Experimental Evaluation of an Adaptive Flash Crowd Protection System

Xuan Chen and John Heidemann *

ISI-TR-2003-573
July 2003

Abstract

Network early warning system (NEWS) is an adaptive admission control scheme that protects server and networks from overloading during flash crowds, and maintains high performance for accepted requests. Unlike other admission control systems, NEWS regulates *requests* by observing *response* performance, automatically adapting to changing traffic mixes. We have previously studied NEWS through simulation; this paper presents an implementation of NEWS on a Linux-based router and evaluates that implementation in testbed experiments with HTTP server log recorded during a flash crowd. This paper has three contributions. First, we use the implementation to evaluate scenarios not considered in simulation. In addition to validating our previous simulation results in network-limited scenario quantitatively, we further consider server memory-limited scenario, confirming that NEWS is effective in both cases. Second, we evaluate the run-time cost of NEWS traffic monitoring in practice, and find that it consumes little CPU time and relatively small memory. Finally, we extend core NEWS algorithms to include hot-spot identification function to protect bystander traffic from flash crowds efficiently.

1 Introduction

We design *Network Early Warning System* (NEWS) to adaptively protect server and networks from persistent overloading during flash crowds [1, 2]. Flash crowds usually happen when many end-users simultaneously send requests to one web site (*target server*) because of sudden events such as earthquakes, breaking news stories, or links from popular web sites (slash-dot effect). As shown in Figure 1, target server may reject some excessive requests, and process other accepted ones slowly due to either its resource limitation (CPU, memory or disk), or congestion in response network path [3]. As a result, most end-users perceive unacceptably poor performance. In the mean while, flash crowds unintentionally deny services for other end-users who either share common links with flash crowd traffic or retrieve unrelated information from target server.

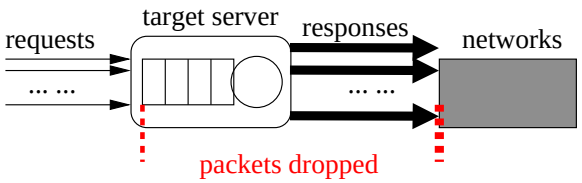


Figure 1: Several factors affect end-user web performance during flash crowds.

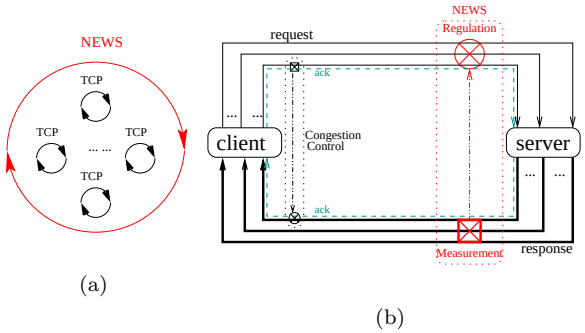


Figure 2: NEWS enforces high-level control among individual connections, and between requests and responses.

Flash crowds are caused by too many concurrent connections, even though each of them is well behaved under TCP congestion control. Based on this observation, we argue that a high-level control is essential to mitigate flash crowds. By deploying NEWS, we impose such a super-visioning control from two aspects: it enforces cooperation among individual TCP connections (Figure 2(a)); it further coordinates requests and responses (Figure 2(b)) so

*Xuan Chen and John Heidemann are with University of Southern California, Information Sciences Institute. This material is based upon work supported by DARPA via the Space and Naval Warfare Systems Center San Diego under Contract No. N66001-00-C-8066 ("SAMAN"), and by the CONSER project supported by NSF.

that neither target server nor network links are overloaded.

We design NEWS as a router-based adaptive admission control scheme. It admits incoming requests based on measurement of response performance. This approach is different from previous admission control algorithms considering explicit service requirement or measuring performance of incoming traffic directly [4, 5, 6, 7]. Moreover, NEWS only measures aggregate performance for fast connections (details in Section 3.1), rather than for all existing flows like measurement-based admission control [6, 7, 8].

Another novel aspect of NEWS is that it detects flash crowds through performance change in response traffic. This is different from traditional detection schemes that monitor increase in request rate directly [9, 10]. As shown in Figure 1, many factors affect end-user perceived performance, including request rate, server's capacity, and network bandwidth. As a result, it is difficult to determine a single optimal threshold in request rate to detect flash crowds. Even if an optimal threshold exists, it changes over times due to variation in response sizes [11]. In another word, request rate is not necessarily correlated with response performance. Therefore, we believe that we should detect flash crowd by monitoring performance changes in response traffic.

We have developed basic flash crowd detection and mitigation algorithms for NEWS in *ns* [12], and evaluate NEWS performance with simulations [13]. In this paper, we report our experience of implementing NEWS on a Linux-based router, and summarize results from testbed experiments. This work has three major contributions.

First, with testbed experiments, we validate NEWS performance in a network-limited scenario quantitatively in a more realistic experimental model than simulations. We also configure a server-limited scenario, where request processing exhausts server's memory. We investigate system performance of target server under flash crowds in details.

We show that NEWS is an adaptive system; effectively prevents server and network overloading in both scenarios. More specifically, NEWS detects flash crowds around one detection interval (88 seconds with 64 seconds' detection interval). It automatically regulates incoming requests to a proper rate. As a result, NEWS protects target server and networks from overloading by discarding about 49% excessive requests. NEWS also maintains high performance for admitted requests: increasing aggregate response rate for high-bandwidth connections by two times. This performance is comparable to the best possible static rate limiter in the same scenario.

Second, by implementing NEWS on a Linux-based router, we examine NEWS overhead on router. We measure CPU and memory usage of NEWS under different detection intervals. Our statistics show that NEWS only consumes less than 5% of CPU time and 3–10M byte memory. Overall, NEWS is a relative light-weighted scheme and applicable to real networks.

Third, we extend NEWS to automatically detect target server by extracting common characteristics among incoming requests. In the case of multiple servers connecting through NEWS router, this *hot-spot identification algorithm* helps NEWS to regulate incoming requests intelligently, that is, automatically identifying one hot server even if many other servers are operational behind the NEWS router. Our results show that this function can protect traffic to nearby, *bystander* servers efficiently. Compared to original NEWS algorithms, it reduces average end-to-end latency for bystander web traffic by about 17 times (Section 5.6).

2 Related Work

In this section, we briefly describe mechanisms to accommodate flash crowds through resource provisioning. We also review other commonly used schemes to protect servers and networks from overloading, such as admission control, congestion control, and server overload control.

Infrastructure vendors such as Akamai [14] deploy web caches and content delivery networks (CDN) to accommodate excessive web requests during flash crowds. However, recent studies [1, 15] show that current web caching schemes are not so efficient to deal with flash crowds as claimed. One important reason is that most web pages are not cached before flash crowds (breaking news, for example). As a result, requests still reach target server eventually. Jung et. al. proposed adaptive web caching [1] for improvement.

It is necessary to provide enough resources to prevent flash crowds from happening, and accommodate excessive traffic afterward. However, due to the nature of flash crowds, there are circumstances that it is either difficult or impossible to provide or even estimate the "enough" resources [2, 16]. Therefore, we believe that we need to deploy control scheme like NEWS to mitigate flash crowds in these cases so that servers and networks survive from overloading, and some end-users still perceive reasonable high performance.

Due to large number of concurrent connections during flash crowds, per-connection (such as TCP)

and per-host (such as congestion manager (CM) [17]) based congestion control algorithms are not sufficient to protect network from overloading. On the other hand, we can apply algorithms like aggregate-based congestion control (ACC) [18] to relief congestion caused by response traffic. However, since ACC does not reduce the volume of incoming requests, it can not mitigate flash crowds either. Therefore, we emphasize that a high-level control between requests and responses is essential to protect servers and networks from flash crowds.

Admission control [4, 5, 6, 7] is important to support quality-of-service. Based on service requirement and current available resources (network bandwidth or circuits), admission control makes decision to accept incoming connections or not. Although appropriate in integrated service networks [19] and public telephone networks [20], it is difficult for some applications to accurately specify their service requirements (for example, web). As a result, traditional admission control may under-utilize server or network resources. NEWS avoids this problem by determining application service requirement dynamically through measurement (that is, web performance). NEWS is also different from measurement-based admission control (MBAC) [6, 7, 8] in that it only measures aggregate response rate of high-bandwidth connections; while MBAC monitors performance of all existing flows.

Another similar work in this region is the network weather service (NWS) [21]. It monitors and forecasts system performance such as link utilization and server load. Both NWS and NEWS share some common ideas in change detection algorithms. Unlike NWS, NEWS does not rely on centralized data processing.

Recent study [6, 16, 22, 23] proposed to apply overload control and service differentiation on web servers to improve web performance under heavy load. These schemes are complimentary to NEWS. However, it is questionable whether busy servers are still capable to monitor current performance and classify requests during flash crowds. On the contrary, by deploying NEWS on access router, we provide early protection by dropping excessive requests before they ever reach target server. NEWS is also more flexible than server-based overload control, effectively protecting bystander traffic to other servers that connect through NEWS router from flash crowds.

3 System Design

The system design of NEWS is primarily based on flash crowd detection and mitigation algorithms we developed in simulation [13]. Rather than discuss all details, we highlight key ideas behind these algorithms and present some improvement in this work. As an extension to original algorithms, we develop hot-spot identification algorithm to automatically discover target server during flash crowds. NEWS uses this function to protect bystander traffic to other servers.

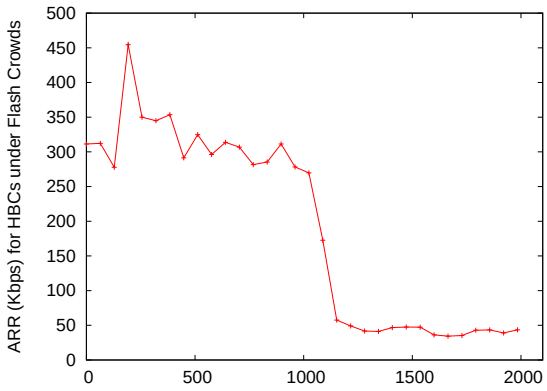


Figure 3: Aggregate response rate of high-bandwidth connections drops dramatically under flash crowds.

We have two assumptions in NEWS design. First, in order to better capture aggregate response performance, NEWS should be deployed on the access router (or firewall) of target server and its intranetworks. We also assume that requests and responses traverse that same access router. In case of a multi-homed domain, we might need to distribute NEWS to all access routers, and coordinate flash crowd detection and mitigation across different points.

3.1 Detecting Flash Crowds

Since connections during flash crowds are usually short, we can not detect flash crowds by monitoring performance change of particular connections. On the other hand, monitoring average response rate of all connections is not helpful either because connections with low access speed show little performance change under congestion.

Instead, NEWS detects flash crowds from decrease in aggregate response rate (ARR) of high-bandwidth connections (HBCs), that is, *fast* connections. We believe that these connections are more sensitive to overloading. For example, in our experiments (detailed methodology in Section 5.1), we observe that 5% fastest connections suffer more than 80% decrease in their aggregate response rate after a flash crowd happens at 1000 second (shown in Figure 3).

Given current measurement on aggregate response

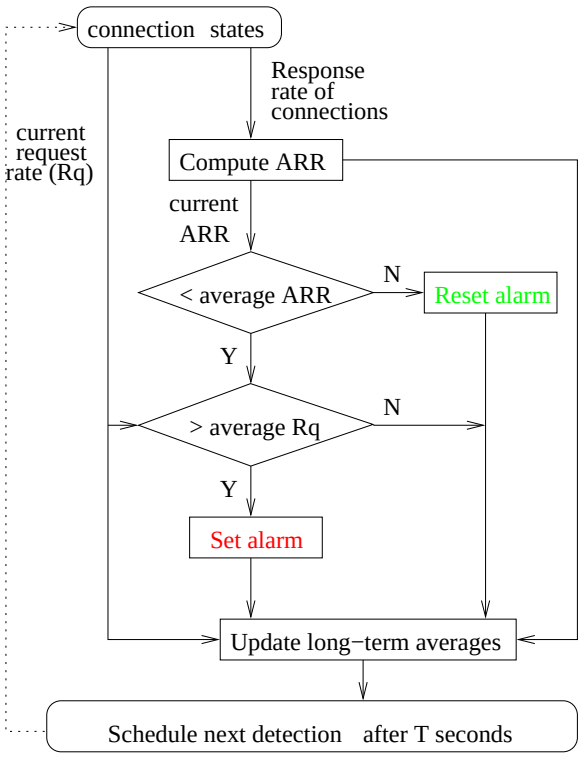


Figure 4: Flow chart of flash crowd detector.

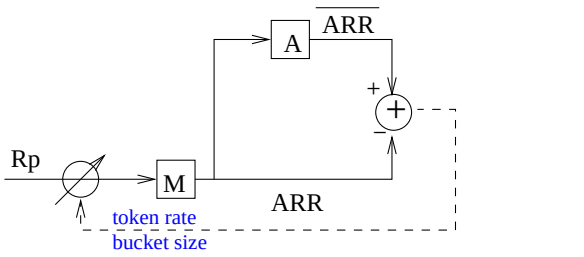


Figure 5: Control logic for request regulation.

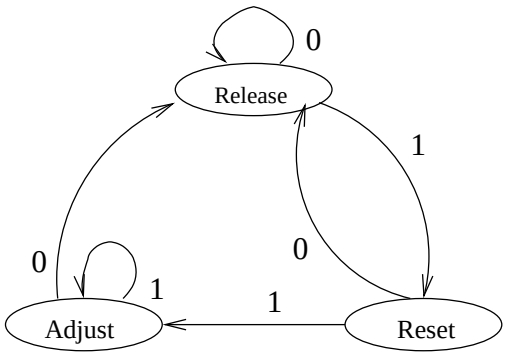


Figure 6: State transition of NEWS control logic based on alarm signal.

rate of high-bandwidth connections, NEWS computes its long-term average with a Exponentially Weighted Moving Average (EWMA) filter. More specifically, we assign a high gain on current measurement under common situations without alarm signal for fast response. After flash crowd alarm is triggered, it switches to a low gain to keep the value of long-term average stable and avoid oscillations.

NEWS catches cases when current aggregate response rate falls below its long-term average. To ensure that this decrease is due to flash crowds rather than mis-measurement of slow connections, NEWS also checks current request rate (Rq). If there's an increase in request rate compared to its long-term average, NEWS triggers flash crowd alarm. NEWS resets alarm signal when it observes an increase in aggregate response rate, which indicates performance recovery. We depict the procedure of flash crowd detection in Figure 4.

3.2 Discovering Target Server

As an improvement to our previous work, we extend NEWS to find the *hot-spot*, that is, target of flash crowd traffic. In many cases, servers operate in server farms, and connect through one access router. If we deploy NEWS on that access router, it must distinguish between flash crowd traffic going to a hot-spot and low-rate, bystander traffic going to other servers.

NEWS uses a hot-spot list to record the frequency of each server being visited. NEWS randomly samples incoming connections, and hashes their destination addresses into hot-spot list. Each entry in hot-spot list records the number of corresponding address being sampled (*hit-number*) within certain time interval. NEWS picks the address with highest hit-number as hot-spot. Periodically, it timeouts inactive entries in the list.

When NEWS detects flash crowds, it matches incoming requests with hot-spot list, and drops those with their destination addresses identified as hot-spot. In the mean while, NEWS admits bystander traffic with a probability proportional to its history of admission. That is, the more bystander connections were accepted in the past, the more likely new connections are dropped. Hot-spot identification algorithm protects bystander traffic. We evaluate its effectiveness in Section 5.6.

3.3 Regulating Requests Adaptively

As shown in Figure 5, NEWS augments a token bucket based rate limiter with control logic to regulate incoming requests adaptively. More specifically,

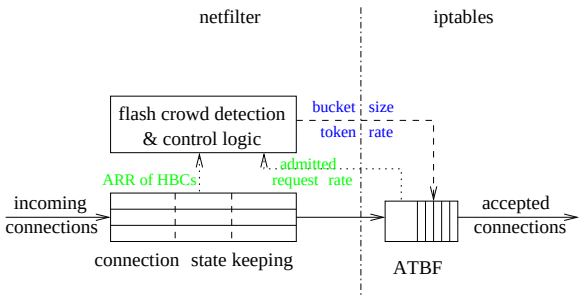


Figure 7: NEWS implementation

NEWS adjusts token rate and bucket size of rate limiter based on current measurement and changes of alarm signal. We illustrate state transition of NEWS control logic in Figure 6.

When alarm signal is set (changes from 0 to 1), NEWS *resets* token rate to long-term average of request rate. Intuitively, requests arriving with higher rate are likely to overload target server or networks, and therefore reduce response performance. NEWS also resets bucket size to a small value to penalize bursty arrivals during flash crowds.

When alarm signal changes back to 0 (from 0 to 0 or from 1 to 0), NEWS attempts to *release* its control on token rate and bucket size to fully-utilize server's capacity and network bandwidth. On the other hand, if alarm signal remains set (from 1 to 1), NEWS control logic enters *adjust* state. It reduces token rate and bucket size further, that is, applying stricter control on incoming requests. With this control logic, NEWS is able to regulate incoming requests to a proper rate, and therefore prevent overloading on server or in networks.

4 Implementation

We implement NEWS under the framework of *iptables/netfilter* [24]. *Iptables/netfilter* is the firewall subsystem for Linux kernel 2.4 and above. It provides various facilities such as stateful or stateless packet filtering, network address translation (NAT), and packet mangling. Netfilter and most functions of iptables (such as packet matching) are implemented in kernel. They process and manipulate packets according to different rules that users configure through iptables user interface.

As shown in Figure 7, we implement NEWS as three modules: connection state-keeping module, flash-crowd detection and control logic, and adaptive token bucket filter (ATBF). By dividing NEWS functions into different modules, we separate fast per-packet processing such as connection state keeping and packet matching and filtering from slow connection-based detection and control functions. We present implementation details below.

4.1 Connection State Keeping

This module utilizes connection tracking function in netfilter. With this function, router keeps one record for each connection. A connection has two directions: *original* and *reply*[1]. For example, request in a web connection is in *original* direction, and response is in *reply* direction.

We add a new state for each connection: response rate. That is, the data transmission rate (measured in bits per second) on *reply* direction. In most cases such as web and FTP traffic, this state records the actual data rate end-users perceive.

We record the length of every incoming packet, and update response rate of corresponding connection using time-sliding window (TSW) algorithm [25, 26] to compute a smoothed rate from individual packet transmissions. We intentionally avoid first few control packets such as SYN-ACK to keep measured response rate stable.

In our currently implementation, NEWS keeps track of all connections. This may consume considerable amount of memory on router. We measure this consumption on NEWS router and investigate options to reduce memory usage in Section5.5.

4.2 Flash Crowd Detection and Control Logic

Applying the algorithm described in Section 3.1, NEWS detects flash crowd periodically with an interval of T seconds. Based on alarm signal, NEWS control logic regulates incoming requests adaptively (Section 3.3) .

The detection interval T is a tunable parameter. Since NEWS keeps traffic measurement for last T seconds, NEWS with a smaller detection interval is more sensitive to traffic change, and detects changes promptly. However, the result is likely to oscillate, that is, have high false alarm rate. On the other hand, NEWS gives more stable output with larger detection interval at the expense of longer detection delay.

A smaller detection interval consumes more CPU time. But, larger T needs more memory to keep connection states. We investigate the effect of different detection intervals in Sections 5.4 and 5.5. Based on our experience, we choose T as 64 seconds.

[1] *Original* and *reply* are specific terms used in netfilter implementation.

4.3 Adaptive Token Bucket Filter

We use an adaptive token bucket filter (ATBF) to regulate incoming requests. ATBF has two parameters: *bucket size* provides accommodation to bursty traffic, and *token rate* limits long-term arrival rate. In long run, connections admitted by ATBF converge to token rate.

We implement ATBF as an extension to iptables matching function. That is, any new connections that matches with ATBF are dropped. More specifically, each new connection adds some token into the bucket. It passes through ATBF when there are enough tokens. Otherwise, ATBF discards connections toward target server by matching with hot-spot list (Section 3.2). ATBF counts the number of accepted connections and reports it to flash crowd detector as current measurement of request rate.

One observation in experiments is that connections have very small inter-arrival times, for example, a few milli-seconds. To accurately keep track of token number, we have to measure time at fine granularity: micro-second. Since we can not afford the computational overhead of floating point number division, we calculate the number of tokens cumulated between two adjacent connections by multiplying inter-arrival time (in micro-second) with token rate (in number per second) directly. This increases token number by 1,000,000 times. Accordingly, we change the number of tokens required to admit one connection from 1 to 1,000,000.

5 System Evaluation

In our previous work, we have evaluated NEWS performance in simulations and shown that NEWS protects target server and networks from overloading and maintains high performance for admitted requests [13]. In this work, we evaluate NEWS implementation with testbed experiments. We validate our previous simulation studies in a network-limited scenario quantitatively. More importantly, testbed experiments allow us to investigate server performance and router overhead (such as CPU and memory usage), which are not modeled in network simulations.

We further evaluate NEWS performance in a server-limited scenario, confirming that NEWS is an adaptive system, and capable to prevent overloading on server or in networks efficiently. We also show that NEWS is a relative light-weighted scheme by examining its run-time overhead on router. Finally, we evaluate the effectiveness of hot-spot identification algorithm through experiments with bystander traffic.

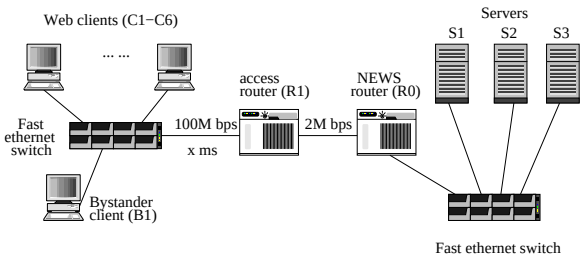


Figure 8: Network topology in experiment.

5.1 Methodology

Figure 8 shows our testbed environment. All machines have 100M bps fast Ethernet network interfaces. The client pool is composed of 7 machines with various configurations ranging from 677M Hz Pentium III CPU with 128M byte RAM to 2GHz Pentium 4 CPU with 512M byte RAM. We use 6 machines (C1–C6) for flash crowd traffic generation, and one (B1) for bystander traffic experiment.

Our server pool has 3 machines. Fast target server (S1) and bystander server (S3) have 2GHz Pentium 4 processor and 512M byte RAM. Slow target server (S2) has 180MHz Pentium Pro CPU and 128M byte RAM. We connect client and server pools together through two routers R0 and R1. We use Linux Redhat 9 (kernel 2.4.20-8) on all machines, including two routers.

The NEWS router (R0) is a PC with 1.5GHz AMD Athlon 4 processor and 1G byte RAM. We deploy NEWS on R0 by enabling iptables service with NEWS extension. In most experiments, we configure NEWS detection interval as 64 seconds. We investigate the effect of different detection intervals in Sections 5.4 and 5.5 .

We use NIST Net [27] to introduce network dynamics in our experiments. NIST Net allows Linux-based router to emulate a wide variety of network conditions, such as link bandwidth, propagation delay, and packet loss. We configure link bandwidth between R0 and R1 as 2Mbps. We also set different propagation delays between R1 and client machines according to network measurement at USC/ISI [28]. While we can not claim that our testbed simulates "the Internet" [29], this experimental study does help us to better understand dynamics of NEWS in a relatively real environment than simulations.

Our web servers use TUX [30] to expedite request progressing. Running partially from within Linux kernel, TUX has demonstrated outstanding performance for serving static contents. We configure web servers with large SYN buffer size and HTTP backlog to ensure that they are fed with enough workload.

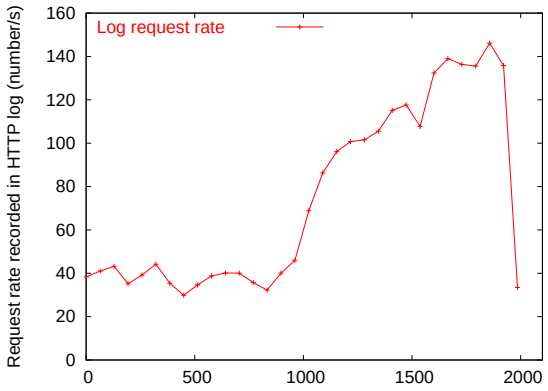


Figure 9: Request rate to World Cup'98 web site.

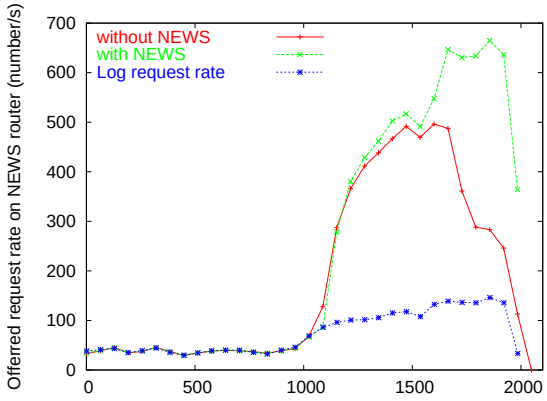


Figure 11: Increased offered request rate due to retransmissions.

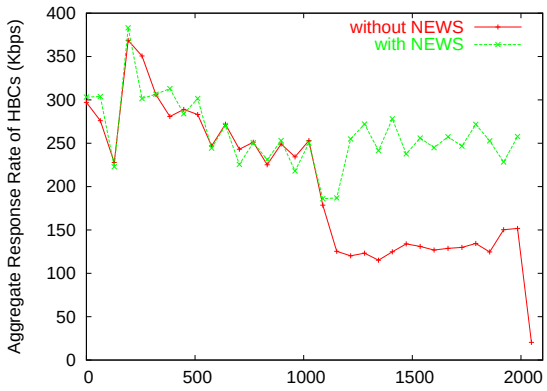


Figure 12: NEWS maintains high aggregate response rate for HBCs during flash crowds.

We implement a traffic generator to playback web server log in experiments. It sends web requests to server according to information in log entries, such as time, server address, and request size. To simulate many concurrent connections, our traffic generation tool creates threads to accomplish request-response exchanges independently. Due to memory constraint on one single client machine, we distribute traffic generation across six client machines (C1–C6).

In our experiments, we use HTTP logs recorded on one server for 1998 World Cup located at Santa Clara, California [31]. Before each game, web server recorded a large (5–10 times) increase in request rate [15], which led to a flash crowd. Our experiments replay 2000 seconds' of log from one of the semi-finals between Brazil and Holland on July 7. As shown in Figure 9, request rate increases by about 5 times in just several minutes after the game started at 1000 second.

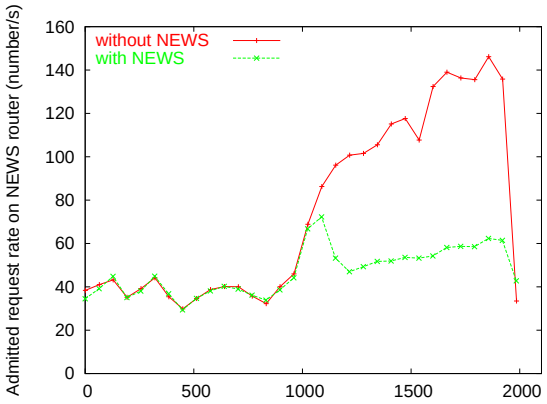
We monitor web request rate both at NEWS router (R0) and target web server. To quantify end-user perceived performance, we record aggregate response rate of high-bandwidth connections on router R0. We also keep track of system and network statistics on each machine, such as CPU and memory usage, network utilization, and packet drop rate. We are particularly interested in changes in these statistics when flash crowds happen.

5.2 NEWS Relieves Network Congestion

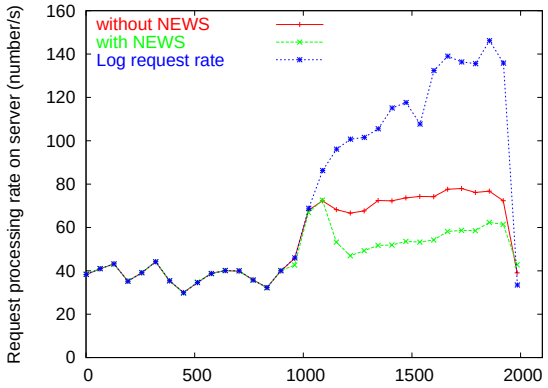
We have two different configurations in our experiments, namely network- and server-limited scenarios. We have studied NEWS performance in a network-limited scenario in simulations. In section, we validate our simulation results quantitatively, confirming that NEWS effectively protects server and networks from overloading and maintains high performance for accepted requests. Further, we investigate system performance of target server in details through experiments.

In this experiment, we send web requests to fast target server S1. These requests do not overload S1; in fact, they only consume about 2–3% CPU time and about 60% memory. However, the response traffic generated by server congest network link between router R0 and R1. Our measurement shows 100% link utilization and about 60% packet drop rate.

As a result, target web server S1 is kept busy with transmitting and retransmitting response traffic, and is not able to catch up with all incoming requests. As shown in Figure 10(b), S1 can only process about 80 requests per second, 31.5% less than average incoming request rate during flash crowds (116.8 requests per second). Congestion in response network link and slow server processing (hence, large server backlog) greatly reduce end-user perceived performance: aggregate response rate of high-bandwidth connections



(a) Admitted request rate on NEWS router.



(b) Request processing rate on target server.

Figure 10: NEWS regulates incoming requests to a proper rate.

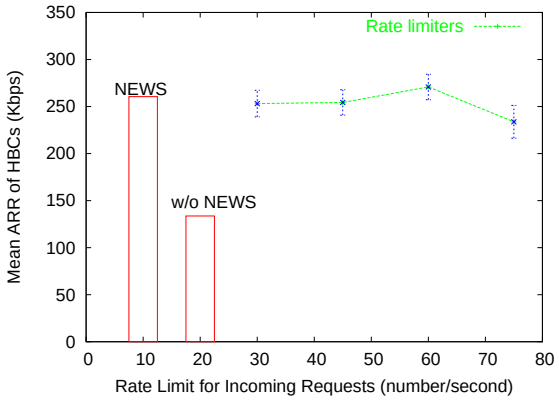


Figure 13: Comparison of NEWS and static rate limiter.

drops in half as shown in Figure 12. Also, about 22% connections timeout due to packet loss. Failed connections resend SYN packets after timeout, and thus increase average offered request rate on NEWS router from 109.9 to 317.8 requests per second (Figure 11).

We deploy NEWS on R0. We find that, with 64 second detection interval, NEWS detects flash crowd at about 1088 second, that is, 88 seconds after flash crowd starts. NEWS protects target server and networks from overloading by regulating admitted request rate down to 55.9 requests per second (Figure 10(a)). As less response traffic generated, packet loss rate on link between R0 and R1 falls to less than 3%.

NEWS also maintains high performance for admitted requests during flash crowds: increasing aggregate response rate of high-bandwidth connections from 133Kbps to 260.52Kbps (Figure 12). As shown in Figure 13, this performance is comparable to static token bucket based request rate limiter with manually configured token rate Therefore, our experiment study confirms that NEWS protects server and networks from flash crowds effectively, and maintains high performance for admitted requests.

As a cost to achieve this performance, NEWS discards or delays about 49.1% excessive requests. As a result, we observe offered request rate to router increases by 44.3% from 317.8 to 458.6 requests per second (Figure 11). Although NEWS by itself can not serve all requests on time, it does ensure that admitted requests still perceive high performance even under flash crowds, rather than leaves all end-users with unacceptable low web performance.

5.3 NEWS Prevents Server Overloading

We further evaluate NEWS performance in a server-limited scenario. This evaluation was not possible in simulation because network-level simulators do not include detailed models of server CPU, memory, and disk performance.

In this experiment, we send web requests to slow target server S2. We find that S2 uses up about 98% of its memory and starts swapping. At the same time, link utilization between R0 and R1 is only 60%. We do not observer any packet loss.

Swapping slows down target web server, and builds up server backlog. Eventually, about 20% connections timeout due to long waiting time. As a result, end-users suffer from low web response rate even there is no packet dropped by networks in this sce-

nario.

On the other hand, we find that CPU usage on S2 is only about 15% on S2. This is not unexpected since web requests in our experiments are only for static contents. There is not much computation involved. It is a potential direction to investigate scenarios where CPU is overloaded by dynamic workload in our future work.

Since NEWS detects flash crowds by monitoring response performance, it adapts to server-limited scenario automatically, that is, detects flash crowds withing about 80 seconds. By regulating incoming requests to 54.4 per second, NEWS reduces server memory consumption to about 88%.

As server stops swapping, it processes requests more promptly. Therefore, admitted requests receive much higher performance: aggregate response rate of high-bandwidth connections jumps from 157.69 Kbps to 260.22 Kbps. Based on these results, we conclude that NEWS is an adaptive scheme, and can detect and prevent both network- and server-overloading during flash crowds.

5.4 Different Detection Intervals Affects NEWS Performance

In this section, we examine NEWS performance with different detection intervals. Figure 14 shows aggregate response rate for high-bandwidth connections and admitted request rate under different detection intervals. NEWS shows best performance with detection interval at 64 or 128 seconds.

We also find that NEWS performance falls with either large or very small detection intervals. For example, with detection interval as 256 seconds, NEWS can't adjust rate limit promptly as traffic changes. On the other hand, with very small intervals like 16 seconds, NEWS is so sensitive to traffic change that it triggers alarm before flash crowd happens (at 760 second). Then, it sets rate limit too low that most connections are discarded. These results confirm our findings in simulations.

With detection interval larger than 16 seconds, NEWS always detects flash crowds slightly after one detection interval. Balancing detection delay and performance, we suggest to configure NEWS with detection interval as 64 seconds.

In our experiments, NEWS does not trigger false alarms with detection intervals larger than 16 seconds. We believe this is because that requests generated from one client machine are highly correlated. With our current testbed configurations, we are not able to emulate various access bandwidths and link propagation delays for connections from one single machine. Therefore, we can not investigate scenarios with large network dynamics due to resource restraints. We will investigate other possibilities to emulate more realistic network conditions in our future work.

Detection intervals	before flash crowds	after flash crowds
16 s	2–3.5%	5–6%
32 s	1.5–2%	3.5–5%
64 s	1–2%	2–4%
128 s and larger	about 1%	about 2%

Table 1: CPU time consumed by NEWS with different detection intervals.

5.5 NEWS Consumes Small Router Resources

By implementing and evaluating NEWS in a testbed environment, we are able to investigate system overhead of NEWS, such as CPU time and memory consumption on router.

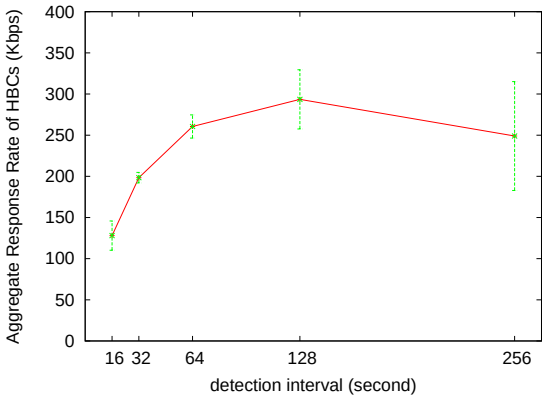
Since we build NEWS on top of connection tracking function in netfilter, memory consumption remains similar before request rate increases: about 3M bytes under different detection intervals. After flash crowds, memory consumption increases, and varies from 10M to 12M bytes depending on the rates that NEWS finds to regulate requests.

It is possible to reduce memory consumption for connection tracking by adjusting connection timeout parameters. For example, we are able to reduce memory consumption after flash crowds to about 6M bytes by timing out connections very 64 seconds. We can further reduce NEWS memory consumption by random sampling incoming connections [32].

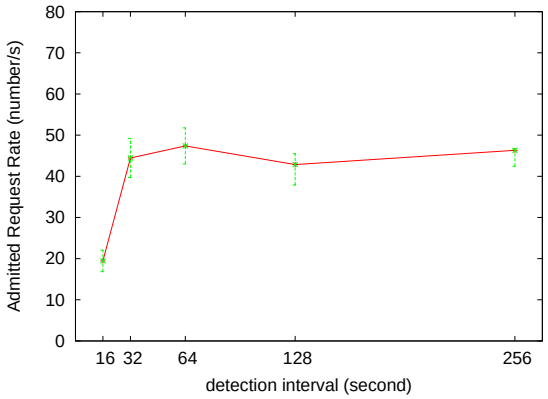
Table 1 shows CPU usage of NEWS under different detection intervals. We find that smaller detection intervals consumes relatively larger CPU time because more frequent flash crowd detections cause more computational overhead. Overall, NEWS consumes less than 5% CPU time with a reasonable detection interval (for example, 64 seconds).

We also measure CPU usage when running NEWS on a slow machine (266MHz Pentium II CPU with 128MB RAM). NEWS consumes about 14% CPU time in average after flash crowds happen. Therefore, we conclude that NEWS is a relatively light-weighted control scheme, and is applicable to real networks to detect and mitigate flash crowds.

In real networks, routers usually have less memory compared to PCs used in our experiments. To have a reference on the overhead of flow stating keeping on commercial routers, we substitute R0 with a Cisco



(a) Aggregate response rate of HBCs.



(b) Admitted request rate.

Figure 14: Different detection intervals affect NEWS performance.

3620 router (100M Hz R4700 CPU, 12MB processor memory and 8MB IO memory). We use NetFlow [33] to keep states for individual unidirectional IP flow in a 70 byte record. With default flow cache size of 256K bytes, our Cisco router is able to track about active 4,000 flows (or 2,000 connections). NetFlow timeouts inactive flows after 15 seconds.

By repeating above experiments, we find that router CPU time is about 2–3% before flash crowd, and 7–8% after. The maximum CPU time reaches 10%. We do not observe any memory allocation failures for incoming flows. In another word, the default 256K byte flow cache is enough to keep states for all flows during the flash crowd in our experiments. Therefore, per-flow state keeping only imposes small overhead to real router in our experiment scenarios.

5.6 NEWS Protects Bystander Traffic from Flash Crowds

Up until now, our experiments only consider flash crowd traffic. In real networks, there always exists other traffic. We call this traffic that shares network links with flash crowds as bystander traffic.

In our previous simulation study, we have shown that NEWS protects bystander FTP bulk traffic from flash crowds. We partially validate this result with experiments. More specifically, we configure server S3 to constantly send 8K byte data chunks to client machine B1. Since NEWS measures aggregate request and response rate, it shows consistent performance in detecting and mitigating flash crowds.

We measure goodput perceived by B1, and find it drops dramatically from 328.6 Kbps to 79.9Kbps as traffic builds up. With NEWS deployed, B1 perceives a consistent goodput of 317.5Kbps Therefore, NEWS can protect bystander bulk data transfer from flash crowds.

Techniques	**Web Latency** (seconds)	
	mean	median
without NEWS	24.44	9.53
with NEWS	7.03	3.15
NEWS + HSI	0.42	0.32

Table 2: Performance of bystander web traffic.

We further evaluate hot-spot identification (HSI) algorithm through experiment with bystander web traffic. In this case, B1 sends web requests to server S3 based on HTTP log under light load. Different from bulk data transfer, we are interested in end-to-end web latency in this case.

We summarize experiment results in Table 2. By discarding about 22% connections, NEWS (with original algorithms) reduces mean web latency for admitted bystander requests by more than 3 times. Further, with hot-spot identification technique, NEWS does not drop any bystander connections. As a result, NEWS further reduces mean web latency for bystander web traffic by about 17 times. Therefore, we conclude that NEWS also protects bystander web traffic; and hot-spot identification algorithm is an effective technique to classify and protect bystander traffic from flash crowds.

6 Future Work

We have evaluated NEWS performance in network- and server-limited scenarios. As a potential direction for our future work, we plan to evaluate NEWS performance in a scenario where server CPU is overloaded with dynamic workload. For example, web requests are for database queries.

With our testbed resources, we also want to compare NEWS with other QoS and overload control schemes in experiments.

7 Conclusion

In this work, we implemented NEWS on a Linux-based router according to flash crowd detection and mitigation algorithms developed in simulations. Besides validate our simulation study quantitatively, we present three major contributions of this work.

First, we investigated system performance of target server during flash crowds in details through experiments. We showed that NEWS is an adaptive system, and can protect server and networks from overloading in both network- and server-limited scenarios. NEWS also maintains high-performance for admitted requests during flash crowds.

We further examined run-time overhead of NEWS on router, and found that NEWS is a relatively light-weighted control scheme, and consumes small amount of resources on router.

To protect bystander traffic from flash crowds, we developed hot-spot identification algorithm to classify flash crowd traffic from others. Our experiments showed that this algorithm reduces the end-to-end latency for bystander web traffic dramatically.

Acknowledgments

We appreciate the fruitful discussion with members in *SAMAN* and *CONSER* projects. We are graceful to Professor Edmond A. Jonckheere for his insightful suggestions on presenting NEWS control diagrams.

References

[1] 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 International World Wide Web Conference*, pages 252–262, Hawaii, USA, May 2002. IEEE.

[2] William Lefebvre. CNN.com: Facing a world crisis. In *USENIX Large Installation Systems Administration Conference*, San Diego, CA, USA, December 2001. invited talk.

[3] Lars Eggert and John Heidemann. Application-level differentiated services for web servers. *World-Wide Web Journal*, 2(3):133–142, August 1999.

[4] Mohammad A. Rahin and Mourad Kara. Call admission control algorithms in atm networks: A performance comparison and research directions. Research report, University of Leeds, February 1998.

[5] P. Mundur, R. Simon, and A. Sood. Integrated admission control in hierarchical video-on-demand systems. In *Proceedings of the IEEE International Conference on Multimedia Computing and Systems*, pages 220–225, Florence, Italy, June 1999. IEEE.

[6] C. Cetinkaya and E. Knightly. Egress admission control. In *Proceedings of the IEEE Infocom*, pages 1471–1480, Tel-Aviv, Israel, March 2000. IEEE.

[7] Lee Breslau, Edward W. Knightly, Scott Shenker, Ion Stoica, and Hui Zhang. Endpoint admission control: Architectural issues and performance. In *Proceedings of the ACM SIGCOMM*, pages 57–69, Stockholm, Sweden, August 2000.

[8] Jamin. S., Danzig. P.B., Shenker. S, and Zhang. L. Measurement-based admission control algorithms for controlled-load service. In *Proceedings of the ACM SIGCOMM*, pages 2–13, Cambridge, MA, October 1995. ACM.

[9] Rudolf B Blazek, Hongjoong Kim, Boris Rozovskii, and Alexander Tartakovsky. A novel approach to detection of denial-of-service attacks via adaptive sequential and batch-sequential change-point detection methods. In *Proceedings of the IEEE Systems, Man, and Cybernetics Information Assurance Workshop*, West Point, NY, USA, June 2001. IEEE.

[10] A. Garg and A.L. Narasimha Reddy. Mitigation of DoS attacks through qos regulation. In *Proceedings of the IEEE International Workshop on Quality of Service*, Miami Beach, FL, USA, May 2002.

[11] Mark E. Crovella and Azer Bestavros. Self-similarity in world wide web traffic: evidence and possible causes. In *Proceedings of the ACM SIGMETRICS*, pages 160–169, Philadelphia, Pennsylvania, May 1996. ACM.

[12] VINT group. UCB/LBNL/VINT network simulator—ns (version 2), 1997. http://www.isi.edu/nsnam/ns/.

[13] X. Chen and J. Heidemann. Flash crowd mitigation via an adaptive admission control based on application-level measurement. Technical Report ISI-TR-557, USC/Information Sciences Institute, May 2002. under submission.

[14] Akamai Technologies Inc., 1995. http://www.akamai.com.

[15] Martin Arlitt and Tai Jin. A workload characterization study of the 1998 world cup web site. *IEEE Network, Special Issue on Web Performance*, 14(3):30–37, May 2000.

[16] Matt Welsh and David Culler. Adaptive overload control for busy internet servers. In *Proceedings of the USENIX Symposium on Internet Technologies and Systems*, Seattle, WA, USA, March 2003. USENIX.

[17] Hari Balakrishnan, Hariharan Rahul, and Srinivasan Seshan. An integrated congestion management architecture for internet hosts. In *Proceedings of the ACM SIGCOMM*, pages 175–187, Cambridge, MA, USA, September 1999. ACM.

[18] R. Mahajan, S. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker. Controlling high bandwidth aggregates in the network. Technical report, ICSI, July 2001.

[19] David D. Clark, Scott Shenker, and Lixia Zhang. Supporting real-time applications in an integrated services packet network: Architecture and mechanism. In *Proceedings of the ACM SIGCOMM*, pages 14–26, Baltimore, MD, USA, August 1992. ACM. citeseer.nj.nec.com/clark92supporting.html.

[20] R.J. Gibbens, F. P. Kelly, and P.B. Key. A decision-theoretic approach to call admission control in ATM networks. *IEEE Journal on Selected Areas in Communications*, 13(6):30–37, August 1995.

[21] Rich Wolski, Neil T. Spring, and Jim Hayes. The network weather service: A distributed resource performance forecasting service for metacomputing. *Journal of Future Generation Computing Systems*, 15(6):757–768, October 1999.

[22] H. Chen and P. Mohapatra. Session-based overload control in qos-aware web servers. In *Proceedings of the IEEE Infocom*, New York, NY, USA, June 2002. IEEE.

[23] C. Lu, T. F. Abdelzaher, J. A. Stankovic, and S. H. Son. A feedback control approach for guaranteeing relative delays in web servers. In *Proceedings of the IEEE Real-Time Technology and Applications Symposium*, Taipei, Taiwan, June 2001. IEEE.

[24] Netfilter/iptables project. Netfilter/iptables, 2003. http://www.netfilter.org/.

[25] D. Clark and W. Fang. Explicit allocation of best effort packet delivery service. *ACM/IEEE Transactions on Networking*, 6(4):362–373, August 1998.

[26] W. Fang, N. Seddigh, and B. Nandy. A time sliding window three colour marker (TSWTCM). RFC 2859, Internet Request For Comments, June 2000.

[27] NIST Internetworking Technology Group. NIST net network emulator, 1998. http://is2.antd.nist.gov/itg/nistnet/.

[28] Kun chan Lan and John Heidemann. Rapid model parameterization from traffic measurement. Technical Report ISI-TR-561, USC/Information Sciences Institute, October 2002. to appear in ACM Transaction on Modeling and Computer Simulation (TOMACS).

[29] Sally Floyd and Vern Paxson. Difficulties in simulating the Internet. *ACM/IEEE Transactions on Networking*, 9(4):392–403, August 2001.

[30] Chuck Lever, Marius Aamodt Eriksen, and Stephen P. Molloy. An analysis of the TUX web server. Technical report, Center for Information Technology Integration, University of Michigan, November 2000.

[31] Lawrence-Berkeley Labs. The Internet Traffic Archive. http://ita.ee.lbl.gov/, 1992.

[32] C. Estan and G. Varghese. New directions in traffic measurement and accounting. In *Proceedings of the ACM SIGCOMM*, Pittsburgh, PV, USA, August 2002. ACM.

[33] NetFlow performance analysis. White paper, Cisco Systems, Inc., 2003. http://www.cisco.com/warp/public/cc/pd/iosw/prodlit/ntfo_wp