Technical Architecture

From Matchi Wiki

Contents

Infrastructure Architecture

Server Infrastructure

The current server design consists of a web server, database server, test server and an application server that runs processes that assist the other servers. This installation is designed to be as flexible and scalable as possible and to easily correct pinch-points so that they can easily be corrected.

Environments

These servers only show the production environment. All other environments (DEV, TEST, etc.) are hosted in local environments on the test server.

Content Delivery Network

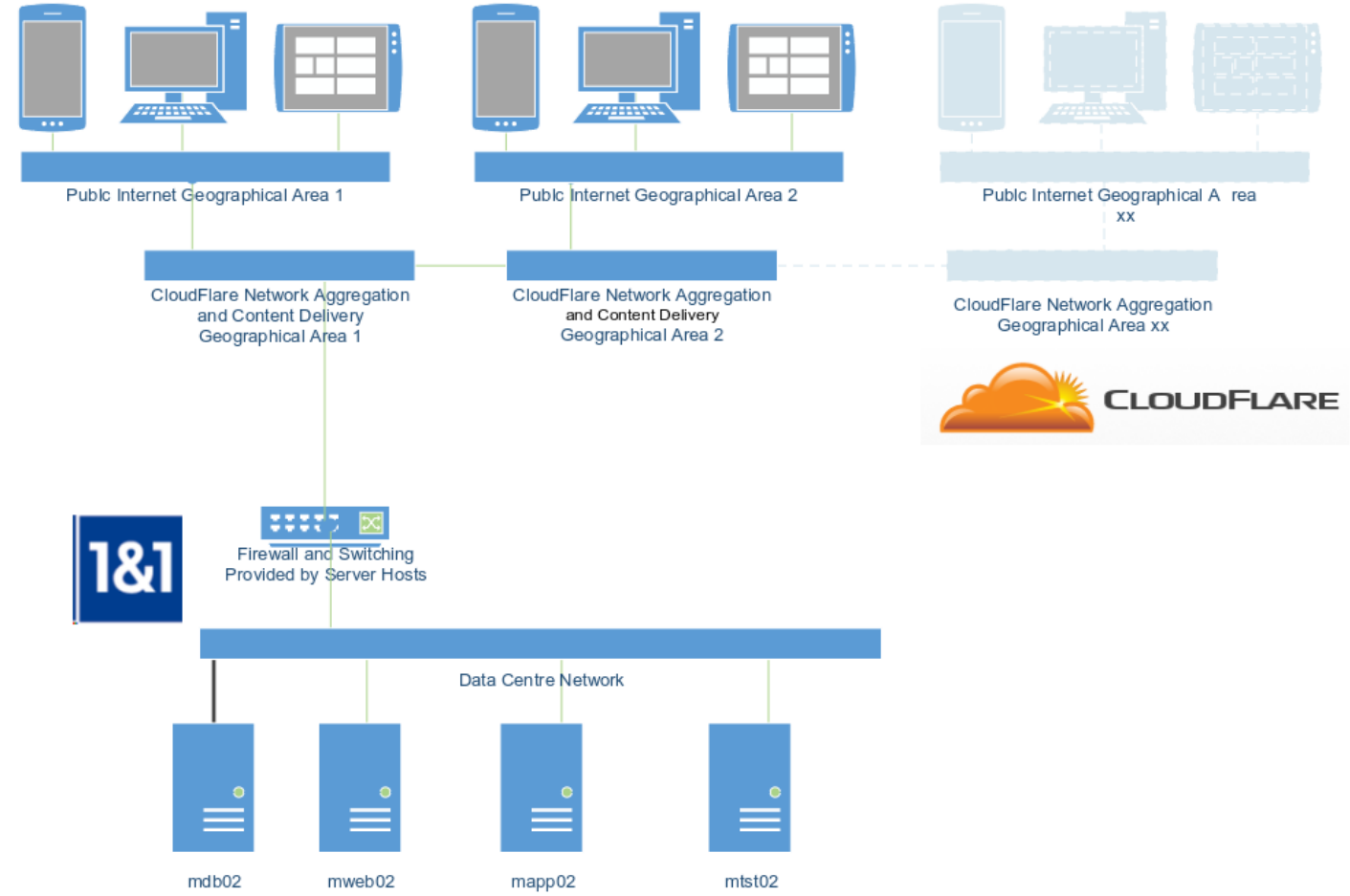
The Matchi production environment is front-ended with the Cloudflare Content Delivery Network (CDN), that distributes the static content over many geographically-dispersed data centers (31 at last count). Ths CDN also offers additional protection from web-based attacks.

Load Balancing

There is currently no explicit load-balancing on any of the servers. It possible to add the load balancing either through software or through a load-balancing switch in front of the servers when then server resource demand increases.

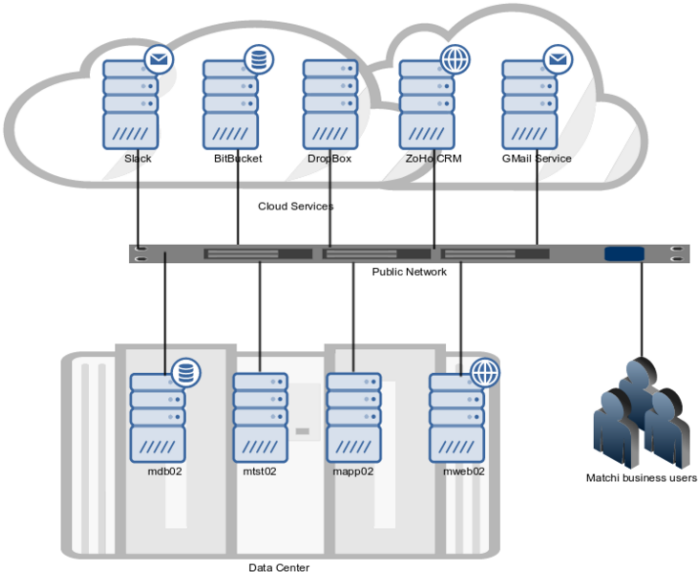
=Performance Overview and Considerations

Since the transaction volume - and therefore the dynamically-generated content - is very low compared to the static content, it is unlikely that load balancing will be required, as long as any dynamic content is generated as quick as possible.



Cloud Infrastructure

Some business services are provided as cloud-hosted services and are directly accessed from the servers and the Matchi business users via the public Internet.



Operating Systems

Server Operating Systems

Servers mdb02 runs CENTOS 6.7 (a clone of Red Hat 6) with 8GB of virtual RAM and 4 virtual CPUs. The virtual hosting environment is the OpenVZ container. Allocated SAN storage is 200GB.

Server mtst02, mweb02 and mapp02 run CENTOS 7.2 (a clone of Red Hat 7) with 16GB of physica RAM and a 6-code physical CPU. Local, physical storage is 1TB on a RAID-1 configuration.

Client Operating Systems

This solution is client operating-system agnostic, which means that the service that if provides can be delivered to machines that a any of the currently-used operating systems: Linux, Android, Apple, Solaris, Haiku, Windows.

Service delivery is via a W3C-compliant web browser. Operational work is performed via a W3C-compliant web browser, a console over an public/private key-encrypted connection, or SFTP.

Server Management, Monitoring and Support

Overview

The solution consists of 3 virtual servers in the same data center provided by 1and1. Neither Affinity (sharing virtual instances of the same back plane of the blade server) nor Anti-affinity (the opposite effect) can be assured under the terms of the hosting contract.

A PLESK interface exists for each server, although this interface makes simplistic assumptions about the configuration of the servers. As a rule, the PLESK tool is ignored in preference to using the command line via the secure shell (SSH) to manage and monitor the servers, as detailed in this document. The PLESK interface is useful for backing up or restoring the entire VM and for configuration management beyond the actual servers (e.g. setting firewall policies on the routers instead of the servers themselves).

Support Contacts

The servers are currently hosted with 1and1.

The contact details for 24-hour support are:

- Server-specific support: +44 333 336 5691
- Account support: +44 333 336 5780
- Email support: support@1and1.co.uk

Note that they are not very quick at answering phone calls. When you eventually get to speak to them, remind them of that.

Server Monitoriting Service

TO DO

Physical Architecture

The physical architecture in the current design iteration consists of these servers:

WEB SERVER #2

- Alias: mweb02
- Hosting Provider: 1and1.co.uk
- Hosting Support: (UK) 0333 336 5691
- Hosting Admin site: https://admin.1and1.co.uk
- Account Number: 469697884
- Password: M*
- Contract Id: 61489336
- Contract Expiry: 30SEP2017
- Contract Name: 1&1 Power Deal Server XL6
- Contract Features:
  - CPU: AMD Hexa-Core
  - Cores: 6
  - Clock: 2.8GHz, 3.3GHz TurboCode
  - Memory: 16GB DDR3 ECC
  - Storage: 1000GB x 2, mechanical
  - Redundancy: RAID1 Software
  - Traffic: Unlimited
  - Bandwidth: 100MBit/s
- IPv4: 87.106.243.116
- IPv6: 2001:8d8:993:5c00:0:0:36:ed67
- IPv6 subnet: 2001:8d8:993:5c00::/56
- Server name: s19550136.onlinehome-server.info
- Initial password: ******
- Support PIN: 14112013
- Technical domain: s649984010.websitehome.co.uk
- MX domain: s649984010.websitehome.co.uk
- Plesk Control Panel: https://87.106.243.116:8443 sysuser: admin/M*
- Virtuozzo Power Panel: None
- Serial Console: sercon.onlinehome-server.info
- Serial Console User: u86643325
- Serial Console Password: M*
- Operating system : CentOS 7 (64-bit)

APP SERVER #2

- Alias: mapp02
- Hosting Provider: 1and1.co.uk
- Hosting Support: (UK) 0333 336 5691
- Hosting Admin site: https://admin.1and1.co.uk
- Account Number: 469697884
- Password: M*
- Contract Id: 61489257
- Contract Expiry: 10OCT2017
- Contract Name: 1&1 Power Deal Server XL6
- Contract Features:
  - CPU: AMD Hexa-Core
  - Cores: 6
  - Clock: 2.8GHz, 3.3GHz TurboCode
  - Memory: 16GB DDR3 ECC
  - Storage: 1000GB x 2, mechanical
  - Redundancy: RAID1 Software
  - Traffic: Unlimited
  - Bandwidth: 100MBit/s
- IPv4: 87.106.145.156
- IPv6: 2001:8d8:8bc:6200:0:0:a8:27ff
- IPv6 subnet: 2001:8d8:8bc:6200::/56
- Server name: s19578458
- Server public URL: s19578458.onlinehome-server.info
- Initial password: ******
- Support PIN: 14112013
- Technical domain: s651646577.websitehome.co.uk
- MX domain: s651646577.websitehome.co.uk
- Plesk Control Panel: https://87.106.145.156:8443 sysuser: admin/M*
- Virtuozzo Power Panel: None
- Serial Console: sercon.onlinehome-server.info
- Serial Console User: u86643439
- Serial Console Password: M*
- Operating system : CentOS 7 (64-bit)

TEST SERVER #2

- Alias: mtst02
- Hosting Provider: 1and1.co.uk
- Hosting Support: (UK) 0333 336 5691
- Hosting Admin site: https://admin.1and1.co.uk
- Account Number: 469697884
- Password: M*
- Contract Id: 61489257
- Contract Expiry: 10OCT2017
- Contract Name: 1&1 Power Deal Server XL6
- Contract Features:
  - CPU: AMD Hexa-Core
  - Cores: 6
  - Clock: 2.8GHz, 3.3GHz TurboCode
  - Memory: 16GB DDR3 ECC
  - Storage: 1000GB x 2, mechanical
  - Redundancy: RAID1 Software
  - Traffic: Unlimited
  - Bandwidth: 100MBit/s
- IPv4: 217.160.206.97
- IPv6: 2001:8d8:96c:c200:0:0:28:c8d3
- IPv6 subnet: 2001:8d8:96c:c200::/56
- Server name: s19554880
- Server Public URL: s19554880.onlinehome-server.info
- Server Initial password: ******
- Support PIN: 14112013
- Technical domain: s651646589.websitehome.co.uk
- MX domain: s651646589.websitehome.co.uk
- Plesk Control Panel: https://217.160.206.97:8443 sysuser: admin/M*
- Virtuozzo Power Panel: None
- Serial Console: sercon.onlinehome-server.info
- Serial Console User: u86643439
- Serial Console Password: M*
- Operating system : CentOS 7 (64-bit)

DATABASE SERVER #2

- Alias: mtst02
- Hosting Provider: 1and1.co.uk
- Hosting Support: (UK) 0333 336 5691
- Hosting Admin site: https://admin.1and1.co.uk
- Account Number: 469697884
- Password: M*

- Contract Id: 57551477
- Contract Expiry: 05NOV2016
- Contract Name: 1&1 Virtual Server XL Linux
- Contract Features:
  - vCPU: 4
  - Clock: 2.8GHz
  - vMemory: 6GB guaranteed
  - Storage: 300GB SAN
  - Redundancy: SAN
  - Traffic: Unlimited
  - Bandwidth: 100MBit/s
- IPv4: 212.227.255.146
- IPv6: N/A
- IPv6 subnet: N/A
- Server name: s614338666
- Server Public URL: s614338666.websitehome.co.uk
- Server Initial password: ******
- Support PIN: 14112013
- Technical domain: s651646589.websitehome.co.uk
- MX domain: s651646589.websitehome.co.uk
- Plesk Control Panel: https://212.227.255.146:8443 sysuser: admin/M*
- Virtuozzo Power Panel: https://212.227.255.146:4643/vz/cp: root/2EtiH7tP
- Serial Console: N/A
- Serial Console User: N/A
- Serial Console Password: N/A
- Operating system : CentOS 6 minimal system (64-bit)

Network Architecture

The network architecture is taken care of by the hosting providers, who provide a basic, first-line network intrusion defense, network routing and DNS services.

Some pertinent facts:

- The network both IP Version 4 and 6.
- The servers are not on the same sub-net
- The bandwidth between all servers is 100 MB/s
- The bandwidth between servers is unlimited
- The bandwidth to the public network is unlimited
- The current hosting vendor does not support virtual networks or trunking
- The current hosting vendor's network intrusion defenses are variable, insufficient and should not be relied on.

Storage Architecture

The current requirement for file storage on the solution is relatively low compared to the number sessions that the solution has to support. Large files such as members' videos are provisionally hosted by third-party providers such as YouTube and Vimeo.

Virtual Server Storage Architecture

Storage Partitioning:

- Data / Application File Storage: Each server has a single 200GByte partition of tier-1 SAN storage on which the entire root partition is mounted. This includes Static web content (on the web server) that is in an easily-identifiable, dedicated root directory that is associated with the application build where appropriate. This directory is ready for sharing with a Content Delivery Network (CDN).
- Operating System storage: The Operating System is hosted on the actual VM system.
- Swap partition: There is no swap partition on the virtual machine since swap space is in reality shared memory on the VM system.

Storage Size: Virtual servers have 200,0000 inodes available for storage use. This limited number of inodes is frequently a problem since many small files can quickly deplete this count. To check inodes and storage consumption:

```
$ df -i .
Filesystem     Inodes  IUsed  IFree IUse% Mounted on
/dev/vzfs      750000 508807 241193   68% /
$ df -m .
Filesystem     1M-blocks  Used Available Use% Mounted on
/dev/vzfs         195313 19935    175378  11% /
```

So even though only 11% of storage is consumed, 68% of the available inodes are consumed. Linux systems on real / non-virtual servers start off with a default of 3,000,000 inodes and are extendible. Because these are virtual servers, the number of inodes is not configurable.

Physical Server Storage Architecture

Storage Partitioning:

Physical servers us Logical Volume Management (LVM)

The basic volume partitioning schema is as follows:

```
$ df -m
Filesystem           1M-blocks  Used Available Use% Mounted on
/dev/md1                  4000   312      3688   8% /
devtmpfs                  7973     0      7973   0% /dev
tmpfs                     7984     0      7984   0% /dev/shm
tmpfs                     7984    89      7895   2% /run
tmpfs                     7984     0      7984   0% /sys/fs/cgroup
/dev/mapper/vg00-usr      4912  3214      1426  70% /usr
none                      7984     0      7984   0% /tmp
/dev/mapper/vg00-var    408085  8326    383094   3% /var
/dev/mapper/vg00-home    55309  6023     46956  12% /home
tmpfs                     1597     0      1597   0% /run/user/10002
```

The volumes can dynamically be created, removed and resized as required, depending on the use of the server, and this is described in the specific server build process instructions.

Storage Size:

The total size of the available storage on a physical server is 1000 GBytes.

The total number of inodes available is 43 million. Check this so:

```
$ df -i | tail  | awk '{print $2}'  | paste -d+ -s  | bc
42995901
```

Resilience Architecture

Virtual Servers

Virtual machines implicitly offer a high degree of resilience and redundancy. Storage is SAN-based and is redundantly held in two geographically separated data centers.

Note
It has been observed that these virtual machines suffer acutely from the *Noisy Neighbour* phenomenon, where adjacent virtual machines on the same physical VM server from other clients consume the (over-allocated) shared resources of that server, and so cause performance issues on Matchi's servers. Although an obvious remedy for the *Noisy Neighbour* phenomenon is to host the servers on dedicated machines (a.k.a. 'tin'), the benefits of resilience offered by VMs would be lost and other approaches (of which there are many available) would need to be applied.

Physical Servers

The current design does not offer any resilience for physical server, and storage is locally based on direct-attached storage devices (DASD).

Recovery depends on the successful restoration of back-ups and re-purposing of an existing server.

Resilience through the Content Delivery Network

Resilience is further enhanced with the introduction of a Content Delivery Network (CDN). We use the Cloudflare CDN (https://www.cloudflare.com) Pro subscription for the production site, which provides local caching of all static data.

Scalability and Content Delivery Architecture

Load balancing

There is currently no load-balancing between servers on any of the environments.

Content Delivery Network

The CloudFlare service is a Content Delivery Network (CDN) that provides distributed, geographical storage and replication of static content on their own infrastructure. It currently has points of presence in 15 global cities countries and this number is constantly growing.

Security Architecture

Local security

The servers have been locally secured with the following precautions:

- Access is restricted to the necessary ports only.
- All remote sessions are over the secure shell protocol (SSH2)
- Authentication to the servers is through Public/Private Key pairs though an unpublicized account name
- Remote root access is disabled.
- The SSH protocol is 1024-bit asymmetrically encrypted.
- The use of password-based access is limited to a single server-maintenance account and will eventually be revoked.

Third-party security

- CloudFlare provides an additional layer of security by protecting against denial of service attacks (DDoS)
- CloudFlare is our SSL certificate provider for HTTPS transport.
- Payment systems will use separate SSL certificated that are provided by the various payment providers.

Notes

Red Hat SELINUX (Secure Linux) version is not installed as it is neither available nor proven on the current virtual server environment.

Data Architecture

Classes of Data Overview

Text data

All text-based content is held on a RDBMS (relational database service). The RDBMS is hosted on a dedicated Database server. Typically, this includes:

- User contents
- Website article content
- Application support data
- Configuration data

Application Graphical data

Application Graphics include:

- Graphic web content used by the application itself
- Images used for web articles
- Images used for public emails

Application Graphics are held in the website's /images directory. This directory is generally readable by the public, however, to prevent the wholesale "slurping up" of all the image data in this directory, many of the sub-directories are set to only display the content if the referring application that wishes to display the content is on the same server. This is achieved by setting a .htaccess file in each directory that needs to be protected, which contains:

```
RewriteEngine on
RewriteCond %{HTTP_USER_AGENT} ^facebookexternalhit.*$ [OR]
RewriteCond %{HTTP_REFERER} !^https*://(.*\.)?matchi.biz/.*$ [NC]
RewriteRule \.(gif|png|jpg|pdf|doc|docx|txt|swf)$ https://matchi.biz/public/errors/NoLeeching.jpg
```

Since this is static data, it is an ideal candidate for cacheing and is cached using the CloudFlare cacheing service.

User Attachment Files

Files that members upload are held in a hashed directory tree that is directly accessible from the web server. Attachments typically include

- Diagrams and images in all known image formats
- Office documents, presentations, spreadsheets, PDFs
- Videos

This data is only readable by signed-in members who have the required access. Since this is static data, it is an ideal candidate for cacheing and is cached using the CloudFlare cacheing service.

Messaging Architecture (still under design)

NOTE: This is still design work under construction.

The eventual introduction of a messaging component into the Matchi Solution Architecture was planned to coincide once the application design and business design has reached an acceptable level of stability.

Reasons for using messaging as an Integration Strategy

Historically, some events have been triggered by custom-written finite state machines that intentionally scan the states of records and react accordingly. Other events, such as file-based events, are monitored using the operating system's INOTIFY event notifyer. Both of these approaches are:

- Computationally wasteful and often do nothing useful
- Error prone and require restart-processes
- The INOTIFY event monitor has a limit on how many files it can physically monitor on a virtual machine. There is already one INOTIFY monitoring process used for detecting malware (a.k.a. maldet) which must take priority.

A far more scalable and distributable approach is to use message queueing to a message broker, that passes messages onto various message consumers where events are asynchronously processed. Asynchronous processing is preferable, as mostly fits our messaging use-case. Here is what we get with a Messaging-oriented middleware in the Matchi solution architecture:

- Distribution / Offloading of jobs onto other servers
- Common Integration protocol between processes and to exchange data
- Scaling of the application
- Queueing of jobs (asynchronous as well as synchronous)
- Scheduling of regular-occurring jobs
- System Event and Error management

Choice of Message Queue Provider

RabbitMQ (http://www.rabbitmq.com) was chosen for the message-oriented middleware, because it is:

- Widely supported by the large technology vendors (Red Hat, Oracle, Cisco, Novell, et al)
- RabbitMQ implements most of the AMQP (http://en.wikipedia.org/wiki/Advanced_Message_Queuing_Protoco) (Advanced Message Queuing Protocol) message protocol. AMQP is an open standard wire-level/binary protocol. Since many external services support this protocol, these services are available for future integration into the Matchi application. Likewise, the Matchi Application can be treated as a service and integrate into other applications.
- It is a scalable message broker, that performs message exchanging, message routing and message queuing and dispatching.
- Direct Interoperability is already available in a number of programming languages like PHP, Perl, Python, C/C++/C#, Java, Erlang, and many more through libraries/plugins/packages. This is helpful when writing message Producers and message Consumers.
- RabbitMQ offers a large support community

Message Architecture Topologies

Available Choices

- Direct: 1 message consumer per exchange
- Fan-out: multiple message consumers per exchange, all consuming the same message
- Topic: multiple message consumers per exchange, but messages are routed according to predetermined logic to each consumer

A mix of these topologies are used in the Matchi Application.

Retrieved from "http://wiki.matchi.info/index.php?title=Technical_Architecture&oldid=1734"

Category: Pages with syntax highlighting errors

- This page was last modified on 29 January 2017, at 21:59.
- Content is available under Creative Commons Attribution unless otherwise noted.