Gerenciamento Autonômico de Capacidade em Face de Ataques de Segurança

Itamar Viana, João Palotti, Ítalo Cunha, Jussara Almeida, Virgílio Almeida

[1]Departamento de Ciência da Computação – Universidade Federal de Minas Gerais
Belo Horizonte, Brasil – 31270-010

{itamar, palotti, cunha, jussara, virgilio}@dcc.ufmg.br

Abstract. *In this work we present an extension of our previous capacity management framework making it robust to security attacks. In a scenario where multiple applications are hosted in a shared infrastructure, our performance and optimization models are extended to minimize the impact of security attacks on the provider's revenue and applications' QoS. A number of scenarios and strategies based on dynamic SLA contracts are designed to help uncover the main tradeoffs considering both the provider's interests and the customer's interests.*

Resumo. *Nesse trabalho apresentamos uma extensão do nosso arcabouço de gerenciamento de capacidade tornando-o robusto a ataques de segurança. Em um cenário no qual múltiplas aplicações são hospedadas em uma infra-estrutura compartilhada, nossos modelos de desempenho e otimização foram extendidos para minimizar o impacto dos ataques de segurança no lucro do servidor e no QoS das aplicações. Vários cenários e estratégias baseadas em SLAs dinâmicos foram formulados para descobrir os principais compromissos de custos e desempenho, considerando tanto os interesses do provedor quanto do cliente.*

1. Introdução

O gerenciamento de capacidade de uma infra-estrutura de hospedagem tem sido tradicionalmente desenvolvido como um conjunto de técnicas para se alcançar objetivos de desempenho. Entretanto, a qualidade do serviço prestado aos clientes e, em última instância, o lucro obtido pelo provedor da infra-estrutura dependem também de outros aspectos.

Um desses aspectos é relacionado ao impacto de ataques de segurança nas decisões de gerenciamento de capacidade. Apesar das várias técnicas existentes para defesa e recuperação de ataques [Gelenbe and Loukas 2007, Walfish et al. 2006, Mirkovic and Reiher 2004], relatórios recentes indicam que ataques de segurança, especialmente aqueles que têm como alvos aplicações específicas [Crosby et al. 2003, Kandula et al. 2005, Mirkovic and Reiher 2004], ainda causam perdas financeiras de grande magnitude [Gelenbe and Loukas 2007, Lesk 2007]. Durante tais ataques, requisições ilegítimas são admitidas no sistema, consumindo recursos disponíveis , o que impacta negativamente na qualidade de serviço provida às requisições legítimas concorrentes para a mesma aplicação. O provedor também é penalizado, pois não capitaliza sobre o uso dos recursos alocados para atender o tráfego de requisições ilegítimas.

Os ataques de segurança considerados nesse trabalho têm como alvo aplicações específicas, sendo feitos por meio de requisições HTTP ilegítimas que imitam o comportamento das requisições legítimas (possuem as mesmas demandas [Kandula et al. 2005]). A modelagem de ataques com demandas heterogêneas (p.ex., ataques semânticos [Crosby et al. 2003]) será deixada como trabalho futuro.

O gerenciamento de um centro de dados grande e complexo impõe vários desafios. As soluções com uma boa relação custo-benefício não podem abordar somente um desses separadamente. O gerenciamento de capacidade demanda modelos e ferramentas que os tratem conjuntamente. A maioria das estratégias presentes na literatura focam somente em questões de desempenho [B. Urgaonkar, et al. 2005, Abrahão et al. 2006, Cunha et al. 2007]. Em particular, propomos um modelo autonômico, isto é, auto-adaptativo, de gerenciamento de capacidade que desloca dinamicamente os recursos entre as aplicações existentes de forma a maximizar os objetivos de negócio do provedor [Abrahão et al. 2006, Cunha et al. 2007]. Definimos nosso arcabouço como a combinação de um modelo de custo, baseado em contratos de acordo de nível de serviço (*Service Level Agreement*, SLA), um modelo de desempenho, baseado em redes de filas, e um modelo de otimização. Ele provê garantias sob a taxa de processamento das requisições, sob a cauda da distribuição do tempo de resposta e captura os principais compromissos de uma plataforma virtual multicamadas. Além disto, a maioria dos trabalhos prévios que abordam ataques de segurança objetivam aumentar a robustez dos sistemas ([Mirkovic and Reiher 2004] e referências nele), estando desassociados dos objetivos de gerenciamento de capacidade.

Este trabalho traz uma abordagem inovadora para este problema, objetivando capturar, em um arcabouço unificado, as características principais relacionadas a desempenho quando considerado questões de segurança. Objetivamos projetar alguma luz sobre as seguintes questões: (1) *quais são os principais compromissos entre desempenho e custos para gerenciamento de capacidade em cenários com ataques de segurança?* e (2) *dado que uma aplicação hospedada é alvo de um ataque de segurança qual é o custo-benefício de se utilizar SLAs dinâmicos, tanto do ponto de vista do provedor quanto dos clientes?* Foram executadas várias simulações, com cargas sintéticas e realistas, para entender os principais compromissos considerando o lucro do provedor e, para o cliente, a taxa de processamento de requisições legítimas, a distribuição do tempo de resposta e a quantia paga por cada requisição legítima servida.

Nossas principais descobertas são: (1) quando uma aplicação está sob ataque, nosso novo arcabouço, estendido para ter ciência dos ataques, aumenta significativamente os lucros do provedor deslocando capacidade entre as aplicações hospedadas de acordo com o ataque, a custo de grandes penalidades na quantidade de requisições atendidas da aplicação vítima; (2) as penalidades podem ser reduzidas (e até mesmo eliminadas), se as vítimas concordarem em pagar uma quantidade extra por cada requisição legítima servida; (3) alternativamente, as penalidades também podem ser reduzidas se a vítima concordar em relaxar o requisito SLA de tempo de resposta, mas o efeito dessa política depende do contrato original e do fator de relaxamento.

Este projeto de pesquisa esteve sob orientação da professora Jussara Almeida e assistência dos co-orientadores Ítalo Cunha e Virgílio Almeida ao longo de todo o desenvolvimento, sendo a participação do João Palotti na síntese dos resultados. Os novos modelos e resultados aqui apresentados são uma parte do que foi feito no artigo [Cunha et al. 2008] (a ser publicado), que além de considerar ataques de segurança, abordou também custos e restrições de energia. Como nas pesquisas meu foco maior foi em ataques de segurança, preferi não abordar a modelagem de energia nesse artigo.

Esse artigo é organizado como a seguir. A Seção 2 revê nosso arcabouço de gerenciamento de capacidade e outros trabalhos relacionados. As extensões introduzidas no arcabouço para capturar os aspectos principais de segurança são apresentadas na Seção

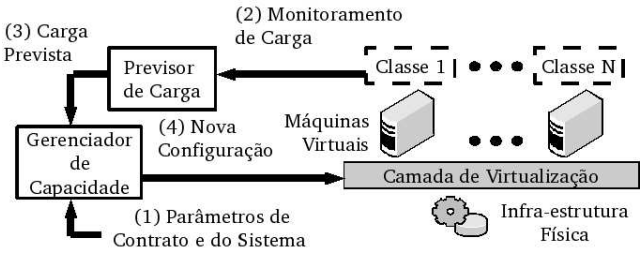


Figura 1. Gerenciamento Autonômico de Capacidade (perspectiva de 1 camada)

3. Os resultados das simulações são mostrados na Seção 4. Finalmente, as conclusões e trabalhos futuros aparecem na Seção 5.

2. Contexto

2.1. Gerenciamento Autonômico de Capacidade para Serviços com Multicamadas

Consideramos um cenário no qual um provedor hospeda múltiplos serviços Web de terceiros com diferentes padrões de carga e demandas por recursos. Tipicamente, tais serviços são compostos de diferentes tipos de requisições, que referenciamos como *classe de aplicação*. Assumimos que a infra-estrutura hospeda N classes independentes oriundas de todos os serviços e que a infra-estrutura do provedor é composta de múltiplas (K) camadas, caso comum para vários serviços Web. Cada camada executa um mecanismo de virtualização que permite a criação e alocação dinâmica de seus recursos para N máquinas virtuais (MV) [Barham et al. 2003]. Assim, cada classe é executada em K MVs dedicadas, uma para cada camada. Depois de ser servida na camada j, uma requisição da classe i deixa o sistema com probabilidade $p_{i,j}$ ($i = 1..N, j = 1..K, p_{i,K} = 1$).

Nesse ambiente alvo, o problema de gerenciamento de capacidade é definido como a determinação da fração da capacidade física que deve ser alocada a cada classe i em cada camada j, bem como a taxa de requisições de cada classe i que deve ser admitida para processamento de forma a maximizar o objetivo de negócio do provedor, tendo como restrição os contratos de nível de serviço (SLA) firmados com os clientes.

Em [Abrahão et al. 2006, Cunha et al. 2007], propomos um arcabouço auto-adaptativo para gerenciamento de capacidade que funciona da seguinte maneira: periodicamente, o *gerenciador de capacidade* recebe uma previsão da carga de trabalho, λ_i, esperada para cada classe, i, bem como os requisitos SLA das mesmas, a média de tempo de serviço por camada, $d_{i,j}$, as probabilidades de roteamento, $p_{i,j}$, e os parâmetros do sistema. De posse desses dados ele calcula a alocação de capacidade, $f_{i,j}$, para o próximo intervalo assim como a taxa de requisições, λ_i^{acc}, que devem ser admitidas. Para isso, ele utiliza um método de previsão de cargas [Abraham and Ledolter 1983] baseado na monitoração das cargas de cada classe para prever a carga esperada para o próximo intervalo de tempo (intervalo de controle), bem como um mecanismo de controle de admissão [Perros and Elsayed 2003] para limitar a taxa de requisições aceitas por uma camada. A Figura 1 mostra uma iteração da perspectiva de *uma* camada.

O gerenciador de capacidade combina um modelo de custo e um de desempenho do sistema em um modelo de otimização para resolver o problema de alocação de capacidade. As principais características desses modelos são resumidas a seguir.

2.1.1. Modelo de Custo

O modelo de custo especifica requisitos de qualidade de serviço e custos, capturando assim os contratos SLAs. Nosso modelo de custo é baseado em dois modos de operação,

o normal e o sobrecarregado. Para o modo de operação normal, é definido X_i^N, como a *taxa de requisições válidas* para cada classe i que o provedor deve atender, dado que a chegada de requisições seja alta o suficiente. No caso de violações do contrato, o provedor concorda em devolver parte do pagamento recebido pelo serviço aos seus clientes. Para o modo de operação sobrecarregado, o contrato define $X_i^S \geq X_i^N$, um limite para a taxa de processamento de requisições válidas até onde o cliente concorda em pagar uma recompensa ao provedor por servir requisições acima de X_i^N. Os valores das penalidades, c_i, e recompensas, r_i, são calculados usando os parâmetros do contrato SLA por unidade de taxa de processamento abaixo ou acima de X_i^N, respectivamente.

A taxa de processamento válida inclui todas as requisições cujos tempos de resposta satisfazem o SLA. Consideramos um requisito sobre a *cauda* da distribuição do tempo de resposta, isto é, um limite na probabilidade do tempo de resposta de uma requisição da classe i exceder um certo limiar R_i^{SLA}. Em outras palavras, consideramos $P(R_i > R_i^{SLA}) \leq \alpha_i$, onde R_i é o tempo de resposta de uma requisição da classe i. Dada a taxa de chegada de requisições da classe i (λ_i) e a taxa de processamento aceita para a classe i (λ_i^{acc}), calculada pelo gerenciador de capacidade, o lucro do provedor obtido da classe i (g_i) é dado por:

$$g_i = \begin{cases} -c_i \left(min(\lambda_i, X_i^N) - \lambda_i^{acc} \right) & \lambda_i^{acc} \leq X_i^N \\ r_i \left(\lambda_i^{acc} - X_i^N \right) & X_i^N < \lambda_i^{acc} \leq X_i^S \end{cases} \quad (1)$$

2.1.2. Modelo de Desempenho

O modelo de desempenho, baseado em teoria de filas, estima para cada classe hospedada, a utilização por camada, a taxa de processamento e a probabilidade de violação do requisito do tempo de resposta. Definimos $d_{i,j}^*$ como o tempo médio de serviço para uma requisição da classe i, executando na camada j com sua capacidade total, que pode ser estimado em um ambiente de pré-produção e inflacionado para capturar a sobrecarga da virtualização [Almeida et al. 2006]. O tempo médio de serviço para uma requisição da classe i assinalada para a MV na camada j, $d_{i,j}$, é então calculado como $d_{i,j} = d_{i,j}^*/f_{i,j}$, sendo $f_{i,j}$ a fração da capacidade da camada j alocada para a classe i. A taxa efetiva de chegada para a classe i na camada j é dada por $\lambda_{i,j}^e$, sendo calculada a partir de λ_i^{acc} e o vetor de probabilidade $p_{i,j}$, $j = 1..K$ [Cunha et al. 2007]. Finalmente, a utilização máxima planejada para a MV da classe i na camada j é fixada em $\nu_{i,j}$ para evitar saturação.

Nosso modelo assume que chegadas de requisições para cada classe i seguem um processo de Poisson com taxa λ_i, como visto em sistemas reais [Menascé and Bennani 2006]. Assumimos que as classes possuem tempos de serviço exponencialmente distribuídos em cada camada j, com média $d_{i,j}$. Então cada MV é modelada como uma fila M/M/1 com escalonamento FCFS [Kleinrock 1975], como utilizado em centros de serviços transacionais [B. Urgaonkar, et al. 2005]. Logo, cada classe é modelada como uma seqüência de filas M/M/1, com tempos de residência independentes.

Sobre essas premissas, a taxa de processamento da classe i é igual a λ_i^{acc}, e a utilização da classe i na camada j é dada por $\rho_{i,j}$=$\lambda_{i,j}^e d_{i,j}$. A probabilidade de uma requisição da classe i violar o tempo de resposta R_i^{SLA}, isto é, $P(R_i \geq R_i^{SLA})$, pode ser derivada da distribuição do tempo de resposta R_i, que, para filas M/M/1, é uma hipoexponencial [Kleinrock 1975], com parâmetros $d_{i,j}$ e $\lambda_{i,j}^e$ (veja Equação 1 em [Cunha et al. 2007]). Alternativamente, pode ser usada uma aproximação mais simples para $P(R_i \geq R_i^{SLA})$ baseada na Desigualdade de Chebyshev [Kleinrock 1975]. Porém,

como a solução baseada na distribuição hipoexponencial é exata e possui tempos de solução razoáveis [Cunha et al. 2007], ela será considerada como linha de base.

2.1.3. Modelo de Otimização

Os modelos de custo e desempenho são combinados em um modelo de otimização com uma função objetivo que expressa os interesses do provedor de maximizar o lucro total (somatório da Eq. 1 para todas as classes). Além do mais, são adicionadas restrições para especificar limites na taxa de requisições aceitas, alocação de capacidade por camada, utilização das MVs e taxa efetiva de requisições aceitas, assim como expressar o requisito da cauda da distribuição do tempo de resposta do SLA para cada classe. Uma descrição detalhada do modelo de otimização, bem como análise de convergência, tempos de solução e avaliação para diferentes cenários foram apresentados em [Cunha et al. 2007].

2.2. Trabalhos Relacionados

Gerenciamento de capacidade tem sido alvo de vários estudos prévios. Alguns deles focam em estratégias de controle de admissão [Popovici and Wilkes 2005]. Outros tratam da alocação de capacidade entre aplicações hospedadas em uma mesma infra-estrutura compartilhada [B. Urgaonkar, et al. 2005, Menascé and Bennani 2006]. Outros trabalhos [Almeida et al. 2006, Abrahão et al. 2006, Cunha et al. 2007] combinam ambos os esquemas dentro do arcabouço autonômico de gerenciamento de capacidade. Entretanto, muitos desses esforços não abordam uma ou mais das seguintes características desejadas: modelos de custos alinhados com os interesses de negócio do provedor [Menascé and Bennani 2006], garantias probabilísticas sobre os objetivos de desempenho, e modelagem de plataformas multicamadas. Em particular, [Abrahão et al. 2006, Almeida et al. 2006] combinam modelos de custos, baseados em SLAs, com modelos de desempenho, baseados em teoria de filas, para derivar garantias sobre a *cauda* da distribuição do tempo de resposta, visando maximizar o lucro do provedor. Entretanto, esses estudos foram feitos considerando plataformas de camada única. Plataformas multicamadas foram estudadas em [B. Urgaonkar, et al. 2005], mas a solução proposta não foi acoplada a um modelo de custos, além de prover garantias apenas sob o desempenho *médio*. Os esquemas de gerenciamento anteriores, incluindo nosso arcabouço apresentado na Seção 2.1, focam principalmente em desempenho, não abordando questões de segurança que podem afetar a relação custo-benefício das decisões de gerenciamento.

Uma longa lista sobre ataques DDoS e mecanismos de defesa, incluindo métodos baseados em IP *backscatter*, detecções de anomalias baseadas em *wavelets* e detecção de intrusão baseado em assinatura, é dada em [Mirkovic and Reiher 2004]. Um arcabouço para classificar ataques DDoS utilizando análise espectral foi apresentado em [Hussain et al. 2003]. Outros mecanismos de defesa implantados na rede [Gelenbe and Loukas 2007] e na vítima [Walfish et al. 2006] também estão disponíveis, mas nenhum desses esforços foi acoplado ao contexto de gerenciamento de capacidade.

3. Modelagem de Ataques de Segurança

Focamos em ataques de segurança que têm como alvo aplicações específicas através de inundação de requisições ilegítimas (requisições de HTTP [Kandula et al. 2005] e *spams*, por exemplo), que apesar de consumirem recursos, não geram lucro. O impacto desses ataques na infra-estrutura e nas aplicações é de particular interesse porque, mesmo

que eles possam ser detectados [1], as requisições ilegítimas não podem ser individualmente bloqueadas antes de serem processadas, uma vez que apresentam características de requisições legítimas. Na verdade, tais ataques foram recentemente reportados como causas de grandes perdas financeiras[Kandula et al. 2005, Gelenbe and Loukas 2007, Lesk 2007]. Além disso, esses ataques são exemplos interessantes de cenários em que o gerenciador de capacidade pode atuar para minimizar a degradação decorrente[2].

Com o propósito de capturar o impacto *primário* de ataques em (classes de) aplicações, nosso arcabouço foi modificado como descrito a seguir. A taxa total de requisições da classe i, λ_i, é subdividida em λ_i^+, para requisições legítimas, e λ_i^-, para as ilegítimas, logo, $\lambda_i = \lambda_i^+ + \lambda_i^-$. Além disso, uma vez que requisições ilegítimas não podem ser individualmente identificadas, elas são admitidas no sistema. A taxa de requisições processadas da classe i é subdividida em legítimas, dada por $\lambda_i^{acc} \times (\lambda_i^+/\lambda_i)$, e ilegítimas, dada por $\lambda_i^{acc} \times (\lambda_i^-/\lambda_i)$. Dessa forma, servir uma fração das requisições legítimas da classe i implica em custos extras para o provedor, uma vez que algumas requisições ilegítimas também serão admitidas no sistema, consumindo recursos. Além disso, assumimos que as requisições legítimas e ilegítimas têm o mesmo tempo médio de serviço por camada ($d_{i,j}^*$), como esperado para ataques que tentam imitar o comportamento dos usuários [Kandula et al. 2005]). A modelagem de ataques com demandas heterogêneas (p.ex: ataques semânticos [Crosby et al. 2003]) é trabalho futuro. Dadas essas premissas, o lucro obtido com a classe i, calculado somente sobre a taxa de requisições legítimas processadas, λ_i^{bom}=$\lambda_i^{acc} \times \lambda_i^+/\lambda_i$, é:

$$g_i^{seg} = \begin{cases} -c_i \left(min(\lambda_i^*, X_i^N) - \lambda_i^{bom} \right) & \lambda_i^{bom} \leq X_i^N \\ r_i \left(\lambda_i^{bom} - X_i^N \right) & X_i^N < \lambda_i^{bom} \leq X_i^S \end{cases} \quad (2)$$

Consideramos quatro soluções alternativas para gerenciamento de capacidade, construídas a partir de variantes do nosso arcabouço original:

• *Indiferente a Ataques* (IA): o provedor não está ciente que a aplicação (classe) i está sob ataque, isto é, λ_i^- é desconhecido. O gerenciamento de capacidade é feito usando nosso arcabouço original, que utiliza estimativas de lucro dadas pela Equação 1. Entretanto, os lucros reais obtidos são computados usando a Equação 2.

• *Ciente de Ataques* (CA): o gerenciador de capacidade age de acordo com a extensão do arcabouço descrita nessa seção.

• *Ciente de Ataques com Custo Adaptativo* (CA-C): quando sob ataque, o cliente vítima concorda em pagar uma quantia para cada requisição legítima servida que corresponde ao valor acordado no SLA inflacionado por λ_i/λ_i^+. Em outras palavras, ele concorda pagar o valor original para todas as requisições servidas, legítimas ou não.

• *Ciente de Ataques com o Requisito de Tempo de Resposta (R_i^{SLA}) Dinâmico* (CA-R): quando sob ataque, a vítima concorda em relaxar R_i^{SLA} por um fator proporcional ao peso do ataque, fazendo $R_i^{SLA,seg} = R_i^{SLA}(1 + w_i^-(\lambda_i^-/\lambda_i))$, onde w_i^- é o fator de relaxamento.

As estratégias CA-C e CA-R são baseadas em contratos SLAs dinâmicos, que, quando frente a um ataque, permitem o inflacionamento, ou do valor pago pelo cliente para cada requisição legítima servida, ou do requisito de tempo de resposta. SLAs

[1] Perfis de cargas de trabalho típicas podem ser usados para detectar mudanças repentinas que não possam ser explicadas por anomalias como *Flash Crowds*.

[2] Ataques na infra-estrutura (como inundação na banda de rede) normalmente requerem esquemas de defesa implantados na rede [Gelenbe and Loukas 2007], e portanto estão fora do escopo de nosso arcabouço.

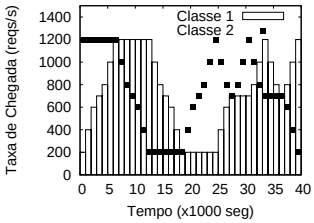


Figura 2. Carga Sintética, $w_1^- = 5$

dinâmicos podem ser vantajosos tanto para clientes vítimas de ataques, que terão um maior número de requisições legítimas servidas, quanto para provedores, que receberão pagamentos mais altos ou terão maior flexibilidade. Portanto, eles podem conduzir a um melhor compromisso entre possíveis conflitos de interesses entre as partes.

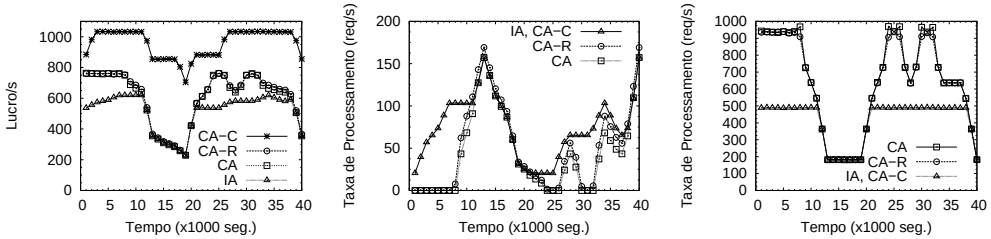
4. Cenário de Avaliação e Análise dos Resultados

Esta seção mostra resultados de simulações que ilustram os principais compromissos para o gerenciamento de capacidade diante de ataques de segurança (Seção 3). As métricas consideradas são o lucro do provedor e, do lado dos clientes, a taxa de processamento de requisições legítimas, a distribuição do tempo de resposta e o custo por requisição legítima atendida. As cargas de trabalho e as configurações do sistema utilizadas na avaliação são descritas a seguir. Os resultados apresentados são valores médios de 5 execuções, com coeficiente de variação abaixo de 2%.

No cenário de avaliação a infra-estrutura contém duas camadas ($K = 2$), que são visitadas por todas as requisições (isto é, $p_{i,1} = 0$, $\forall i$). Assumimos que as classes de aplicações possuem parâmetros de contrato homogêneos, definimos $\alpha_i = 0,1$, $c_i = 1$, $r_i = 0,5$, $\nu_{i,j} = 0,95$ e, a menos que seja dito o contrário, $R_i^{SLA} = 20 \sum_{j=1}^{K} d_{i,j}^*$. Assumimos também que a carga de trabalho é conhecida *a priori*, e que o gerenciador de capacidade é executado sempre que a taxa de requisições (legítimas e ilegítimas) muda. Em [Cunha et al. 2007], mostramos que essa simplificação causa um impacto pequeno (menor que 11%) nos lucros do provedor. O lucro é calculado considerando-se apenas os pagamentos definidos no contrato, sendo, portanto, sempre positivo.

O cenário de avaliação, construído a partir de cargas sintéticas, consiste de duas classes com requisições, cujas chegadas seguem processos de Poisson não homogêneos em degraus, mostrados na Figura 2, a qual cobre diferentes padrões (carga crescente versus decrescente, alta versus baixa). Cada degrau dura 1000 segundos. Os tempos médios de serviço, $d_{i,j}^*$, são mostrados na Tabela 1, juntamente com os parâmetros de SLA (taxas de processamento e tempo de resposta) para cada classe. A classe 1 está sob ataque durante toda simulação a uma taxa $\lambda_1^- = 5000$ req/s e a clase 2 não sofre ataque. Começamos discutindo os resultados mais relevantes, a Figura 3-a mostra o lucro do provedor para as quatro soluções de gerenciamento de capacidade analisadas. As Figuras 3-b e 3-c mostram a taxa de processamento de requisições legítimas para as classes 1 e 2.

As principais diferenças entre as soluções Indiferente a Ataques (IA) e Ciente de Ataques (CA) são explicadas pelo fato de CA retirar uma fração dos recursos alocados para a classe 1 (sob ataque), realocando-os para a classe 2 a fim de capitalizar sobre uma maior taxa de requisições legítimas da classe 2. A solução CA evita alocar capacidade para requisições ilegítimas, aumentando, assim, o total de requisições legítimas atendidas e, logo, o lucro do provedor. Quanto mais pesada for a carga na classe 2, maior será a fração dos recursos realocada para ela, e mais altos serão os ganhos de CA sobre IA.



(a) Lucro/s (b) Classe 1 (c) Classe 2

Figura 3. Lucro e Taxa de Processamento legítimo ($\lambda_1^- = 5000$ req/s)

Tabela 1. Parâmetros das Classes de Aplicação

Cenário	i	$d_{i,1}^*$	$d_{i,2}^*$	R_i^{SLA}	X_i^N	X_i^S
Sintético	1	0,9 ms	0,6 ms	0,03 s	500	1200
	2	0,6 ms	0,9 ms	0,03 s	500	1200

Por exemplo, durante os instantes 5000 e 30000, quando a carga da classe 2 é alta (veja Figura 2), os lucros obtidos com CA são muito maiores do que os obtidos com IA (Figura 3-a), ao custo de uma redução na taxa de processamento da classe 1 e um aumento na taxa de processamento da classe 2 (Figuras 3-b e 3-c). Além disso, se o ataque for muito pesado (ex: até o instante 7000), a classe vítima pode ser *desligada* por CA. Porém, se a carga da classe 2 for baixa (ex: instante 15000), não há benefício em deslocar capacidade e, ambas soluções são equivalentes. Em geral, CA provê ganhos significativos no lucro obtido, sobre IA (16% em média). Para os clientes, o tempo de resposta e os custos não são afetados, mas a taxa de processamento da vítima é muito penalizada (41%), enquanto a classe 2 se beneficia da capacidade extra, atingindo uma taxa 34% maior.

Quanto às soluções baseadas em SLAs dinâmicos, a solução Ciente de Ataques com Custo Adaptativo (CA-C), diferentemente da CA, não retira capacidade da classe vítima, já que os custos extras pagos pelo cliente eliminam o impacto do ataque no lucro do servidor. De fato, os lucros da solução CA-C são os maiores das quatro soluções (em média 59% maior que CA). Além do mais, a taxa de requisições legítimas servidas da vítima é tão alta quanto na solução IA, e 70% maior que na CA, em média. Entretanto, o valor pago pela vítima por cada requisição legítima servida é, em média, quase 7 vezes maior, o que pode ser interessante somente para aplicações críticas.

A solução CA-R tira proveito do relaxamento no tempo de resposta ($w_1^- = 5$) para admitir um maior número de requisições da classe 1 no sistema, aumentando as utilizações das MVs. Comparada à CA, a CA-R aumenta a taxa de processamento de requisições legítimas da classe vítima em 18%, em média, com um pequeno impacto na taxa de processamento da classe 2. Em compensação, o 90^o percentil do tempo de resposta cresce de 0,03s para 0,13s. Se o acréscimo do atraso for aceitável, a CA-R pode ser uma opção com boa relação custo-benefício para aplicações sob ataque. Para o provedor, CA-R resulta em leve aumento nos lucros (1,2%) em relação a CA.

Esses resultados são sumarizados na Tabela 2, que mostra os valores agregados (médios) do lucro do provedor, do custo por requisição e da taxa de processamento das requisições legítimas, para cada classe.

O impacto do peso do ataque (isto é, da taxa de requisições ilegítimas) no lucro médio obtido é mostrado na Figura 4. A diferença entre CA e IA aumenta com λ_i^-, como era esperado. Além disto, a solução CA-C leva a um pequeno aumento no lucro quando há ataques ($\lambda_1^- > 0$). Isso ocorre devido à alocação de uma pequena fração de capacidade

Tabela 2. Resumo do Impacto dos Ataques de Segurança: Resultados Médios considerando somente as requisições legítimas (λ_1^-=5000 req/s, w_1^-=5)

Solução	Lucros/s	Classe 1		Classe 2	
		Custo/Req	Req/s	Custo/Req	Req/s
IA	518	1	74,7 req/s	0,97	456 req/s
CA	604	1	44,0 req/s	0,81	689 req/s
CA-C	959	6,89	74,7 req/s	0,97	455 req/s
CA-R	611	1	51,9 req/s	0,81	687 req/s

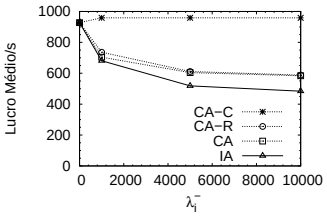


Figura 4. Lucro Médio vs. λ_1^- (w_1^-=5)

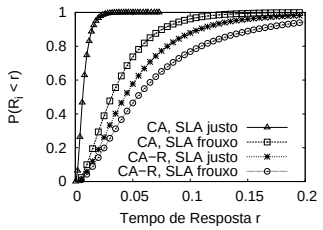


Figura 5. Impacto do CA-R no R_i (λ_1^-=5000 req/s, w_1^-=5)

antes ociosa para servir requisições ilegítimas, o que na solução CA-C gera lucros.

Quanto às diferenças entre as soluções CA e CA-R, três pontos merecem menção. Primeiro, as diferenças nos lucros e nas taxas de processamento dependem do peso do ataque. Quanto maior o ataque, menor a fração de requisições legítimas servidas e menores os ganhos de se relaxar o R_i^{SLA} da vítima (ver Figura 4). Segundo, quanto mais justo for o R_i^{SLA}, maior o impacto de relaxá-lo. Isso ocorre devido à relação entre utilização (das MVs) e tempo de resposta não ser linear. Os ganhos obtidos com o relaxamento do R_i^{SLA} original são maiores quando a utilização é baixa. Isso é ilustrado na Figura 5, que apresenta a distribuição do tempo de resposta para CA e CA-R, com o R_i^{SLA} original igual a $10d$ (justo) e $50d$ (frouxo), onde d=$\sum_{j=1}^{K} d_{i,j}^{*}$. A distância entre as curvas CA e CA-R para cada caso (justo e frouxo) captura os aumentos na utilização das MVs e na taxa de processamento. A taxa de processamento de requisições legítimas da vítima (e o lucro) dados pela solução CA-R aumenta de 4% (0,5%) para o R_i^{SLA} frouxo e de 77% (3,1%) para o justo, sem impacto na taxa de processamento da classe 2. Por fim, todos os resultados mostrados são para um fator de relaxamento de w_1^-=5. Como a utilização das MVs já está próxima do máximo permitido ($\nu_{i,j}$), valores maiores de w_1^- levam a resultados similares. Valores menores levam a diferenças menos significativas entre CA-R e CA.

Foram feitas simulações com uma carga realista derivada a partir de registros de um portal Web. Os resultados ilustram os mesmos compromissos identificados no cenário sintético. Eles foram apresentados em [Cunha et al. 2008] e não foram discutidos aqui devido à restrição de espaço.

5. Conclusão e Trabalhos Futuros

Nesse artigo, estendemos o nosso arcabouço de gerenciamento de capacidade para capturar os compromissos introduzidos por ataques de segurança e através de experimentos obtivemos as seguintes conclusões. A introdução dos modelos de custo de ataques de segurança no arcabouço leva a lucros muito maiores para o provedor, ao custo de uma degradação significativa na taxa de requisições servidas, particularmente para aplicações sob ataque. Porém, estratégias baseadas em SLAs dinâmicos podem contribuir para reduzir esta degradação ao custo de um aumento, ou no valor pago pelo cliente por requisição

legítima servida, ou no tempo de resposta.

Trabalhos futuros incluem modelos mais sofisticados de ataques de segurança (p.ex., ataques semânticos) e extensões para considerar a perspectiva do usuário final.

Agradecimento

Este trabalho foi desenvolvido em colaboração com a HP Brasil R&D.

Referências

Abraham, B. and Ledolter, J. (1983). *Statistical Methods for Forecasting*. John Wiley and Sons.

Abrahão, B., Almeida, V., Almeida, J., et al. (2006). Self-Adaptive SLA-Driven Capacity Management for Internet Services. In *Proc. IEEE/IFIP NOMS*, Vancouver, Canada.

Almeida, J., Almeida, V., et al. (2006). Resource Management in the Autonomic Service-Oriented Architecture. In *Proc. 3rd IEEE ICAC*, Dublin, Ireland.

B. Urgaonkar, et al. (2005). Dynamic Provisioning of Multi-tier Internet Applications. In *Proc. 2nd IEEE ICAC*, Seattle, WA.

Barham, P. et al. (2003). Xen and the Art of Virtualization. In *Proc. 19th ACM SOSP*, Bolton Landing, NY.

Crosby, S. et al. (2003). Denial of Service via Algorithmic Complexity Attacks. In *Proc. 12th USENIX Sec. Symp.*, Washington, DC.

Cunha, Í., Almeida, J., Almeida, V., and Santos, M. (2007). Gerenciamento de Capacidade Auto-Adaptativo para Ambientes Virtualizados Multicamadas. In *Proc. SBRC*, Belém, Brasil.

Cunha, Í., Viana, I., Palotti, J., Almeida, J., and Almeida, V. (2008). Analyzing Security and Energy Tradeoffs in Autonomic Capacity Management. In *Proc. IEEE/IFIP NOMS*, Salvador, Brazil.

Gelenbe, E. and Loukas, G. (2007). A Self-Aware Approach to Denial of Service Defence. *Computer Networks*, 51(5).

Hussain, A. et al. (2003). A Framework for Classifying Denial of Service Attacks. In *Proc. ACM SIGCOMM*, Karlsruhe, Germany.

Kandula, S. et al. (2005). Botz-4-Sale: Surviving Organized DDoS Attacks That Mimic Flash Crowds. In *Proc. 2nd NSDI*, Boston, MA.

Kleinrock, L. (1975). *Queueing Systems*. John Wiley and Sons.

Lesk, M. (2007). The New Front Line: Estonia under Cyberassault. *IEEE Security & Privacy*, 5(4).

Menascé, D. and Bennani, M. (2006). Autonomic Virtualized Environments. In *Proc. 2nd IEEE ICAS*, Silicon Valley, CA.

Mirkovic, J. and Reiher, P. (2004). A Taxonomy of DDoS Attack and DDoS Defense Mechanisms. *Comp. Comm. Rev.*, 34(2).

Perros, H. and Elsayed, K. (2003). Call Admission Control Schemes: A Review. *IEEE Communications Magazine*, 34(11).

Popovici, F. and Wilkes, J. (2005). Profitable Services in an Uncertain World. In *Proc. ACM/IEEE SC*, Seattle, WA.

Walfish, M. et al. (2006). DDoS Defense by Offense. In *Proc. ACM SIGCOMM*, Pisa, Italy.