Exploiting Cloud Utility Models for Profit and Ruin

Joseph Idziorek
Department of Electrical and Computer Engineering
Iowa State University
Ames, IA
idziorek@iastate.edu

Mark Tannian
Department of Electrical and Computer Engineering
Iowa State University
Ames, IA
mtannian@iastate.edu

***Abstract*—This paper discusses an attack on the cloud computing model by which an attacker subtly exploits a fundamental vulnerability of current utility compute models over a sustained period of time. Internet-accessible cloud services expose resources that are metered for billing purposes. These resources are subject to fraudulent resource consumption that is intended to run up the operating expenses for public cloud service customers. The details and significance of this attack are discussed as well as two detection methodologies and there respective experimental results. This work investigates a potentially significant vulnerability of the cloud computing model that could be exploited from any Internet connected host. Well-crafted transactions that only differ in intent but not in content are challenging to differentiate and thus this attack may be difficult to detect and prevent.**

***Keywords*-cloud computing; utility compute model; fraudulent resource consumption attack; application-layer DDoS; anomaly detection;**

I. Introduction

Computing services that were traditionally hosted on organizations' servers and networks are being outsourced to third-party Cloud Service Providers (CSPs). Transferring sensitive data and computing operations outside of a trusted environment raises obvious concerns of confidentiality, integrity and availability for all aspects of a CSP's service platform. Initial focus, research, and threat modeling has concentrated on both the confidentiality and integrity of data stored and computed in the cloud. Absent from these works is an analysis of the external threat sources that have the ability to affect the availability of cloud-based services and exploit the integrity of the billing model that governs this emerging computing model.

While semantic and flooding DDoS attacks are well known and the associated risks are well researched [1], this work will explore a subtle attack more akin to an application-layer DDoS attack. The threat-source considered in this paper is an attacker who seeks to fraudulently consume bandwidth and computational resources of Web-based cloud services that in turn incur a financial burden on the cloud consumer. Utility computing is particularly vulnerable to an attack by which the attacker seeks to exploit the utility pricing model in order to financially harm the victim.

The attack scenario depicted in Figure 1 illustrates a vulnerability of the cloud utility model. A botnet consisting of potentially thousands of bot clients under skillful control of the botmaster can consume cloud resources by mimicking legitimate client behavior.

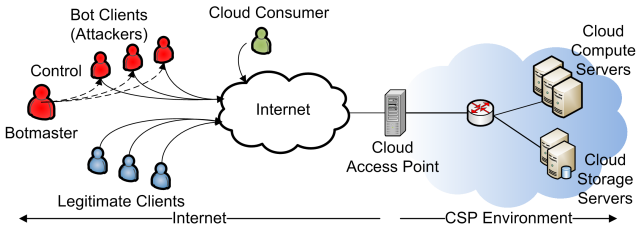

Fig. 1. Cloud Attack Network Model

In this paper, two methodologies to detect Fraudulent Resource Consumption (FRC) attacks on metered Web resources are presented. The first methodology applies the properties of Zipf's law in the analysis of aggregated user consumption patterns. The second approach is an entropy-based methodology that detects anomalous behaviors in attacker request dynamics. The experimental results demonstrate that both methodologies show promise of being effective at detecting FRC attacks.

The rest of the paper is organized as follows. Section II provides the background for the cloud computing model. A detailed description of the FRC attack is described in Section III and the detection methodologies as well as experimental results are presented in Section IV. Related areas of work are analyzed in Section V. Finally, future work and the conclusion are presented in Sections VI.

II. Background

The attack described in this paper is not unique to the cloud model. However, the cloud model does provide a well-documented and practical application of the utility computing model to demonstrate such an attack. This section provides a brief overview of cloud computing and the utility model.

A. Cloud Computing

To keep the terminology straight throughout the rest of the paper, the following roles are defined:

- *Cloud Service Provider (CSP)* - The CSP (e.g. Amazon or Microsoft) offers client-provisioned and metered computing resources that can be leased for flexible time durations.
- *Cloud Consumer* - The cloud consumer is a person or organization that employs the services of a CSP and is financially responsible for resource consumption.

978-0-7695-4460-1/11 $26.00 © 2011 IEEE
DOI 10.1109/CLOUD.2011.45

IEEE computer society

- *Client* - The client is a legitimate user that requests services offered by the cloud consumer.
- *Attacker* - The attacker is a malicious user that fraudulently consumes resources offered by the cloud consumer.

B. Utility Compute Pricing Model

The utility model, by which cloud services are offered, enables the attractive payment model of pay-as-you-use. Customers pay only for the resources they consume and only for the time that they consume them. The flexibility that this model facilitates is advantageous to a cloud consumer because of the low cost of entry and avoidance of major capital expenses. Table I represents a subset of the direct costing metrics established by Amazon's Elastic Compute Cloud (EC2) platform [2]. Although cloud computing makes for a compelling use case of the utility model, the concept of utility computing has origins that date back to time-sharing on mainframe computers.

TABLE I
AMAZON EC2 PRICING METRICS FOR A LARGE LINUX INSTANCE RESIDING IN NORTHERN VIRGINIA

Standard Compute Instance	
Large Linux	$0.34 per inst. hour
Data Transfered In	
All Transfered In	$0.10 per GB
Data Transfered Out	
First 1 GB per Month	$0.00 per GB
Up to 10 TB per Month	$0.15 per GB
Next 40 TB per Month	$0.11 per GB
Next 100 GB per Month	$0.09 per GB
Over 150 GB per Month	$0.08 per GB

As seen in Table I, the cost of computing in the cloud is billed in units of Cost-Per-Hour (CPH) consumed and Cost-of-Data-Transferred (CDT) in and out of the CSP's environment. The total cost of computing in the cloud model can then be modeled as follows:

$$Total\ Cost = CPH(hours) + CDT(bytes) \qquad (1)$$

Equation 1 represents the cost of cloud resources as a generic model that allows for the analysis of consumer costs independent of a particular CSP and their respective cost metrics.

III. FRAUDULENT RESOURCE CONSUMPTION (FRC) ATTACK

This section provides a conceptual foundation for the FRC attack. In order to provide a comprehensive explanation, the target of the attack, the threat model, an attack description and an exploration of the direct costs associated to this attack are provided.

A. Target

For the purposes of this paper, the target of this attack is a Website or Web application hosted in a third-party CSP environment. The CSP will generate revenue by providing hosting services on a utility model basis. In this service environment, resources consumed by clients result in a direct cost to the cloud consumer.

The characteristics of the Websites considered in this paper are those that have been designed to be serving predominantly public Web content that is accessible to Internet users. Although the use of authentication on the site would significantly reduce the amount of content readily available to the general anonymous public and thus potentially restrict the amount of exploitable resources, this Website feature is not considered for it is assumed the cloud consumer desires to host public content. It has also been assumed that the target Website is hosted in an environment in which the Web server is properly configured, patched and is protected behind a firewall that conforms to a well-considered information security policy and employs best practice filtering techniques.

An additional characteristic is that the Website does not make use of reverse-Turing tests [3] to differentiate humans from zombie computers. The use of such tests is detrimental to the overall goals of the cloud consumer, as these types of tests will result in a certain percentage of legitimate users choosing not to solve the puzzles as well as preventing search bots from indexing the site's content. Therefore employment of such techniques to restrict access to public-facing Web content is considered excessive and not considered in this work.

In an effort to simplify the experimental design used to assess the chosen detection techniques, the scope of HTTP protocol request methods is limited to HTTP GET requests. All HTTP request methods consume resources on the server and supporting network, but do vary by degree of consumption. These consumption distinctions would be necessary to consider if a precise cost calculation is desired, but for the purpose of this initial paper on the FRC attack the focus is on the general relation between client actions and direct cost to the cloud consumer.

B. Threat Model

With a black market of hackers or botnets for hire [3], a threat-source is not required to be capable of performing the attack themselves. Whoever performs the attack will require sufficient compute resources and bandwidth to implement a sustained and significant resource utilization attack, which is fairly easy with current computer technology and bandwidth.

In the past, Internet attacks were generally regarded as being less financially motivated and driven by attackers need for self-fulfillment, political motivation, fun, or proof of skill [4]. Today, however, cyber criminals have been moving towards making a profit. The motivation of the hired attacker in this threat model is purely financial and the attacker benefits directly from either from a service fee or from an extortion fee paid by the victim. The original threat-source, who hired the botnet master, achieves their objective by decreasing the economic health of the victim.

In this threat model, the attacker will factor in time for attack completion, attack success likelihood, and attack detection as key variables as resources are allocated and as an attack methodology is chosen. By understanding the utility

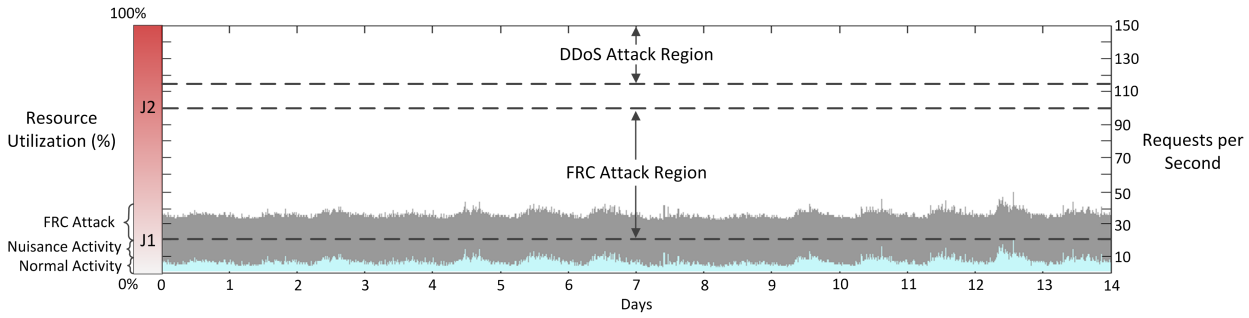

Fig. 2. Malicious Resource Utilization Continuum

models published by CSPs, the attacker can determine what transactions or actions will cost. Although optimal attack strategies are outside the scope of this paper, a wise attacker will construct an attack to consume significant amounts of resources but will stay clear of extreme resource consumption to avoid detection. The attacker will craft proper functional transactions in order to ideally exercise as many billable resources per transaction as possible while remaining undetected. Unlike a SYN flooding attacks that seeks to consume available socket resources by forming numerous incomplete TCP connections, fully established connections are much more effective at consuming large volumes of resources while remaining undetectable by current signature and anomaly-based detection mechanisms.

The threat-source attacks by generating Web traffic consistent in comparison to legitimate traffic. However, it is assumed that the threat source does not have the ability to access historical records/logs from the victim's servers nor the ability to insert a traffic-logging device in front of the victim's Web-based services. Although collusion with an inside person would do away with the previous assumption, these fairly realistic restrictions prevent the attacker from creating statistically indistinguishable traffic patterns in comparison with legitimate traffic seen by the site. From the victim's perspective, malicious and legitimate traffic are interwoven and only differs in intent not content.

C. Attack Description

As evidenced by recent trends in DDoS attacks, attackers are employing the services of botnets consisting in size of upwards of tens of thousands of compromised hosts and are using these botnets as an attack medium [3], [5]. To increase effectiveness and circumvent current detection mechanisms, attackers are moving away from network-layer attacks such as SYN floods and attacking application-layer resources by means of HTTP flooding attacks. This paper and the description of the FRC attack anticipates a natural evolution of these attacks on the computing resources metered for utility pricing as found in the cloud computing model.

The recent emergence of cloud computing and its attractiveness has raised the prospect that current utility model structures may prove to be a significant vulnerability to cloud consumers. The attack is simply a matter of making properly formed and seemingly legitimate requests for application services in sufficient quantity that expenses accumulate over time to a level that is unsustainable for the cloud consumer. One key objective of the attacker is to blend into the noise of legitimate activity so that their malicious resource consumption is undetected by current measures and their activities remain unimpeded. What makes the attack unsustainable for the victim is that the victim's business objectives for the cloud-based services are not achieved regardless of the disproportionate amount of expenses paid.

In order to describe this attack more precisely, one could consider a continuum of Malicious Resource Utilization (MRU) as seen in Figure 2. Reading from bottom to top, the y-axis on the left-hand side of the figure depicts a gradual increase of resource utilization (%) for a busy NASA Web server over a two-week duration of time (x-axis). The y-axis on the right-hand side of the figure denotes the number of requests per second experienced by the Web server. Because of the historical nature of the data set, a direct mapping of this relationship is not known, but is depicted to represent a conservative estimate given the capacity of modern day Web servers.

Initial attack intensity above normal activity is a range labeled nuisance activity because the resultant costs are insignificant for the cloud consumer. However, as the malicious activity intensifies beyond the nuisance activity range, the malicious costs to the cloud consumer start to become a matter of concern. This transition point is labeled J1. Malicious activity that exceeds J1 enters into the FRC Attack Region. Within this region, a FRC attack is neither nuisance activity nor does it significantly degrade the QoS of the Web server. With a utility model assigning costs for all resources consumed, this region is of interest to an attacker who wishes to inflict economic pain. If the attack intensity increases above J2, the aggregate resource consumption will reach a point when the QoS starts to significantly degrade as the increase resource utilization results in an increase in system response latency. It is at this point detection like, as stated in [3], [6], [7], current application-layer DDoS detection and mitigation schemes will be activated. The transition point between the inability and ability to detect malicious resource consumption is denoted as

J2 on Figure 2. The initial objective of the FRC attack research is to improve detection sensitivity and push J2 closer to J1 by improving detection of attacks that appear as legitimate traffic and transactions, but differ in the requestor's intent.

Availability in the context of this discussion is not a binary measure in which the system is nearly incapacitated at the time of the attack. The technical infrastructure of a Website and its provider will have no trouble functioning while the FRC attack is underway. Instead, availability is a long-term consideration defined as the cloud consumer's ability to withstand the financial consequences of such an attack over a prolonged period of time. Unlike a short-lived DDoS attack, the duration of a FRC attack is intended to last weeks or months. As shown in Figure 3, a FRC attack is similar to a slow-and-low approach in which the costs for resources consumed maliciously are additive to that of normal traffic. The challenge FRC attack raises is that of detecting malicious activity that blends in with normal behaviors with the intention of subtly exploiting the resource sensitivity of current utility pricing models in order to incrementally increase operating costs thus inflicting financial damage.

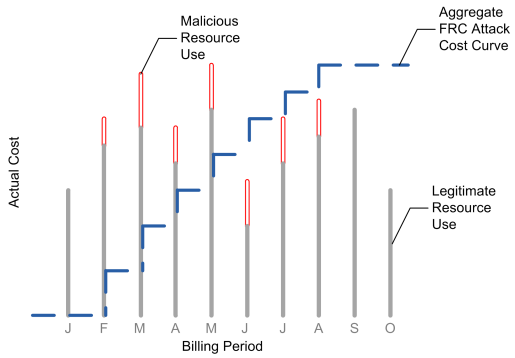

Fig. 3. Aggregate FRC Attack

D. Direct Cost of a FRC Attack

DDoS attacks have always resulted in some from of financial loss for the victim. Whether directly or indirectly, the victim experiences loss when legitimate clients are not able to access revenue generating services, productivity is halted, or the service or corporation's reputation is damaged. Not until resource consumption was directly billable to the consumer, as is the case in the cloud model, was it possible to associate a direct monetary cost to the compute resources consumed during a DDoS attack. The focus of this work can be seen as a much more subtle variation of a DDoS attack. Because of this, both FRC and DDoS attacks on services hosted in the cloud model can be modeled in terms of resources consumed and the resulting monetary loss for the victim.

Regardless of the motive, an attacker and cloud consumer must consider the same set of parameters in order to calculate the expected or actual cost of resources consumed in the cloud. The one distinction between these calculations is that the attacker only considers resource usage in excess of normal activity while the cloud consumer must account for the total cost of all resources consumed despite the intention of the requestor.

TABLE II
CSP BANDWIDTH COST PARAMETERS

Parameters	Description
Number of clients (δ)	Number of distinct clients requesting resources
Average resource size (ϕ)	Average size in bytes for each outbound resource request
Average request size (ρ)	Average size in bytes for each inbound request
Request frequency (θ)	Requests per time period
Time duration (γ)	Time elapsed between the beginning and end of an observed period
Cost model (μ)	CSP pricing model
Cost function f(D_{in},D_{out},γ,μ)	Cost of resource consumption

From the parameters presented in Table II, the total amount of data transferred into the cloud consumer's environment via HTTP GETs over a given time period (γ) can calculated as $D_{in} = \rho \cdot \delta \cdot \gamma \cdot \theta$ and the corresponding amount of data transferred out during the same time period can be calculated as $D_{out} = \phi \cdot \delta \cdot \gamma \cdot \theta$. The subscripts "N" and "A" are used to differentiate between normal activity and resources consumed as part of a FRC attack respectively.

$$Base\ Cost = f(D_{in_N}, D_{out_N}, \gamma, \mu) \quad (2)$$

$$Total\ Cost = f(D_{in_N} + D_{in_A}, D_{out_N} + D_{out_A}, \gamma, \mu) \quad (3)$$

$$FRC\ Attack\ Cost = Total\ Cost - Base\ Cost \quad (4)$$

Tiered costing models such as the one used by Amazon's EC2 (Table I) require that the FRC Attack Cost be calculated as the Total Cost of all activity minus the Base Cost. Data consumed during a FRC attack is additive to that of normal activity and the FRC Attack Cost cannot be accurately calculated without knowing the Base Cost.

The consequences of a FRC attack may be best illustrated by quantifying the cost of an attack based on a realistic scenario. Next is a scenario of a FRC attack on a Web service hosted on Amazon's EC2. Proposing hypothetical attacks in conjunction with the FRC Attack Cost highlights the potential impact FRC attacks can have on a Web-based resources hosted in the cloud.

1) Scenario - EC2: Google calculates that the average Web page currently found on the Internet is 320KB in size [8]. Assuming this is the average page size of the cloud consumer's Website, which consists of multiple distinct pages, an attacker is able to consume on average 320KB per HTTP GET request. In this scenario, normal activity is assumed to be 1TB of data per month resulting in a Base Cost of $153.45. At the rate of requesting one page per minute every minute for a month, a single attacker is able to consume approximately 13 GB of data. Applied to the FRC Attack Cost equation, this attack

alone results in a charge of $2.04 for data transferred in and out of the cloud consumer's environment.

TABLE III
SINGLE ATTACKER SCENARIO

Parameters	Value
Number of clients (δ)	1
Average resource size (ϕ)	320KB
Average request size (ρ)	1KB
Duration (γ)	31 days
Request frequency (θ)	1 req/min
Cost model (μ)	Amazon EC2
FRC Attack Cost	$2.04

The cost accrued from a single attacker at the given rate in Table III would likely be characterized as nuisance activity found below J2 as established on the MRU continuum (Figure 2). Although a non-zero cost, this malicious resource consumption is likely to blend in with the noise of an average monthly service bill.

TABLE IV
MULTIPLE ATTACKER SCENARIO

Parameters	Value
Number of clients (δ)	1000
Average resource size (ϕ)	320KB
Average request size (ρ)	1KB
Duration (γ)	31 days
Request frequency (θ)	200 reqs/day
Cost model (μ)	Amazon EC2
FRC Attack Cost	$283.81

In the next attack variation, the number of active bots in the attack is increased and each bot performs with a request frequency of 200 transactions per day. These adjustments change the magnitude of the attack from nuisance activity to an attack intensity above J1 and thus into the FRC attack region. The consequence of the utility model vulnerability as presented in Table IV is a bit more apparent. If the attack were distributed throughout the course of a month, even at 200 requests per day from a 1000 bots, the attack does not begin to significantly degrade the QoS of the Website (4 x 1.2 GHz 2007 Xeon CPUs, 7.5 GB RAM, 850GB storage) - assuming the system was designed with sufficient performance headroom for the normal activity.

As seen by the attack scenarios in the case study of Amazon's EC2, a FRC attack can inflict a noticeable financial burden on the cloud consumer over time. As the number of attacking resource consumption bots increases, damaging attacks can be mounted without being able to associate a significant usage footprint to a single client, as compared to what was seen in the single attacker scenario. Because current detection efforts are focused on excessive amounts of HTTP requests over a short period of time [5], [6], [7], as is the case in DDoS attack and flash crowds, it is likely that FRC attacks will go undetected.

One important observation worth mentioning is the relative cost of conducting the FRC attack. Although there may be costs associated lease time or attack period that is governed by attack intensity, the amount of bandwidth and compute cycles for a bot to perform 200 requests per day (i.e. on average, one request every 432s) over 31 days is a mere fraction of what a reasonably modern computer is capable of producing. So there should be sufficient performance margin for many more requests per second for each bot being devoted to the attack or other attacks concurrently.

IV. DETECTION METHODOLOGY

In this section two FRC attack detection methodologies will be discussed. The realm of FRC attack detection has been largely unexplored. The methodologies explored here are an initial attempt to push J2 down the Malicious Resource Consumption Continuum towards J1. By pushing J2 to a lesser intensity, attackers will need to sustain their attack longer to achieve the same cost impact or increase the number of bots needed to mount a successful attack. As performance of these initial detection methods are evaluated, the common concerns of computational efficiency and overall detection latency are less of a factor for detection of a long duration attack like the FRC attack.

A. Zipf's Law

The objectives of this section are to investigate an application of the properties of Zipf's law [9] for detecting anomalies in Web request logs as well as to discuss experimental design and empirical results of applying Zipf's law to the detection of FRC attacks.

An effective anomaly detection approach could be a useful indicator for fraud in metered Web services. In the past, Zipf's law has been used for detection of anomalous patterns in large data sets such as in detecting blog spam [10] and accounting fraud [11]. With respect to applying Zipf's law to the Web, Zipf-like distributions of Web logs have been used for modeling and formalizing Web caching models and algorithms [12]. A key property of Zipf's law is that it allows for the broad analysis of very large sets of data.

Consider a Web server log that contains user-generated request records for a Website consisting of multiple Web pages. Let N be the total number of requested Web pages over an elapsed period of time and let f_i be the frequency of requests for the i^{th} of M Web pages. Let all pages and their respective corresponding request frequencies be ranked in descending order of their popularity where the most frequently referenced page is assigned the rank of one and the i^{th} page is the i^{th} most popular page. If Zipf's law holds, the frequency of a request for the i^{th} most popular Web page is inversely proportional to the rank of the page and is represented as follows:

$$f_i \propto \frac{1}{i} \tag{5}$$

It has been shown through research efforts [7], [12] that Web page requests generally following a Zipf-like distribution where the frequency of a request for the i^{th} most popular page is a power-law function such that:

$$f_i \propto \frac{1}{i^{\psi}} \tag{6}$$

Common distributions of Web page requests are not truly consistent with Zipf's law as Zipf's law states that ψ is unity. However, Web page requests tend to be consistent with a more general Zipf-like distribution that allows ψ to be close to but not unity. When plotted on a log-log scale as a rank-frequency plot, the observed slope (ψ) of the best-fit line, when applied to different Web page request traces, is typically negative with a value not unity but close to unity.

Experiments were performed using data sets from Web request logs produced by Iowa State University's public Web server (www.iastate.edu) over the course of nine consecutive weeks from late 2010. The first week of data served as the training data set and was used as the model of the site's normal Web page access patterns. The subsequent weeks served as test data sets as well as the background traffic in which synthetically generated attack patterns were inserted.

The evaluation consisted of generating an attack through synthetic construction of malicious requests and then interleaving these requests within the test data sets. The advantage of synthetic generation is that it enables expedient testing of a number of scenarios and request patterns of attackers in order to test the limitation of the Zipf distribution based detection approach. In order to emulate a FRC attack, synthetic requests were generated to consume a percentage of Web-based resources above that of normal activity. The synthetic request construction methodology assumed that the attacker had *a priori* knowledge of the magnitude of normal Web page requests for a given site. Although contrary to Section III-B, such an assumption was made to allow the attacker to generate more challenging attacks that start at J1 and increase in intensity up the MRC (Figure 2) continuum. The Web request pattern generated was the union of normal and attack Web requests. For each week of test data, five synthetic attack data sets were constructed in addition to the original data set. In the attack data sets, the attack modeled uniformly random page requests totaling 5%, 10%, 20%, 30% and 40% more requests than that of the original data set for a given week.

The detection methodology compared the training data set and each of the test data sets using Analysis of Covariance (ANCOVA). The first step was to compute the Zipf-like distribution for the training data set and each test data set. The second step was to determine if there was a statistically significant difference between the slopes of the two Zipf-like distributions under examination. This required the computation of a linear regression line for each distribution. The slopes of the respective linear regressions are compared with the statistical hypothesis that the slopes are the same. If analysis indicated that the slopes were significantly different, then this result was interpreted to signify that a sufficiently large Web page request pattern anomaly took place thus performing as the fraud detection threshold. Given the lack of specific attacker attribution, additional detection techniques could then be employed.

To measure the effectiveness of this application of Zipf's law for the 48 tests performed, a confusion matrix of detection results is given in Table V. Preliminary empirical results show that the methodology produces a False Positive Rate (FPR) of 5% and a False Negative Rate (FNR) of 11%. As a result, this broad analysis methodology appears to be an effective way to examine large sets of Web logs for the purpose of detecting possible fraud motivated access patterns in Web request logs.

TABLE V
CONFUSION MATRIX

Actual/Predicted	Positive	Negative
Positive	95%	5%
Negative	11%	89%

B. Entropy Detection

Web server user interactions can be modeled as a series of successive requests grouped together to form Web sessions over a given time period. Requests are human-initiated events utilizing HTTP protocol commands such as GET in order to retrieve Web content. A series of related requests generated by a specific user form a single Web session. Web request logs provide no indication of when sessions end or start. In order to establish user sessions within the request logs it was assumed that a 900 second or greater pause between consecutive requests of a single user indicated an end of a previous and start of a new Web session [13]. Session length was defined as the number of Web documents requested during a session. While previous research has examined the use of statistics derived from Web request logs such as source IP address frequency [14] or request inter-arrival times [5] within a Web session, the focus of this detection methodology is to model individual user behavior by analyzing the entropy of session lengths generated by an individual over a fixed duration of time in order to detect FRC attacks on the utility model.

Publicly available Web request logs, commonly used for flash crowd and DDoS detection research [5], [14], were used as normal activity to conduct the entropy-based detection experiments. One data set originated from activities observed by a busy ISP Web server over a two-week period. The second collection is two months of Web logs from a busy NASA Website [15].

The attacks within the experiments were modeled as a botnet that consisted of 500 distinct bots that generated for each bot, on average, a total of 200 malicious requests over a given week. This fixed volume of traffic represented 46-113% more traffic than the original volume for a particular week. Malicious requests were composed of uniformly random session lengths between 1 and 15 Web content requests.

The hypothesis is that randomly generated session lengths, as previously described, deviate sufficiently from a profile of normal user behavior in such a way that detection and attribution of attacking bots is possible using an entropy-based detection scheme.

The proposed detection methodology includes both a learning stage and a detection stage. The learning stage involves computing a standard of entropy of normal session lengths based on a Web request log designated as the training data set.

The detection stage consists of computing entropy of session lengths for each unique user and comparing the entropy result to the standard. If a user's session length entropy is outside the standard, the user is designated as malicious.

Entropy has been used in many detection contexts, including application-layer DDoS detection [16], [17]. If the probability that a discrete random variable X takes on a value x_i, given by $p(x_i) = P(X=x_i)$, then the entropy of session length for session j is H_j composed of the n events is defined as follows:

$$H_j = -\sum_{i=1}^{n} p_i log_2(p_i) \tag{7}$$

The entropy of session lengths for a given data set is a random variable H that exhibits the properties of a normal distribution. Each weekly Web request log data set served as a training data set while the remaining seven data sets represented potential FRC attacks. One advantage of training on each week is that if flash crowds were present they would not be errantly detected as malicious as is the case with some DDoS approaches. Only relative anomalous behavior is flagged as malicious.

To distinguish normal user behaviors from anamolous behaviors, a tolerance interval bounding 90% of the assumed usual traffic (γ=0.90) was calculated with 95% confidence (α=0.05) using a two-sided tolerance interval $\bar{h} \pm k_2 s$ where $\bar{h} = \sum_{i=1} h_i/n$ is the mean of the entropy of session lengths for the n respective clients, s is the sample variance and k_2 is:

$$k_2 = \sqrt{\frac{(N-1)(1+\frac{1}{N})z^2_{(1-p)/2}}{\chi^2_{\gamma,N-1}}} \tag{8}$$

For this two-sided tolerance interval, γ is the critical value of the chi-square distribution with N-1 degrees of freedom. This test considers α percent of the sample population with $z_{(1-p)/2}$ as the critical value for the normal distribution with confidence (1-p)/2.

This metric was applied as the standard of entropy to the test data sets that potentially contained attack traffic mimicking a FRC attack. Experimental results for FPR and FNR are shown in Table VI and Table VII respectively. These tables summarize the findings of the 56 experiments utilizing the two months of NASA Web request logs that were segregated into 8 weekly subsets. The ISP Web logs experiments produced the following results: week 1 as the training set FPR: 4.0%; FNR: 1.2%, week 2 as the training set FPR: 9.1%; FNR: 0.6%.

TABLE VI
FALSE POSITIVE RATES (%)

Train/Test	Wk1	Wk2	Wk3	Wk4	Wk5	Wk6	Wk7	Wk8
Wk1	-	4.4	5.1	8.2	6.1	7.7	7.1	11.6
Wk2	8.0	-	6.5	9.2	8.2	13.9	5.7	14.2
Wk3	4.0	4.4	-	7.1	6.1	7.7	7.1	12.2
Wk4	4.0	3.7	4.4	-	6.1	7.7	5.7	10.3
Wk5	4.0	3.7	4.4	4.1	-	7.7	5.7	10.3
Wk6	1.3	2.5	2.9	3.1	4.1	-	4.3	7.1
Wk7	2.7	3.1	3.6	3.1	4.1	4.6	-	7.7
Wk8	2.0	1.9	2.9	3.1	4.1	1.5	5.7	-

TABLE VII
FALSE NEGATIVE RATES (%)

Train/Test	Wk1	Wk2	Wk3	Wk4	Wk5	Wk6	Wk7	Wk8
Wk1	-	3.4	5.8	4.8	4.4	3.4	3.2	4.0
Wk2	7.0	-	6.4	4.8	6.8	8.8	4.6	6.4
Wk3	5.4	3.2	-	4.6	6.0	3.8	2.6	5.2
Wk4	3.6	3.6	3.8	-	3.0	3.6	3.2	4.2
Wk5	6.8	4.6	6.4	6.8	-	4.2	7.2	5.4
Wk6	9.2	10.2	9.0	6.8	6.0	-	7.0	7.6
Wk7	7.2	5.0	7.4	4.8	6.8	5.6	-	7.0
Wk8	5.4	6.0	7.0	6.4	5.0	6.0	5.4	-

Initial experimental results show that entropy-based detection of session length variation is potentially an effective means to detect and reduce the effectiveness of FRC attacks. Like most detection schemes, the methodology as it is currently devised can be defeated. By increasing the number of attacking bots and decreasing the amount of sessions produced by each bot, the attacker can achieve their objective and moderate the amount of entropy the attack produces thus remaining within the tolerance limit. In a practical context with an attacker lacking the insight on usage patterns, staying within the tolerance limit can be difficult to judge by the attacker.

V. RELATED WORK

This section provides a survey of related work that has bearing and similarities with that of a FRC attack.

A. Economic Denial of Sustainability

The notion of an Economic Denial of Sustainability (EDoS) attack has been previously discussed in non-academic forums. The term was first presented on a blog posting by cloud computing security professional Christopher Hoff [18] and has since been discussed in similar contexts. Hoff describes the EDoS attack as a purposeful manipulation of a utility pricing model that exceeds the economic means of a cloud consumer. This description of the vulnerability of the utility model in the cloud has served as a key motivation for this work. This work on the FRC attack is a refinement that considers a more subtle threat model and subsequent detection.

B. Application-Layer DDoS

There have been a number of key works that have explored application-layer DDoS attacks that have resulted in potential detection and mitigation techniques that may be applicable to FRC attacks. Although the attacker's objectives and request intensities of DDoS attacks are significantly different from that of a FRC attack, this particular body of work is relevant because both attacks employ similar attack methods to mimic the behaviors of legitimate clients.

Within this body of work, some have sought to distinguish flash crowds from DDoS attacks [6], [7], [17]. It was found that the number of overall client requests in a flash crowd was proportional to the number of users [17] and that flash crowds do not exhibit higher per-client request rates [7]; both are behaviors that differ from DDoS attacks. While significant, these findings are not applicable to the detection on a FRC attack. The FRC attack is fundamentally different from the behaviors of flash crowds as these events are composed of

dramatically increased amounts of *normal traffic* over a short period of time.

Due to the nature of DDoS attacks - a large amount of requests during a short period of time - detection and attribution of malicious clients has focused on statistical methods for detecting such behavior. Ranjan et al. [19] implemented a DDoS defense that consists of a suspicion assignment technique that is reliant on an abnormal increase in the inter-arrival request frequency for individual users and an increased session inter-arrival frequency over all clients to be successful. Oikonomou et al. [5] proposed a technique to model normal user behavior by exploring increased metrics for session and request inter-arrival times as well as the average inter-arrival rate per client session. Finally, Jung et al. [7] used similar methods of flagging offending attackers by monitoring per-client request rates. The effectiveness of each of these three solutions is contingent on the individual malicious clients performing requests at a significantly higher rate during a DDoS than that of normal traffic. This is contrary to a FRC attack in which a malicious attacker would only need to make on the order of 200 requests over a given day that would not be detected by these DDoS detection methodologies.

Others have analyzed request semantics of users [5], [17], which appears *prima facie* to be a promising area for future work that could be used in tandem with the presented FRC detection methodologies.

To be fair, the approaches discussed in this section were not designed nor analyzed with respect of a FRC attack by their respective authors. Until now, the FRC attack has not been a research subject. Future examination and experimentation will answer the question of whether or not these detection methodologies are limited to application-layer DDoS attacks or whether they possess qualities that can be adapted for the purpose of detecting significantly reduced but still malicious consumption of Web content.

VI. Future Work & Conclusion

Future research on this topic will focus on shifting the J2 point on the MRU continuum ideally to where J2 = J1 - ϵ or in other words the FRC attack has been relegated to below nuisance activity. While research in the area of application-layer DDoS attacks has been focused on mitigating attacks when the QoS of the target begins to degrade, methodologies and approaches from this body of work serve as a promising catalog of detection techniques on which to base detection methodologies that are appropriate for more subtle attacks like the FRC.

Utility models as they are structured today for cloud computing are vulnerable to remote exploits. By allowing any user with access to consume resources that are in turn metered and billed to the cloud consumer, exposes the cloud consumer to a risk that is only mitigated by time, detection and accountability. Until now there have been no previously known mitigation strategies. Awareness and understanding are a key means of defense, and this paper strived to achieve those goals. This paper provided a thorough description of the FRC attack and described two detection methodologies that may prove to be solutions to detect such attacks. Unless utility models are restructured to obviate the FRC attack, research in mitigating the FRC attack is necessary in order to ensure long term sustainability of cloud consumers and remove one more impediment that has dissuaded organizations from adopting utility model based computing services like cloud computing.

Acknowledgment

The authors would like to thank the anonymous reviewers and Dr. Doug Jacobson.

References

[1] J. Mirkovic and P. Reiher, "A taxonomy of ddos attack and ddos defense mechanisms," *SIGCOMM Comput. Commun. Rev.*, vol. 34, pp. 39–53, April 2004.

[2] Amazon Web Services. (2010, Nov.) Amazon EC2 pricing. [Online]. Available: http://aws.amazon.com/ec2/pricing/

[3] 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 conference on Symposium on Networked Systems Design & Implementation - Volume 2*, ser. NSDI'05, 2005, pp. 287–300.

[4] Z. Li, Q. Liao, and A. Striegel, "Botnet economics: Uncertainty matters," in *Managing Information Risk and the Economics of Security*. Springer US, 2009, pp. 245–267.

[5] G. Oikonomou and J. Mirkovic, "Modeling human behavior for defense against flash-crowd attacks," in *IEEE International Conference on Communications*, 2009, pp. 1–6.

[6] S. Wen, W. Jia, W. Zhou, W. Zhou, and C. Xu, "Cald: Surviving various application-layer ddos attacks that mimic flash crowd," in *2010 4th International Conference on Network and System Security (NSS)*, 2010, pp. 247–254.

[7] J. Jung, B. Krishnamurthy, and M. Rabinovich, "Flash crowds and denial of service attacks: characterization and implications for cdns and web sites," in *Proceedings of the 11th international conference on World Wide Web*. New York, NY, USA: ACM, 2002, pp. 293–304.

[8] S. Ramachandran. (2010, May) Web metrics: Size and number of resources. [Online]. Available: http://code.google.com/speed/articles/web-metrics.html

[9] G. K. Zipf, *Human Behavior and the Principle of Least Effort*. Addison-Wesley, 1949.

[10] K. Narisawa, D. Ikeda, Y. Yamada, and M. Takeda, "Detecting blog spams using the vocabulary size of all substrings in their copies," in *In Proceedings of Workshop on Weblogging Ecosystem*, 2006.

[11] S.-M. Huang, D. C. Yen, L.-W. Yang, and J.-S. Hua, "An investigation of zipf's law for fraud detection," *Decis. Support Syst.*, vol. 46, pp. 70–83, December 2008.

[12] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and zipf-like distributions: evidence and implications," in *Proceedings of IEEE INFOCOM '99*, vol. 1, Mar. 1999, pp. 126–134 Vol.1.

[13] L. Kroc, S. Eidenbenz, and J. Smith, "Sessionsim: Activity-based session generation for network simulation," in *Winter Simulation Conference (WSC), Proceedings of the 2009*, 2009, pp. 3169–3180.

[14] Q. Le, M. Zhanikeev, and Y. Tanaka, "Methods of distinguishing flash crowds from spoofed DoS attacks," in *Next Generation Internet Networks, 3rd EuroNGI Conference on*, May 2007, pp. 167 –173.

[15] Traces available in the internet traffic archive. [Online]. Available: http://ita.ee.lbl.gov/html/traces.html

[16] J. Wang, X. Yang, and K. Long, "A new relative entropy based app-DDoS detection method," in *IEEE Symposium on Computers and Communications (ISCC)*, 2010, pp. 966–968.

[17] Y. Xie and S.-Z. Yu, "Monitoring the application-layer ddos attacks for popular websites," *Networking, IEEE/ACM Transactions on*, vol. 17, no. 1, pp. 15–25, 2009.

[18] C. Hoff. (2008, Nov.) Cloud computing security: From DDoS (distributed denial of service) to EDoS (economic denial of sustainability). [Online]. Available: http://www.rationalsurvivability.com/blog/?p=66

[19] S. Ranjan, R. Swaminathan, M. Uysal, A. Nucci, and E. Knightly, "DDoS-shield: DDoS-resilient scheduling to counter application layer attacks," *IEEE/ACM Trans. Netw.*, vol. 17, pp. 26–39, February 2009.