O *teste de carga* é realizado para verificar qual o **volume de transações**, **acessos simultâneos**ou **usuários** que um servidor/software/sistema suporta.

Cenários de teste de carga incluirão pontos de validação, transações e medições.

O teste de carga é realizado para verificar qual o volume de transações, acessos simultâneos ou usuários que um servidor/software/sistema suporta. Serve basicamente para medir o volume de carga suportado. Através desse procedimento é possível:

– Analisar a estabilidade de um servidor ou sistema em um período de grande carga de modo a estabelecer um limite de operação;

– Medir o desempenho de um software mediante um grande fluxo de dados e informações;

– Encontrar itens do sistema que podem vir a falhar ou incorrer em erros durante momentos de grande carga;

– Evidenciar como o sistema se comporta enquanto a carga de informações vai aumentando, de modo a verificar lentidão, falhas, bugs etc.

– Fornecer dados e informações para que o sistema/site seja melhorado e otimizado. Entre essas informações, temos o tempo de resposta das páginas;

**Qual a importância dos testes de carga?**

Por meio dos testes de carga é possível avaliar a potência de um sistema em relação ao recebimento de acessos simultâneos, de modo a ampliá-la se necessário ou prevenir falhas. Desse modo, pode-se saber até quando um sistema irá funcionar de modo adequado, com rapidez e agilidade, e quando é necessária uma intervenção ou otimização.

**Softwares que podem ser úteis nos testes de carga**

**JMeter** – esse software é um dos mais conhecidos do mercado na hora de simular acessos para a realização de testes de carga e medir o desempenho de aplicações. Feito todo em Java, com código open source, ele pode ser usado no teste de servidores JMS, LDAP, SOAP, Web (HTTPS e HTTP) entre outros. O foco desse software é conseguir simular cenários realistas de utilização dos sistemas pelos usuários, de modo a entregar resultados e informações mais próximas possíveis do que ocorre.

**Teste de stress** é realizado para submeter o software a situações extremas. Basicamente, o teste de stress baseia-se em testar os limites do software e avaliar seu comportamento. Assim, avalia-se até quando o software pode ser exigido e quais as falhas (se existirem) decorrentes do teste.

Vamos considerar como exemplo a implementação de um servidor web utilizado em e-commerce. Ao estabelecer os requisitos do sistema, fixou-se um máximo de 5000 transações por minuto para uma determinada plataforma de execução. Um teste de stress pode, então, ser feito para responder a várias perguntas:

         O sistema consegue atingir o objetivo?

         Qual o número máximo de transações realmente possível?

         Se a plataforma de execução se degradar (por exemplo, uma falha parcial de rede, falta de espaço em disco, etc.), como o sistema se comportará?

Em certos casos é preferível que a execução do programa seja mantida mesmo que se degrade, evitando uma parada completa. Um exemplo típico são sistemas financeiros.

Possibilidades de falhas sob condições de operação difícil, como registrar operações incorretas, devem ser detectadas e evitadas. Um bom teste de stress deve poder revelar essas informações aos avaliadores.

O teste de stress é uma forma de medir os limites de um software/servidor/site/hardware. É uma maneira de avaliar se o software funciona bem no limite e a partir de que ponto ele não consegue mais funcionar direito, apresentando erros.

Basicamente, utilizam-se procedimentos que visem avaliar a capacidade máxima de um software/site, para se descobrir quais erros que podem ocorrer antes dele atingir esse limite. Exemplos:

– Você pode usar o teste para constatar quantos acessos um servidor pode suportar ao mesmo tempo. Digamos que o limite dele seja de 10.000 usuários para o seu site. Por meio do teste de stress você pode confirmar ou refutar essa informação. Se ele cair em 9.000 acessos, você poderá levantar informações e tentar descobrir porque ocorreu a queda antes do limite, se foi erro do sistema ou se era essa a capacidade mesmo;

– Dá para você avaliar a quantidade máxima e a mínima de memória que um programa utiliza para funcionar adequadamente num pc, de modo a comparar com as informações fornecidas pelo fabricante ou desenvolvedor;

– Pode verificar quantas transações comerciais feitas por minuto num site de e-commerce um servidor aguenta;

– Quantos programas podem ser rodados ao mesmo tempo num pc etc.

**Por que ele é importante?**

Imagine só que um dia você resolveu investir em publicidade para atrair clientes para o seu site. Porém, uma hora depois que os seus anúncios no Google Adwords começaram a rodar, o seu site sai fora do ar por causa do alto número de acessos simultâneos. Entretanto, o limite preestabelecido para isso ocorrer nem foi atingido. Se o site suportaria em torno de 20.000 acessos simultâneos, porque caiu com 13.000?

Se tivesse feito um teste de stress antes, poderia ter descoberto qual era capacidade máxima real do servidor que hospeda o seu site ou qual o erro que ocasionou a queda. Desse modo, você poderia ter investido no aumento dessa capacidade ou trocado de servidor.

Agora, pense sobre isso acontecendo com uma empresa de grande porte que trocou seus servidores sem testar o limite dos novos. Consegue imaginar o tamanho do prejuízo que ela pode ter?

Sem falar que existem falhas que são mais fáceis de serem descobertas nesse tipo de teste do que em outros.

**Diferença entre teste de carga e teste de stress**

Vale ressaltar a diferença para o teste de stress, que embora pareça semelhante, diferencia-se no que tange ao limite de capacidade de um sistema. Enquanto o teste de carga pretende descobrir qual a capacidade de uso, processamento, acessos etc. de um sistema, o teste de stress pretende descobrir em que ponto ele já não aguenta mais funcionar. Ou se ele apresenta erros antes de chegar nesse limite anteriormente estabelecido, de modo a que sejam realizadas ações de reparo, melhoria, troca de componentes entre outros.

**1. Teste de Configuração**

Testa se o software funciona no hardware a ser instalado.

**2. Teste de Instalação**

Testa se o software instala como planejado, em diferentes hardwares e sob diferentes condições, como pouco espaço de memória, interrupções de rede, interrupções na instalação etc.

**3. Teste de Integridade**

Testa a resistência do software à falhas (robustez).

**4. Teste de Segurança**

Testa se o sistema e os dados são acessados de maneira segura, apenas pelo autor das ações.

**5. Teste Funcional**

Testa os requisitos funcionais, as funções e os casos de uso. “A aplicação faz o que deveria fazer?”

**6. Teste de Unidade**

Testa um componente isolado ou classe do sistema.

**7. Teste de Integração**

Testa se um ou mais componentes combinados funcionam de maneira satisfatória. Há quem diga que o teste de integração é composto por vários testes de unidade.

**8. Teste de Volume**

Testa o comportamento do sistema operando com o volume “normal” de dados e transações envolvendo o banco de dados durante um longo período de tempo.

**9. Teste de Performance**

O teste de performance se divide em 3 tipos:

         Teste de carga

         Testa o software sob as condições normais de uso. Ex.: tempo de resposta, número de transações por minuto, usuários simultâneos etc.

         Teste de stress (como explicado acima)

         Teste de estabilidade: Testa se o sistema se mantém funcionando de maneira satisfatória após um período de uso.

**10. Teste de Usabilidade**

Teste focado na experiência do usuário, consistência da interface, layout, acesso às funcionalidades etc.

**11. Testes de Caixa Branca e Caixa Preta**

Basicamente, teste de caixa branca envolve o código e o de caixa-preta, não.

**O teste da caixa cinza**

O teste da caixa cinza é resultado da combinação do teste da caixa branca com o teste da caixa preta, embora não aborde todas as características dacaixa branca. Aqui o avaliador fica sabendo dos algoritmos implementados no programa, faz consulta SQL a uma base dados, tem acesso a operações internas, pode manipular arquivos de entrada e saída XML, acessar bancos de dados relacionados à aplicação para verificação de dados, alterar parâmetros etc.

Ou seja, ele avalia as saídas (externas), tendo como contrapartida os processos internos que foram usados para ocasioná-las. Desse modo, caso o programa dê um erro quando ele testa a interface, ele poderá encontrar a causa do problema na parte interna.

**Teste da caixa branca e teste da caixa preta**

Como o teste da caixa cinza é uma derivação do da caixa branca e do da caixa preta, é interessante saber como esses funcionam também. Desse modo, dá para entender qual o objetivo de cada e como isso foi absorvido pelo teste da caixa cinza. Veja mais sobre cada um dos dois logo abaixo:

**Teste da caixa preta** – o avaliador não tem acesso ao código-fonte do programa, avaliando somente a parte funcional, ou externa, dele. Pode-se verificar como o usuário final responderia ao navegar pela interface de um programa, quais os possíveis equívocos que ele cometeria e como o sistema reagiria a eles.

**Teste da caixa branca** – nesse o avaliador tem acesso total ao código fonte do programa, de modo a analisar a parte estrutural dele. É necessário ter um conhecimento técnico mais aprofundado, já que poderá ser preciso utilizar códigos específicos dentro do software.

**12. Teste de Regressão**

Reteste de um sistema ou componente para verificar se alguma modificação recente causou algum efeito indesejado, além de, certificar se o sistema ainda atende os requisitos.

**13. Teste de Manutenção**

Testa se a mudança de ambiente não interferiu no funcionamento do sistema.