spinning platters). However, they also support "spin-down", shutting down the motor and thereby cutting down the majority of its power consumption, at a high (latency) cost of resuming service.

The ACPI (Advanced Configuration and Power Interface) specifications [1] are introduced to unify the power management of various types of devices in computer systems and provide well defined power management interfaces for both hardware and software. Within the specifications, multiple performance states are defined for a computer component. Each performance state corresponds to a specification of the expected performance and power consumption. At least one state is well defined: a full power state corresponds to the maximum performance. Depending on device type and manufacturing technology, additional number of reduced performance states can be defined.

Although modern operating systems are all capable of utilizing the ACPI to conserve energy under light load or in idleness, previous generations of server systems (such as our System A below) are not very energy proportional. This is because performance and security used to be the primary concerns, and thus the underlying hardware provides little or no support of performance states with reduced power consumption. However, as energy concerns weigh increasingly heavily, today's server systems have been becoming more energy proportional.

2.2 Real Server Measurements

We perform a small measurement study on system power consumption, using two server systems with different generations of hardware configurations, which are listed in Table 1. System A was bought in 2006 and System B was bought in mid-2009. We believe that both servers are representative of the mainstream system configurations at the time of purchase.

We measure the whole system power consumption in three different load scenarios: completely idle (IDLE), processors being fully utilized (CPU), and processors and hard drives being fully utilized (CPU+HDD). The "CPU" workload is generated by running multiple instances of a classic CPU benchmark program "linpack", and the number of instances corresponds to the number of logic cores. The "CPU+HDD" workload is generated

	System A	System B
CPU	2 * Xeon 5130 Dual Core	2 * Xeon 5520 Quad Core
Memory	4 * 1GB DDR2 FBDIMM	6 * 1GB DDR3 FBDIMM
HDD	4 * 7200RPM SATA	6 * 7200RPM SATA

Table 1: Configurations for Server Systems

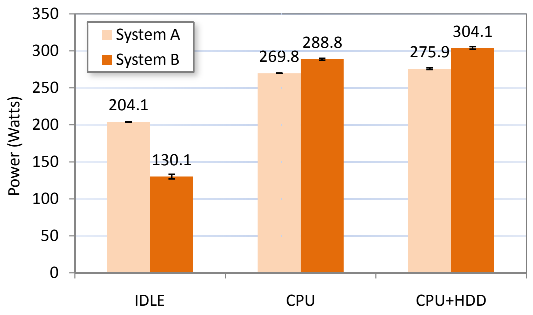

Figure 1: Whole System Power Consumptions

by running the "CPU" workload with the highest `nice` value and, at the same time, writing a large volume of data to the hard drives using the `dd` utility. The power consumption data are collected using "Watts up? .Net" digital power meter [23].

Two observations can be made from our measurement results shown in Figure 1: first, in high utilization scenarios System B (the newer server) consumes slightly more power than System A; second, and more interestingly, in the IDLE scenario, the power consumption of System B is significantly less than that of System A. While the first observation can be explained by System B having increased overall computation power than System A, the second observation presents us the direct proof that newer server system is becoming more energy proportional than previous generations. With higher computation power and improved energy proportionality, one can expect System B to yield more energy saving than System A under the same workload. However, we make an additional, alarming observation when we look at the advancements in energy proportional computing from a security perspective.

2.3 Threat of Energy Attacks

The improved energy proportionality has significantly changed the power profile of today's server systems. For example, our measurement data in Figure 1 shows that compared to IDLE, the CPU+HDD power consumption of System A increases by only 35%, while that of System B increases by 134%. The larger power consumption increase of System B indicates that it has a wider dynamic power range than System A. In other words, the power consumption of System B (energy proportional server) is more alterable than that of System A (non-energy proportional server). The increased power consumption alterability represents a new threat to server systems. The power management mechanism of a server can be attacked by maliciously crafted workloads that target at consuming disproportional amount of energy, rendering the power saving ineffective, and resulting in significant energy waste of a victim.

Alarmingly, we realize that the threat of energy attacks is in fact an exploitable vulnerability because currently there is no effective defense against it. Existing power management schemes mainly focus on improving energy efficiency under normal operating conditions with benign workload, and thus they do not provide any defense against energy attacks. Moreover, most server systems do no have an efficient mechanism to measure power consumption, and thus could not even detect energy attacks, let alone defend against them.

3 Energy Attack on Server Systems

In this section, we demonstrate the feasibility of launching an energy attack. First, we describe the scenario selection and the characterization of energy attacks. We then design a realistic energy attack against an open Web server as a case study, covering the attack vector discovery, exploitation, and detection avoidance.

3.1 Scenario Selection

A great variety of tactics can be used to mount energy targeted attacks against server systems. For example, if attackers obtain "root" or "administrator" privilege on a victim system, they can deliberately mis-configure drivers and/or firmware, e.g., over-clock processor and memory, to operate the hardware components out-of-specs. Even with the privilege of a normal user, attackers can still easily increase the power consumption by running badly behaving programs such as a tight dead loop. However, the above mentioned scenarios are not the focus of our study, because they are generally difficult to implement from remote, due to the high requirements for attackers (e.g., having privileged or physical access to the victim system).

We are interested in more commonly encountered scenarios, in which energy attacks can be launched without any special privileges. We assume that (1) the victim server runs an open service, which accepts service requests from the Internet; (2) the attackers have no physi-

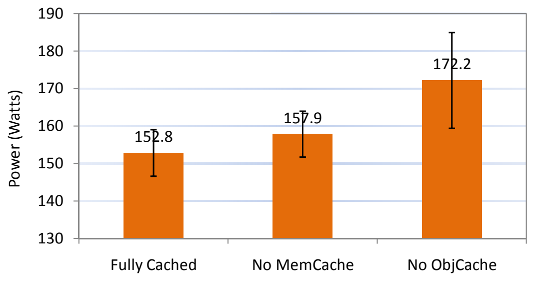

Figure 2: Power Draw vs. Caching Scenarios

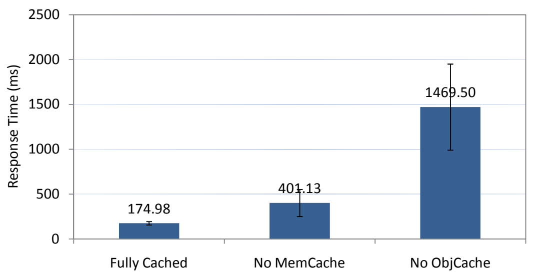

Figure 3: Response Time vs. Caching Scenarios

cal access to the victim server; (3) the attackers only have equivalent privileges of "anonymous users" on the victim server (for example, they cannot change system configurations or execute arbitrary code); and (4) there are no exploitable security vulnerabilities on the victim system to escalate the attackers' privileges. In other words, the attackers communicate with the victim server using the same method as legitimate users, and the major variable they can manipulate is the server's workload by crafting and submitting malicious service requests.

Thanks to the generic setting of attack environment, we believe that our scenarios are applicable to a wide range of servers, particularly, public Web services such as news, blogs, and forums, public data services including file and image sharing sites, and search engines.

3.2 Attack Characterization

Attempting to be stealthy, energy attacks incur their damage in an accumulative fashion over a long period of time. Thus, the key to the success of energy attacks is to be low profile and avoid detection. As a result, energy attacks on a server system must meet two requirements. First, the attack should not exhibit traffic anomalies or have unique traffic patterns, because the server traffic is often monitored for security purposes. Second, the attack should cause minimal performance anomaly on the victim server, as unusual performance degradation is a very visible sign that the server is under attack.

The first requirement precludes high service request rate attacks, due to their obvious traffic anomalies. The malicious requests in an energy attack need to be sent at low to normal rate, and hence should be crafted to ensure a high per-request energy cost. In order to fulfill the second requirement, energy attacks must be adaptive to the workload condition of the victim server. Because the victim hosts an open service, its normal workload tends to vary significantly in time. The workload may be correlated to the day-night and weekday-weekend cycle. Inflicting a fixed malicious workload on the victim may either cause performance anomaly during high-load periods, or fail to incur the maximum energy cost damage.

3.3 Case Study: Wikipedia Mirror Server

We perform a case study of designing an energy attack on an open Web server. We use System B as the victim server, running a mirrored Wikipedia service. The service setup is detailed in Section 4.1. We choose Wikipedia mirror as our attack target because it is a freely available, large single server Web service—a representative of real world production-use open Web services.

3.3.1 Identifying an Attack Vector

The Wikipedia mirror is powered by MediaWiki, a large-scale content management system. The contents of all MediaWiki pages are stored in a marked up format different from standard HTML, and pages are dynamically generated when they are requested. Two levels of caching, object cache and in-memory cache, help to optimize the performance.

MediaWiki stores the dynamically generated HTML contents in an "object cache"—a database table. When a page is requested repeatedly, the HTML content is retrieved directly from the object cache without being repeatedly generated. A cached HTML page expires either after a period of inactivity or the associated page content has been modified. In addition to the object cache, the MySQL database speeds up operations by storing a portion of frequently queried table entries, as well as table search indices and query results in a memory, employing a modified LRU replacement algorithm.

We profile the power consumption and service latency characteristics of the two caching mechanisms on the target server. Figures 2 and 3 show the average power usage and average response time for serving page requests from a single client in three different caching scenarios: pages being fully cached (in both memory and object cache), pages only in object cache, and pages not being cached. The lower bound of Y-axis in Figure 2 is set to 130 watts, the system idleness power consumption. Thus, the columns in the figure represent the additional power consumption caused by the service requests.

From this measurement, we can observe that compared to fully cached requests, requests with memory cache misses incur 3% power increase and 129% processing time increase, and requests with object cache misses incur 12.7% power increase and 840% processing time increase. Because energy is defined as the product of power and time, the effect of cache misses on energy consumption increase is multiplicative. The high energy cost rendered by cache misses forms an effective energy attack vector to our Wikipedia mirror server.

3.3.2 Exploiting the Attack Vector

Our next step is devising a method to exploit the discovered attack vector, that is, to generate requests that can cause cache misses, especially object cache misses. We examine previous studies in Web browsing behaviors. According to Barford and Crovella [2], Web page accesses on a Web server follow Zipf distribution, i.e. access frequency of a page correlates to its rank, and most accesses concentrate on a small number of pages while a large number of pages are rarely accessed. It is clear that the caching mechanisms in our Web server work well in handling such an access pattern because they are designed to optimize for similar access patterns. However, this knowledge also hints a practical cache attack scheme. To generate page requests with high probability of cache miss, we just need to access pages in patterns following a very different distribution from Zipf. For the ease of study and implementation, we choose a uniform random page access pattern to exploit our attack vector.

3.3.3 Detection Avoidance

The selected attack vector enables us to increase the victim's energy consumption without sending a large amount of requests. To avoid generating abnormal traffic patterns, we model the attacking request rate after "normal" Web clients.

Barford and Crovella [2] also show that Web browsing exhibits an "active-inactive" behavioral pattern. During the active period, a client submits requests in a bursty manner, which is attributed to the browser downloading multiple resources (images, scripts, etc.) linked to a document. During the inactive period, the client pauses sending requests, presumably reading the page content. The length of the inactive period follows Pareto distribution.

For our experiments, we simplify our model by "condensing" the active period into a single request, and only model the inactive period for request inter-arrival time. This is because all Wikipedia pages are text oriented and structurally alike. The client behaviors in all the active periods would be very similar.

In addition to traffic shaping, we also need to adaptively adjust the injection of malicious requests based on the workload of the victim server. The server workload can be approximated by the service response time. We build a profile of the victim, correlating the server load with the response time. During the attack, we monitor the response time of the server and adjust the sending rate of malicious requests accordingly.

4 Attack Evaluation

In this section, we first describe the experimental setup setup. Then, we detail the attack preparation, measurements of the energy attack. And finally we assess the achievable damage.

4.1 Configuration and Setup

We set up a Wikipedia mirror server on System B using the classical LAMP (Linux, Apache, MySQL, and PHP) combination. The database is imported from a Wikipedia dump containing 9,053,725 page entries. With a number of tests, we find that the server is capable of caching about 10,000 pages in memory. Therefore, we randomly pick 50,000 pages for use in our experiment.

We simulate client requests using a custom client program running on a desktop computer. The client program simulates multiple clients each running in a separate thread. The "normal" clients are configured to access selected pages following Zipf distribution with $\alpha = 1$, and the request interarrival time follows Pareto distribution with $k = 1$ and $\alpha = 1.5$. The "malicious" clients are configured to access selected pages with uniform random patterns, and have the same request interarrival time distribution as the "normal" clients.

4.2 Workload – Response Time Profile

Before launching the attack, we first profile the victim server and establish the correlation between its workload and response time. We find out that the server is capable of stably supporting up to 100 normal clients and thus define 100 clients as the full workload of the server.

Figure 4 shows the correlation between workload and response time. Each data point is the average of 250 samples of service response time obtained under the corresponding workload. The error bar represents the standard deviation of response time. For light and moderate workloads (up to 50 clients), the server's response time increases quite slowly. When the workload increases beyond 60%, or 60 clients, the response time starts to rise significantly. With workloads in which the number of active clients is beyond 100, the server starts to show symptoms of being overloaded—all clients experience intermittent short burst of request failures in the form of

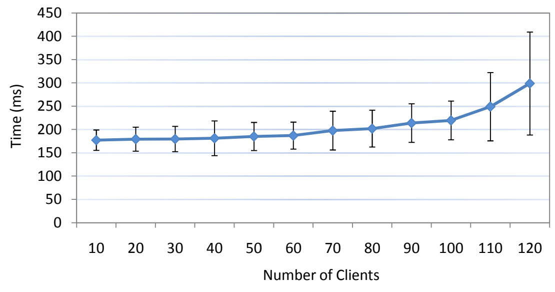

Figure 4: Workload vs. Response Time

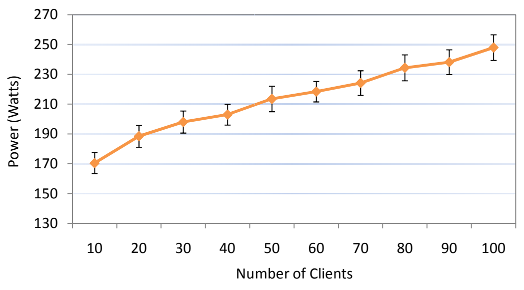

Figure 5: Workload vs. Power Consumption

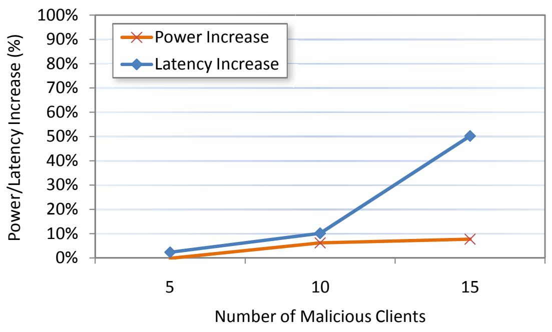

Figure 6: Attack Effect with 100 Normal Clients

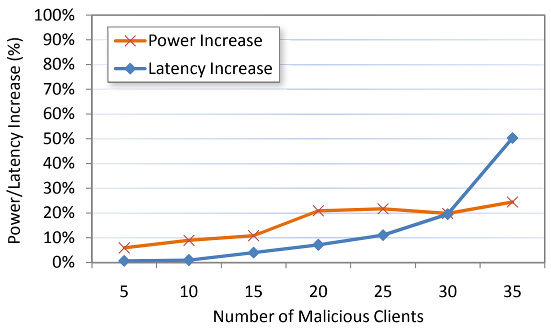

Figure 7: Attack Effect with 50 Normal Clients

"HTTP 500" errors. Figure 5 shows the correlation between stable workload and system power consumption, from which we can see that the server system power consumption is indeed proportional to its workload.

4.3 Attack Measurements

We use server-side power consumption and client-side perceived response latency to measure the effects of the energy attack. We conduct the experiments using different server workloads, which range from 10 to 100 normal clients with the increment of ten clients. For each workload, we inject energy attack traffic by adding a number of malicious clients. Due to the large volume of data, we only present the results corresponding to 100, 50, and 10 normal clients and depict them in Figures 6, 7, and 8, respectively. These figures show the increases in power consumption and response latency caused by the introduction of malicious workloads.

At 100% of the full load, as shown in Figure 6, the response latency of the victim server is very sensitive to the addition of malicious workloads. The malicious workload of ten malicious clients increases the response latency by 7.6%, and the workload of 15 malicious clients increases the response latency by 50.2%. The power consumption, however, does not increase with the response latency, as the server is already fully loaded.

At 50% of the full load, as shown in Figure 7, with 20 malicious clients, the attack results in 20.9% of extra power being consumed while only incurs 7.1% increase in response latency. However, with 30 or more malicious clients, the response latency increase surpasses the power consumption increase.

At 10% the full load, as shown in Figure 8, the energy increase caused by the attack becomes very significant. With 40 malicious clients, the victim server's power consumption increases by 39.0%, while the service response latency only increases by 7.4%.

4.4 Damage Assessment

Our measurement results show that, at any stable workload, energy attacks will cause increased power consumption on the victim server. The more malicious clients, the larger the power increase. However, a larger number of malicious clients also results tangible performance degradation. Figure 9 presents the collective results of service response time increases for all ten different workloads with varying numbers of malicious clients. In this figure, we omit sample points with response time increment larger than 50%.

To guarantee the success of an energy attack, low attack profile takes precedence over the power consumption increment. Therefore, the number of malicious clients need to be limited to avoid significant response time impact. We refer to the workload – response time profile for a reasonable threshold. The standard deviation of response time at stable workloads (10-100 clients)

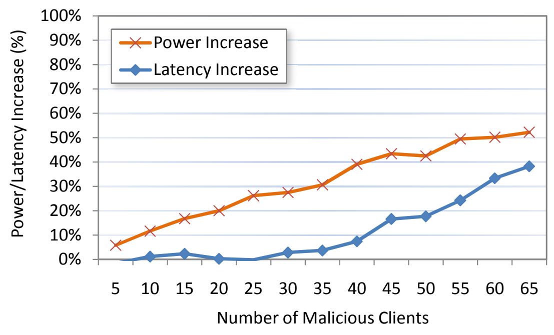

Figure 8: Attack Effect with 10 Normal Clients

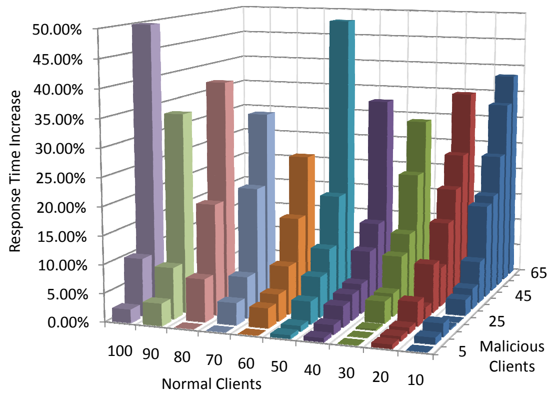

Figure 9: Attack Resulted Response Time Increases

Utilization	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
Power Increase	39.0%	42.3%	36.3%	31.6%	21.7%	14.8%	11.6%	9.0%	11.3%	6.2%

Table 2: Percentage of Power Increases due to Attack

varies between 12.3% and 21.1% of the measured values. We set the response time increment threshold to the smallest percentage, 12.3%.

With the chosen response time increase constraint, for each workload, we determine the maximum power consumption achievable by the attacks and present them in Table 2. We observe that, the power increase effect of the energy attack is inversely correlated to the workload of the server – an idle server suffers significant extra power consumption, while a very busy server only incurs a small power consumption increase.

To assess the gross damage of the energy attack to a typical server, we refer to the study of typical server workloads. Barroso and Hölzle [4] observe that most servers have average utilization between 10% and 50%. Correspondingly, under such utilization, our energy attack can result in 21.7% – 42.3% power consumption increase.

5 Discussion

In this section, we first describe other possible energy attack vectors, the applicability of energy attacks, and then we discuss the challenges of defending energy attacks.

5.1 Attack Variations

Besides using cache miss as an attack vector, energy attacks can also be launched by exploiting other energy related vulnerabilities.

For example, a file depositing server running an unmodified Linux kernel and allows users to control the names for stored files (such as a public FTP server) is vulnerable to energy attacks. The attacker can exploit a well known *nix kernel file name resolution vulnerability [2], and launch a low-rate algorithmic complexity attack [6, 9] to stealthily increase processor utilization. Because a file depositing service is storage and network bandwidth bound, a well-controlled energy attack can avoid generating any throughput anomalies.

Besides the processors, other components with large dynamic power range can also be exploited by energy attacks. For example, hard drives normally consume 12 to 16 watts during operation, but their power consumption can be reduced to under one watt by spin-down the platters during long period of idleness. As a result, an energy attack on hard drives can be mounted by performing sleep deprivation attack to prevent expected spin-down. Although the energy cost of a single attacked hard drive seems to be insignificant, the damage can accumulate to a significant amount when the energy attack targets at a decent sized storage server with 10 to 20 installed hard drives.

5.2 Orthogonality to DoS attacks

Energy attacks may seem to be connected to DoS (Denial of Service) attacks [15, 20, 22], as they seemingly share some related vulnerabilities, such as cache exploits and algorithmic complexity weakness. However, they are orthogonal classes of attacks.

On one hand, DoS attacks have mixed energy effects. This is because the introduction of DoS attacks to server

[2] A simple hash data structure is used by the kernel for file name caching and lookup. By maliciously naming files, one can cause a large number of file names collide onto the same hash slot, resulting in expensive linear searches for file name related operations.

systems pre-dates the era of energy proportional computing, and thus energy was never an attack target when servers have constant power consumption whether busy or idle. As an intuitive example, a TCP SYN flooding DoS attack exhausts the victim server's socket resource, and thus prevents the victim from receiving normal service requests. This attack causes most components of the victim server to become idle, and thus significantly reduces its power consumption. On the other hand, energy attacks would never attempt to cause denial of service. To the contrary, it tries hard to avoid causing denial of service, because staying low-profile and undiscovered is critical to a successful attack. Therefore, energy attacks and DoS attacks are distinct in terms of their designed purpose, execution methodology and effects.

5.3 Attack Applicability

We have thoroughly investigated the proposed energy attack against a standalone server system. We use the case of single standalone server as the first step to study energy attack, because it is relatively easy to perform a clear analysis and repeatable evaluations. However, the attack vectors on a standalone server are not applicable to other hosting configurations, such as clustered servers and load balanced server farm. For example, our proposed energy attack on our Wikipedia mirror server is not effective on the actual Wikipedia website, which employs load balanced server clusters and heavy proxy caching techniques. However, we believe energy attacks also pose serious threats to large scaled systems, such as cloud hosting environment [11]. Competing cloud vendors may use energy attack as a powerful weapon to increase the operation cost of their opponents, making the attackers' service rates more attractive. To extend the scope of this work, we plan to study and profile the interactions of workload and power consumption of server clusters, discover viable attack vectors, as well as devise defending techniques.

5.4 Challenges of Defense

To defend against energy attacks, it is necessary to measure the amount of energy consumed by a user's requests and use it to differentiate malicious users from benign users. Therefore, measuring and accounting power consumption for processing each request is a fundamental requirement. Unfortunately, even though it is possible to measure the power consumption of the whole system in a coarse time granularity (e.g., using a power meter), there is no field-deployable mechanism available for fine-grained power measurement.

Neugebauer and McAuley [18] suggest using performance counter data such as CPU cycles, disk operations, and screen pixels to approximate power consumption for laptops and mobile devices. Buennemeyer *et al.* [5] present a battery-sensing intrusion protection system for mobile computers, which correlates device power consumption with Wi-Fi and Bluetooth communication activities. Kim *et al.* [16] propose a power-aware malware detection framework by collecting application power consumption signatures.

These techniques, however, are hardly applicable to a server system. This is because mobile devices are designed to be used by individuals, and they run few applications concurrently. In contrast, server systems are designed to process a large number of requests from multiple users in parallel. As a result, power consumptions of server systems are heavily correlated with the *collective* service requests coming from the network, from which one hardly extract signatures of *individual* users. In addition, performance counter readings on server systems (especially at fine granularity such as per-request processing) of independent processes can be heavily coupled and inaccurate for power approximation. For example, an SMT (Simultaneous Multi-Threading) processor allows two or more threads to execute in parallel, sharing the same underlying hardware. This may lead to unrelated processes competing for processor resources and interfering with each other's cycle count readings. Another example is that modern hard drives can intelligently reorder the sequence of operations to improve efficiency; however, this can cause the operation latency disproportional to the request data size.

6 Conclusion

Server systems have become more power efficient and energy proportional as power management technologies advance. However, the security aspect of power management has not yet been studied. In this paper, we investigated the potential vulnerabilities in server power management. First, we exposed the threat of energy attacks by measuring the power consumption of real server systems. Then, we designed and evaluated energy attacks on server systems. In particular, we validated the threat of energy attacks on an open Web server running Wikipedia mirror service. By profiling power consumption of the target server under different operation conditions, we realized a viable energy attack vector. We conducted a series of experiments, in which energy attacks with varying attack intensities were carefully mounted to avoid incurring tangible degradation of server performance. Our experimental results show that the proposed energy attack can incur 21.7% — 42.3% additional power consumption on the victim server. Finally, we discussed the challenges in protecting victim servers against energy attacks.

References

[1] Advanced configuration and power interface. `http://www.acpi.info`, 2009.

[2] P. Barford and M. Crovella. Generating representative web workloads for network and server performance evaluation. In *Proceedings of the 1998 ACM SIGMETRICS*, pages 151–160, 1998.

[3] L. A. Barroso. The price of performance. *ACM Queue*, 3(7):48–53, September 2005.

[4] L. A. Barroso and U. Hölzle. The case for energy-proportional computing. *IEEE Computer*, 40(12):33–37, Dec. 2007.

[5] T. K. Buennemeyer, M. Gora, R. C. Marchany, and J. G. Tront. Battery exhaustion attack detection with small handheld mobile computers. In *Proceedings of the IEEE PORTABLE*, 2007.

[6] X. Cai, Y. Gui, and R. Johnson. Exploiting unix file-system races via algorithmic complexity attacks. In *Proceedings of the 30th IEEE Symposium on Security and Privacy*, 2009.

[7] E. V. Carrera, E. Pinheiro, and R. Bianchini. Conserving disk energy in network servers. In *Proceedings of the 17th ICS*, pages 86–97, 2003.

[8] J. S. Chase, D. C. Anderson, P. N. Thakar, A. M. Vahdat, and R. P. Doyle. Managing energy and server resources in hosting centers. In *Proceedings of the 18th ACM SOSP*, pages 103–116, 2001.

[9] S. A. Crosby and D. S. Wallach. Denial of service via algorithmic complexity attacks. In *Proceedings of the 12th conference on USENIX Security Symposium*, 2003.

[10] M. Elnozahy, M. Kistler, and R. Rajamony. Energy conservation policies for web servers. In *Proceedings of the 4th conference on USENIX USITS*, 2003.

[11] X. Fan, W.-D. Weber, and L. A. Barroso. Power provisioning for a warehouse-sized computer. In *Proceedings of the 34th ISCA*, pages 13–23, 2007.

[12] S. Gurumurthi, A. Sivasubramaniam, M. Kandemir, and H. Franke. Drpm: Dynamic speed control for power management in server class disks. In *Proceedings of the 30th ISCA*, pages 169–182, 2003.

[13] J. Hamilton. Where does the power go and what to do about it? In *Proceedings of the USENIX HotPower*, 2008.

[14] T. Horvath, T. Abdelzaher, K. Skadron, and X. Liu. Dynamic voltage scaling in multitier web servers with end-to-end delay control. *IEEE Trans. Comput.*, 56(4):444–458, 2007.

[15] S. Kandula, D. Katabi, M. Jacob, and A. Berger. Botz-4-sale: Surviving organized ddos attacks that mimic flash crowds. In *Proceedings of the 2nd USENIX NSDI*, 2005.

[16] H. Kim, J. Smith, and K. G. Shin. Detecting energy-greedy anomalies and mobile malware variants. In *Proceeding of the 6th MobiSys*, pages 239–252, June 2008.

[17] R. Nathuji and K. Schwan. Virtualpower: coordinated power management in virtualized enterprise systems. In *Proceedings of the 21st ACM SOSP*, pages 265–278, 2007.

[18] R. Neugebauer and D. McAuley. Energy is just another resource: Energy accounting and energy pricing in the nemesis os. In *Proceedings of the 8th USENIX HOTOS*, 2001.

[19] E. Pinheiro, R. Bianchini, E. V. Carrera, and T. Heath. *Dynamic cluster reconfiguration for power and performance*, pages 75–93. Kluwer Academic Publishers, Norwell, MA, USA, 2003.

[20] S. Ranjan, R. Swaminathan, M. Uysal, and E. Knightly. Ddos-resilient scheduling to counter application layer attacks under imperfect detection. In *Proceedings of the 25th IEEE INFOCOM*, 2006.

[21] U.S. Environmental Protection Agency. Report to congress on server and data center energy efficiency, 2007.

[22] H. Wang, C. Jin, and K. G. Shin. Defense against spoofed ip traffic using hop-count filtering. *IEEE/ACM Transactions on Networking*, 15(1), Feb. 2007.

[23] Watts up? Watts up? .net digital power meter. `https://www.wattsupmeters.com/secure/products.php?pn=0`, 2009.