

ZingMe Practice

For Building Scalable PHP Website

By Chau Nguyen Nhat Thanh
ZingMe Technical Manager
Web Technical - VNG

Agenda

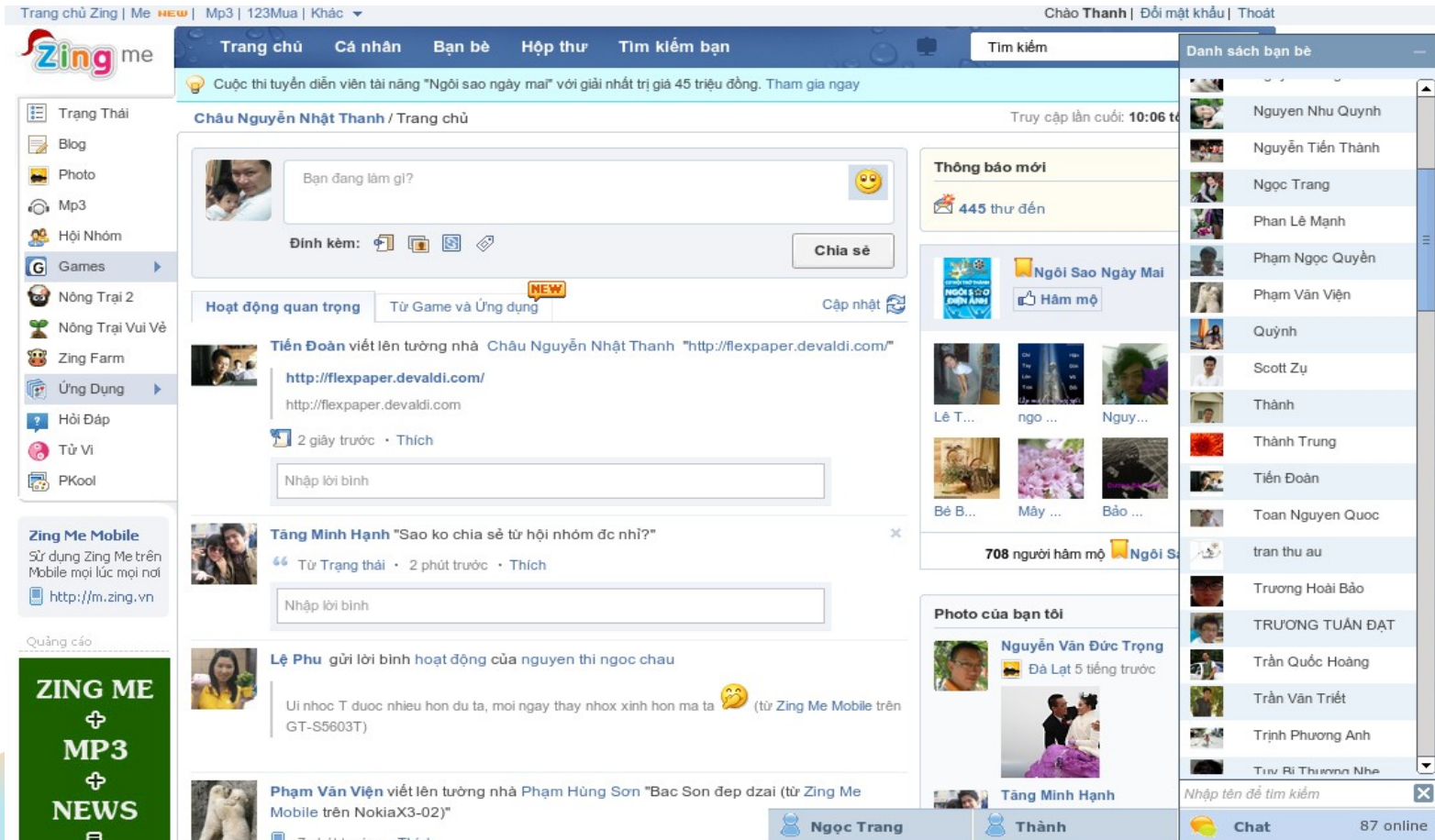
- About ZingMe
- Scaling PHP application
 - Scalability definition
 - Scaling up vs scaling out
 - Load balancing
 - Problem with PHP when using load balancer
 - User session management
 - Problem with code deployment

Agenda(cont.)

- ZingMe practice in building scalable PHP applications
 - Choosing scale out
 - Load balancing model
 - User session management system
 - Code deployment system
- ZingMe more
 - Open social API

About ZingMe

- A social network



The screenshot displays the ZingMe social network interface. At the top, there is a navigation bar with options like 'Trang chủ', 'Cá nhân', 'Bạn bè', 'Hộp thư', and 'Tìm kiếm bạn'. Below this, a user's profile for 'Châu Nguyễn Nhật Thanh / Trang chủ' is shown, including a status update with a photo and a 'Chia sẻ' button. The main content area features several posts from other users, such as 'Tiến Đoàn' and 'Tăng Minh Hạnh', each with a profile picture, text, and a 'Thích' button. On the right side, there is a 'Danh sách bạn bè' (Friends list) with names and profile pictures. The bottom of the interface shows a chat window with the name 'Ngọc Trang' and a 'Chat' button indicating 87 online users.

Zing Me Mobile
Sử dụng Zing Me trên Mobile mọi lúc mọi nơi
<http://m.zing.vn>

Quảng cáo

ZING ME
+
MP3
+
NEWS



About ZingMe

- 25M registered accounts
- 1.7M daily active users
- 5M monthly active users
- > 300 servers (1.2 K cores)
 - 50 DB servers, >40 Memcached servers
 - ~ 100 Web servers
 - 40 servers for Hadoop farm
 - Others (storage, ...)

About ZingMe

- Load balancing: HA proxy, LVS, hardware (?)
- Web server: Nginx, lighttpd, apache, hardware (?)
- Web caching: squid, varnish, hardware (?)
- Caching: Redis, memcached
- Programming language: PHP, C++, Java, python, bash script
- Searching: Solr, lucene
- DB: MySQL , NOSQL
- Log system: Scriber + Hadoop

Scaling PHP application



- Scalability definition
- Scaling up vs scaling out
- Load balancing
- User session management
- Code deployment

Scalability definition

“Scalability is a desirable property of a system, a network, or a process, which indicates its ability to either handle growing amounts of work in a graceful manner or to be enlarged” *from Wikipedia*

“A web application is scalable when is able to manage a growing traffic with additional resources (CPU, RAM) without software changes” *by Jan Burkl, Enrico Zimuel (Zend Technologies)*

Scaling up vs scaling out



- Scaling up (vertically)
 - Add more resources to single node in a system
 - Resources can be software or hardware
 - Make system more powerful by increase node powerful
 - Has an upper limit
 - Hard to implement HA solution
 - Cost: expensive

Scaling up vs scaling out



- Scaling out (horizontally)
 - Add more nodes to a system
 - Make system more powerful by increase number of nodes in the system
 - Maybe has no upper limit
 - Easy to implement HA solution
 - Cost: cheap

Scaling up vs scaling out



VS.



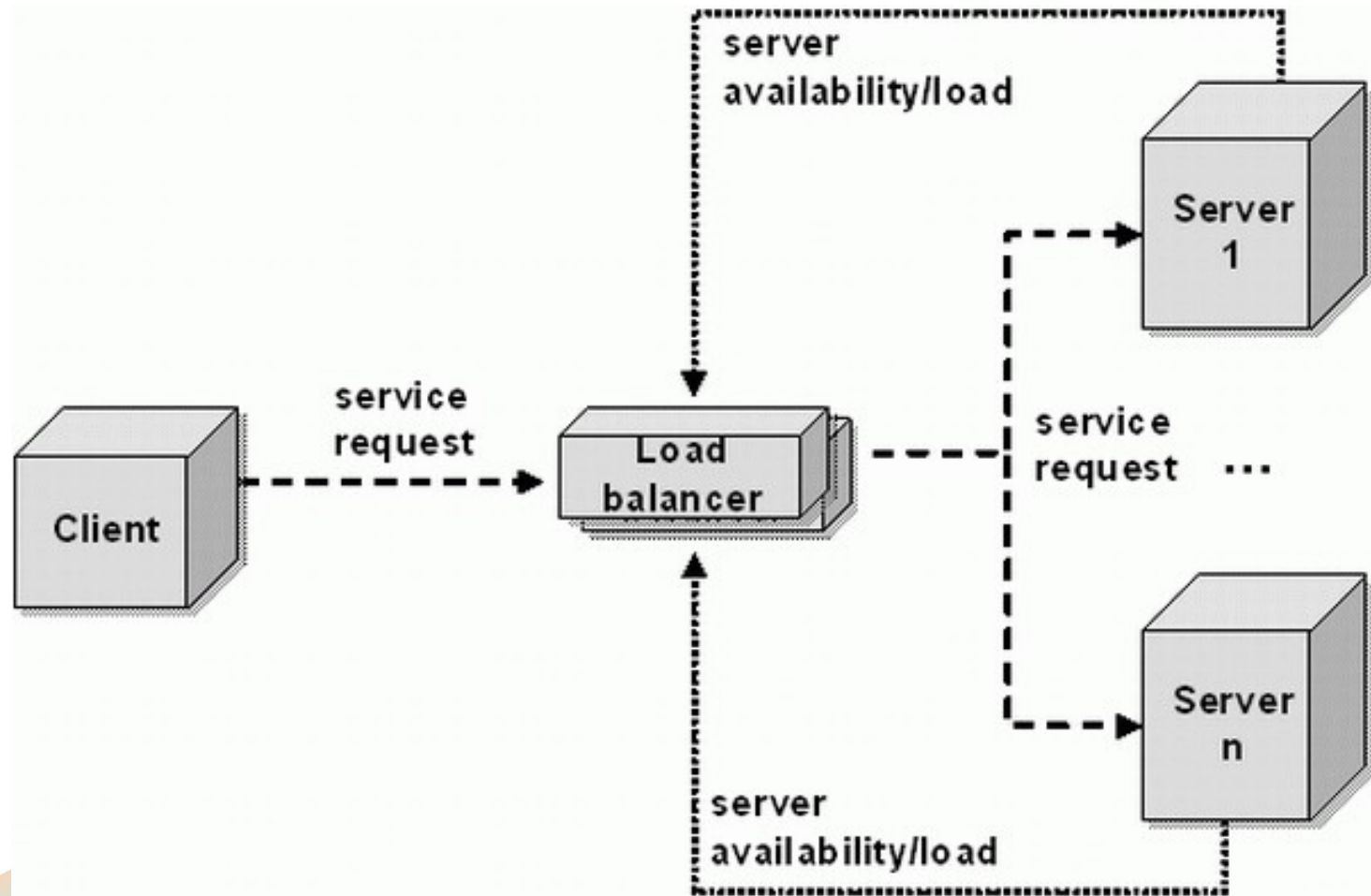
·From slide "How to scale PHP application"
·by Jan Burkl, Enrico Zimuel (Zend Technologies)

Load balancing



- The need of a load balancer
 - Most of big web site choose scale out
 - Need a component for distribute load among the nodes
 - Such component called load balancer

Load balancing



·from "Server load balancing architectures"
·by Gregor Roth, JavaWorld.com, 10/21/08

Load balancing

- How load balancer work?
 - Has 2 interfaces: Back-end, Front-end
 - Front-end receive the request from client
 - Back-end forward request to internal node
 - Back-end receive the response from node
 - Front-end response to client
 - Status monitor for monitoring the back-end interface

Load balancing

- Web application load balancer
 - Work in layer 7
 - Can filter HTTP header and body in both request and response
 - Can be hardware or software
 - F5 BIG-IP, Citrix Netscale, Cisco
 - HA Proxy, Nginx, lighttpd

Problem with PHP when using load balancer

- PHP stores session info into local disk by default
- How to scale session to 2 nodes ?
- Can be scaled if:
 - Load balancer has a method for routing the same user requests into the same nodes (1)
 - Override the default PHP mechanism for storing session of all nodes in the same storage (2)

Problem with PHP when using load balancer

- Method (1)
 - Load balancer must keep state for connection
 - Overhead for routing request
 - Not balancing
- Method (2)
 - Must change software
 - No overhead for routing request
 - More balancer than (1)

PHP session management

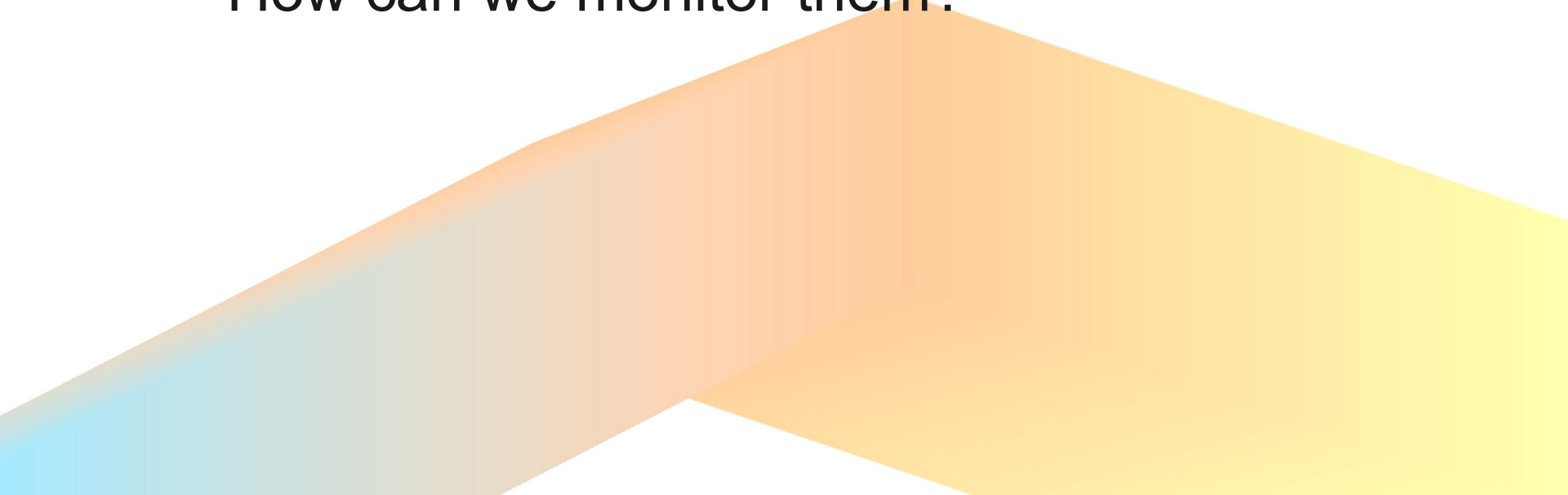


- Choose method (2)
- Centralize storage among web server
 - Storage can be NFS, DB, memcached...
 - All web servers access the same storage
- Override the default behavior of PHP session functions
- Bottleneck can be in storage

PHP session management

- Call `session_set_save_handler(callback $open , callback $close , callback $read , callback $write , callback $destroy , callback $gc)` to override default behavior
- *More infos:*
 - <http://php.net/manual/en/function.session-set-save-handler.php>

Problem with code deployment

- Adding new server to system:
 - What kind of configuration must be changed?
 - Which code will be deployed ? In which server?
 - How can we monitor them?
- 

Problem with PHP code deployment

- What will we change in PHP code ?
 - Separate all configuration to configuration file
 - Do not hard code any configuration
 - Put the code in a central server like SVN
 - Use scripts to pull all code to production server
 - Build profiler system for monitor the code

ZingMe practice for building scalable web site



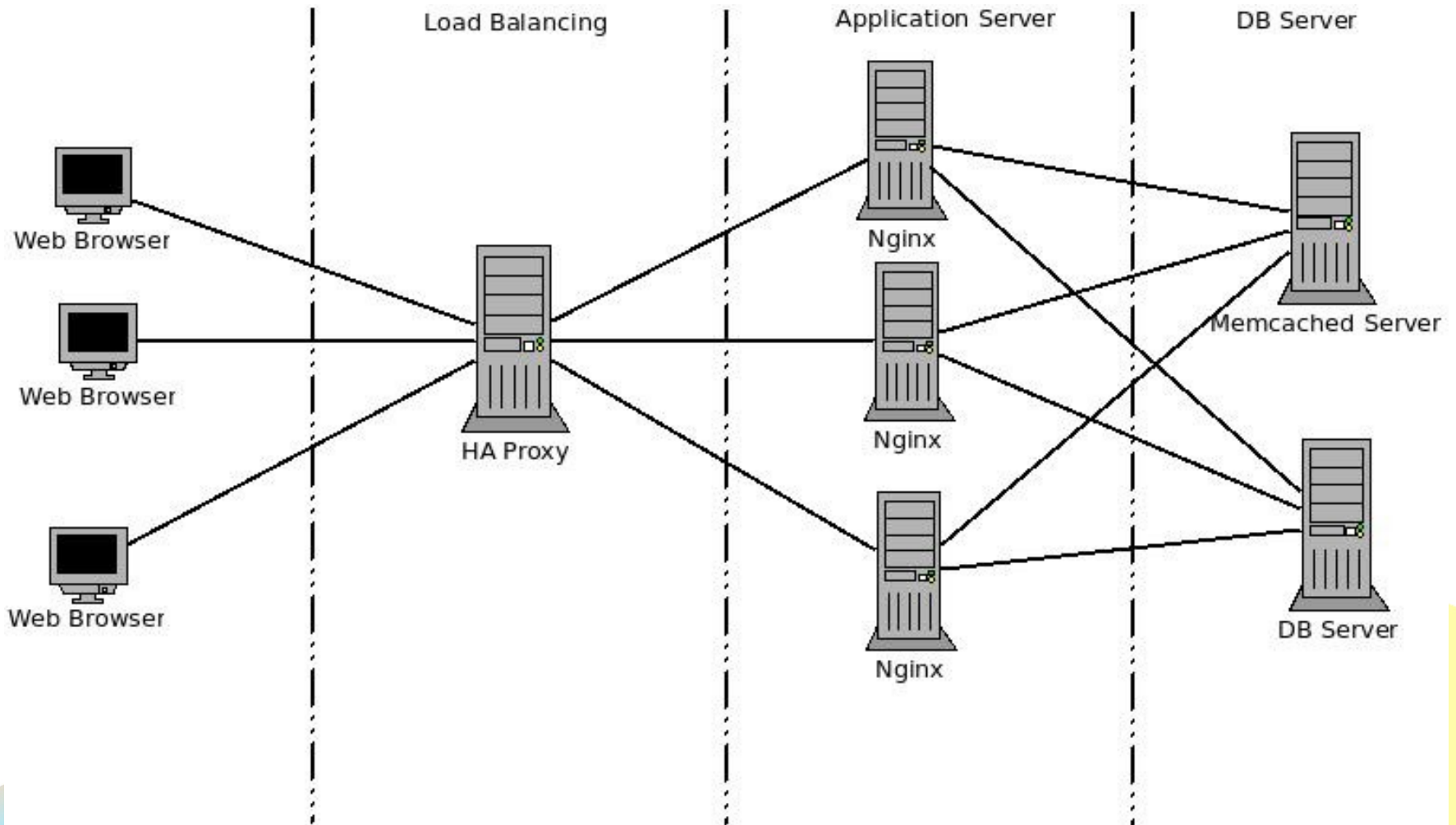
- Choose Scale out
- Use HA proxy for load balancing
- Use Memcached for session storage
- Simple code deployment system base on SVN and bash script

ZingMe practice for building scalable web site



- Divide into small services
- Use commodity hardwares
- Use open source software
- Server has the same environment
- Add new server < 5mins

ZingMe load balancing model



ZingMe load balancing model

- Use HA proxy
 - HTTP based proxy
 - High availability
 - Use epoll
- Use content switching to divide the service into group
- Define load balancing factors based on the power of the back-end server

ZingMe load balancing model

Normal traffic



ZingMe load balancing model

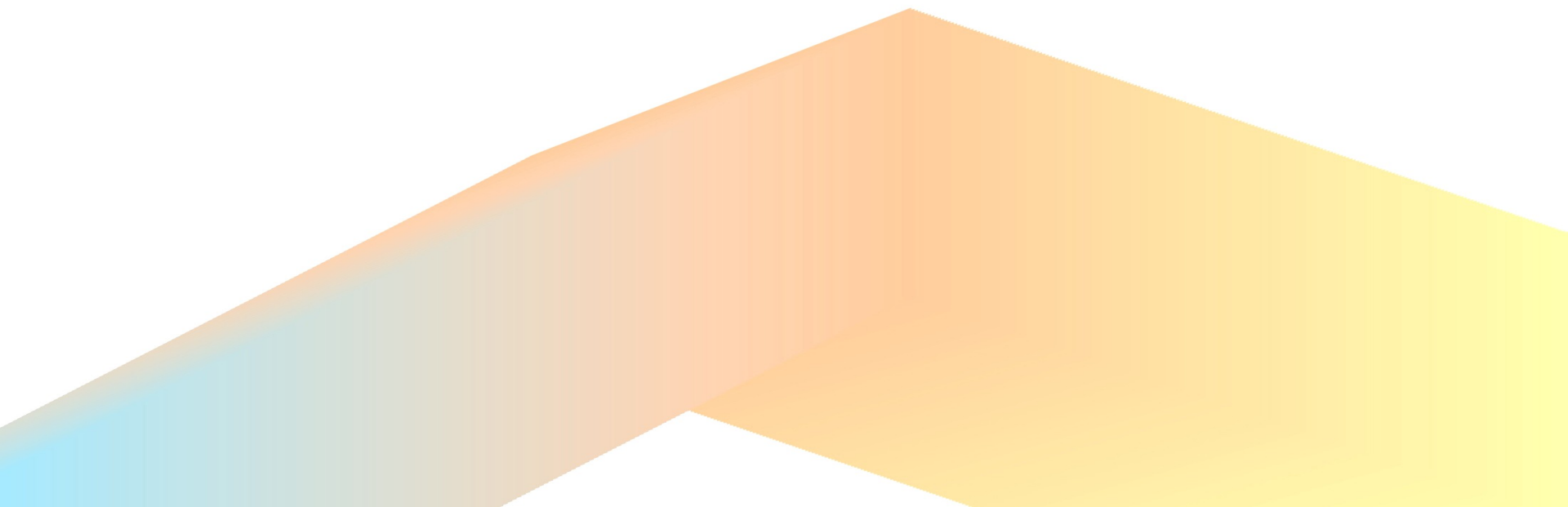
ZingMe traffic



ZingMe user session management



- Use memcached for high performance
- Override Zend_Session
- 3K hits/s



Why we choose memcache?



- L1 cache reference 0.5 ns
- Branch mispredict 5 ns
- L2 cache reference 7 ns
- Mutex lock/unlock 25 ns
- **Main memory reference 100 ns**
- Compress 1K bytes with Zippy 3,000 ns
- Send 2K bytes over 1 Gbps network 20,000 ns
- Read 1 MB sequentially from memory 250,000 ns
- Round trip within same datacenter 500,000 ns
- **Disk seek 10,000,000 ns**
- Read 1 MB sequentially from disk 20,000,000 ns
- Send packet CA->Netherlands->CA 150,000,000 ns

From Jeff Dean - google.com

ZingMe code deployment



- Developer upload code to SVN
- SO runs scripts to deploy code to production server
- Has staging server for real environment test
- Dev cannot touch to production environment
- Using environment parameter
 - Nginx: fastcgi_param
 - Apache: setenv

ZingMe more



- Standard latency for server side code processing < 200ms
- Live demos Profiler system
- Open social API updated infos
 - Sandbox for developer
 - Will be open soon

ZingMe more



- Optimize node before scale it up
 - Performance C++ > Java > PHP
 - Why PHP ?
 - Easy to implement business
 - Easy to deploy
 - Why C++?
 - Performance
 - Can we mix PHP and C++?
 - Next session will answer

Q&A



THANK
YOU!

Contact info:

Chau Nguyen Nhat Thanh
thanhcnn@vinagame.com.vn
me.zing.vn/thanhcnn2000