Monitoring von Linux Servern und Services

Thema dieses Talks

- Was ist Monitoring?
- Warum braucht man Monitoring?
- Was soll man überwachen?
- Effektives Monitoring ist keine einfache Aufgabe
- Wie man effektives Monitoring konfiguriert (Am Beispiel Nagios/Icinga)
- Ausblick auf komplexe Infrastrukturen

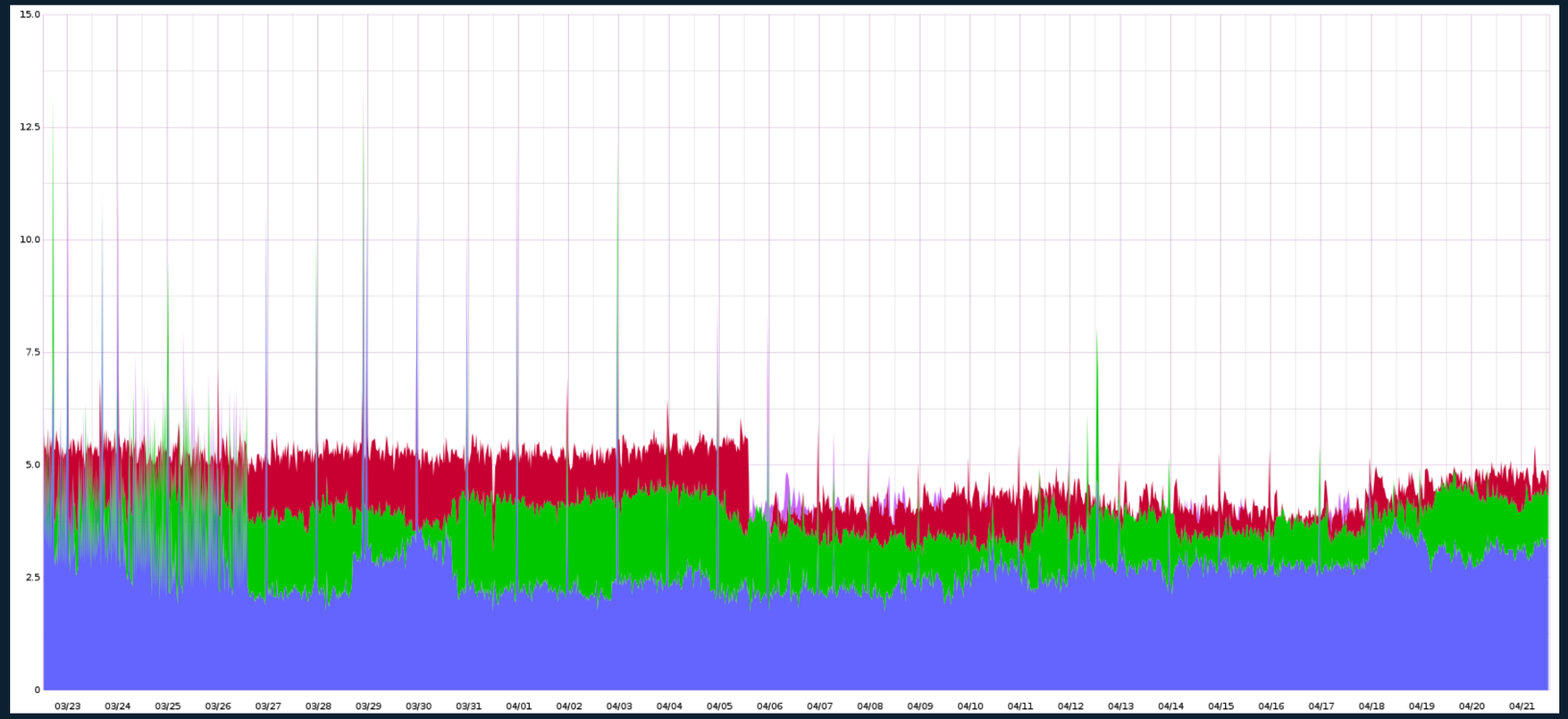
Was ist Monitoring?

- Auswerten von Daten
  - Service Checks überprüfen ob Grenzwerte überschritten oder Dienste nicht erreichbar sind
  - Admins werden bei einem Problem sofort informiert
  - Idealfall: Notification bereits vor einem Ausfall
- Aufzeichnen von Daten
  - Zur Analyse nach oder während eines Ausfalls
  - Um zukünftige Probleme vorherzusehen

Was ist Monitoring?

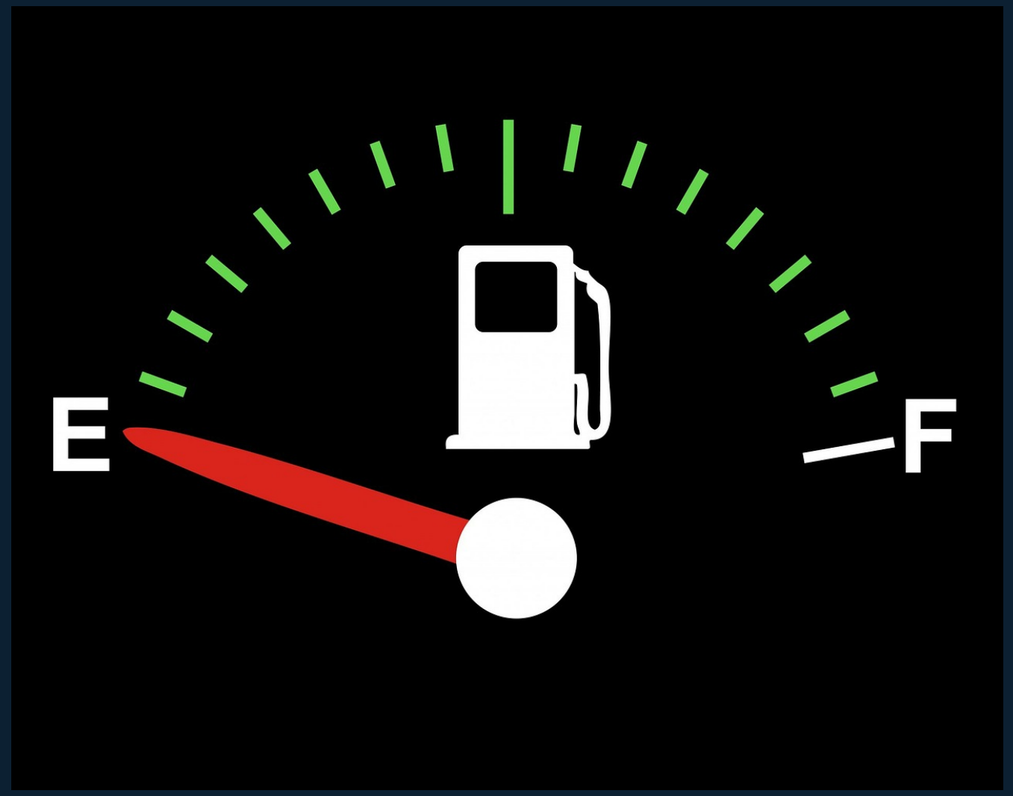
check_by_ssh_cron_active	OK	21-04-2017 13:23:27	11d 1h 12m 49s	1/3	FILE_AGE OK: /tmp/icinga-cron.log is 26 seconds old and 0 bytes
check_by_ssh_disk	OK	21-04-2017 13:18:17	58d 20h 31m 45s	1/3	DISK OK
check_by_ssh_huge_pages_free	OK	21-04-2017 13:22:20	11d 1h 8m 18s	1/3	OK - 29% Huge_Pages free
check_by_ssh_ipmi_sensor	OK	21-04-2017 13:20:34	11d 1h 12m 59s	1/3	IPMI Status: OK
check_by_ssh_last_puppet_run	OK	21-04-2017 13:14:19	58d 20h 24m 12s	1/3	FILE_AGE OK: /var/lib/puppet/state/last_run_summary.yaml is 323 seconds old and 925 bytes
check_by_ssh_load	OK	21-04-2017 13:24:17	11d 1h 13m 13s	1/3	OK - load averages are at 2.98, 3.72, 3.84
check_by_ssh_mailq	OK	21-04-2017 13:17:55	41d 22h 58m 19s	1/3	OK: mailq (0) is below threshold (25/50)
check_by_ssh_ntp_time	OK	21-04-2017 13:22:56	5d 5h 2m 40s	1/3	NTP OK: Offset 0.001296758652 secs
check_by_ssh_ntpd	OK	21-04-2017 13:21:20	11d 1h 13m 15s	1/3	PROCS OK: 3 processes with command name 'ntpd', args 'ntpd'
check_by_ssh_procs_state	OK	21-04-2017 13:07:44	58d 20h 29m 14s	1/3	PROCS OK: 0 processes with STATE = X,Z
check_by_ssh_swap	OK	21-04-2017 13:20:14	41d 22h 59m 13s	1/3	SWAP OK - 64% free (4845 MB out of 7628 MB)
check_by_ssh_users	OK	21-04-2017 12:55:48	58d 20h 29m 11s	1/3	USERS OK - 1 users currently logged in
check_ssh	OK	21-04-2017 13:22:07	41d 22h 59m 4s	1/3	SSH OK - OpenSSH_7.2p2 Ubuntu-4ubuntu2.1 (protocol 2.0)

Was ist Monitoring?



Warum braucht man Monitoring?

So wünscht man sich Monitoring



Was soll ins Monitoring?

Hostgebunden

- Network Availability
- Load
- Memory (Swap)
- Disk

Servicegebunden

- HTTP
- SQL
- TCP
- Unix-Socket

Effektives Monitoring: Keine einfache Aufgabe

- Monitoring Schnittstellen (Metriken die geprüft werden)
- Geeignete Grenzwerte für jedes System
- Vermeidung von False Positives
- Abdeckung Möglichst aller Details
- Redundantes Monitoring
- Erschwerung aller vorherigen Punkte bei wachsender Infrastruktur

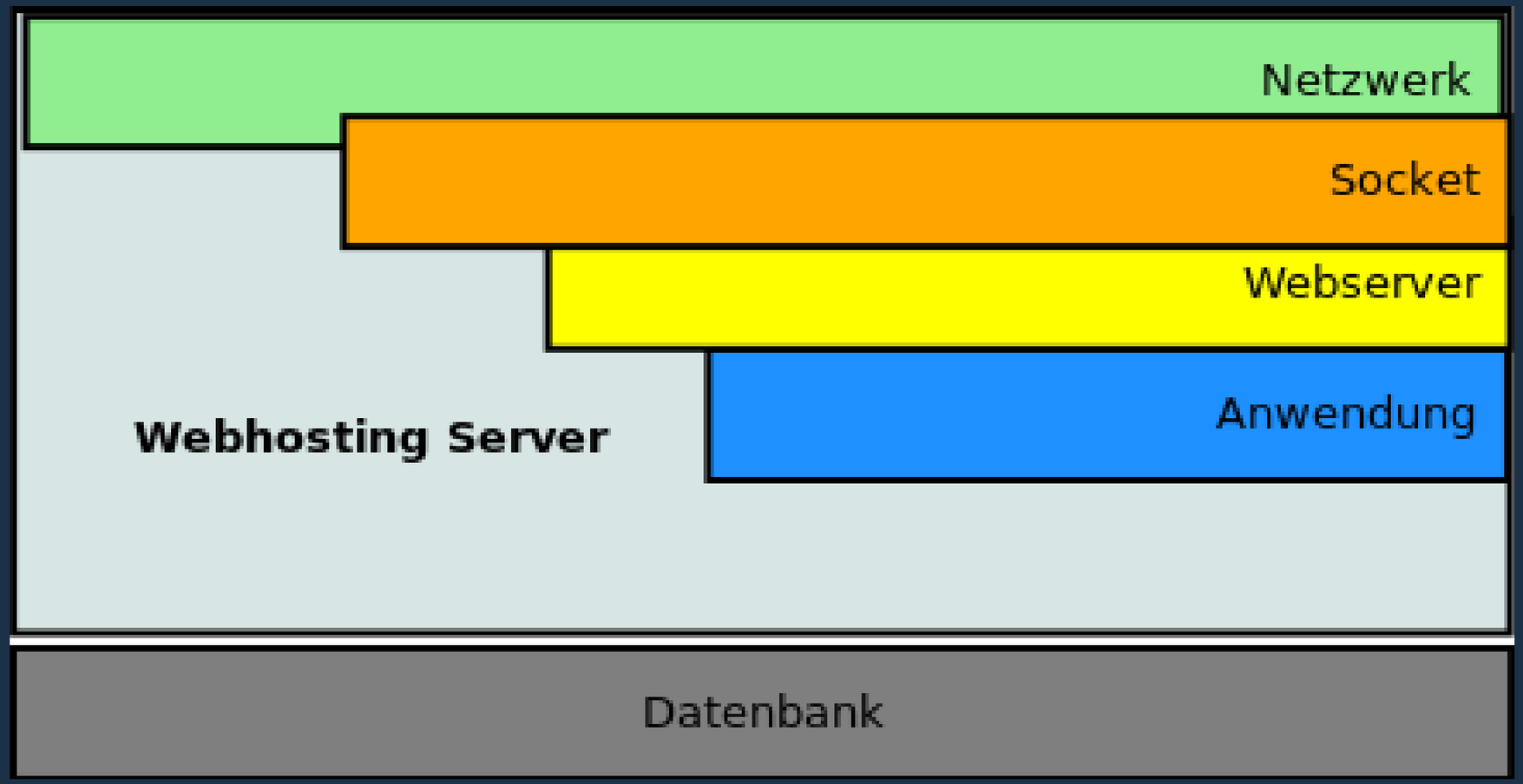
Was macht einen guten Service Check aus?

- Prüfen auf korrekte Funktionalität
  - Antwort eines Dienstes bedeutet nicht..
    - dass dieser eine korrekte Antwort liefert
    - dass die Antwort in akzeptabler Zeit ausgeliefert wird
- Benachrichtigung zum richtigen Zeitpunkt
  - Nicht bevor ein Problem droht oder eines vorhanden ist
  - Rechtzeitig um das Problem schnell zu beheben
  - Idealfall: Schon bevor es zu Einschränkungen kommt

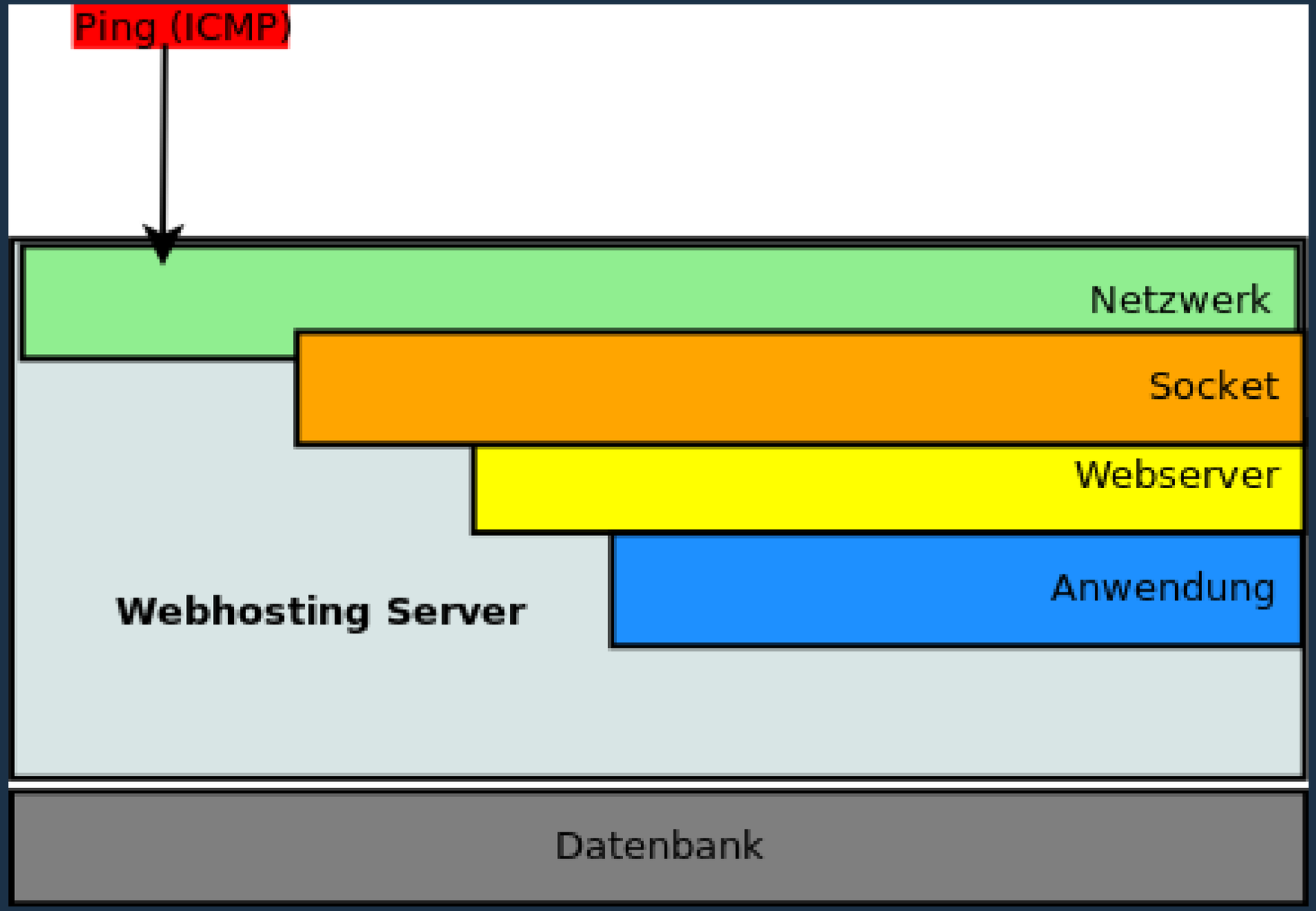
Service Check Beispiel

Ziel: Prüfen ob eine Webseite erreichbar ist

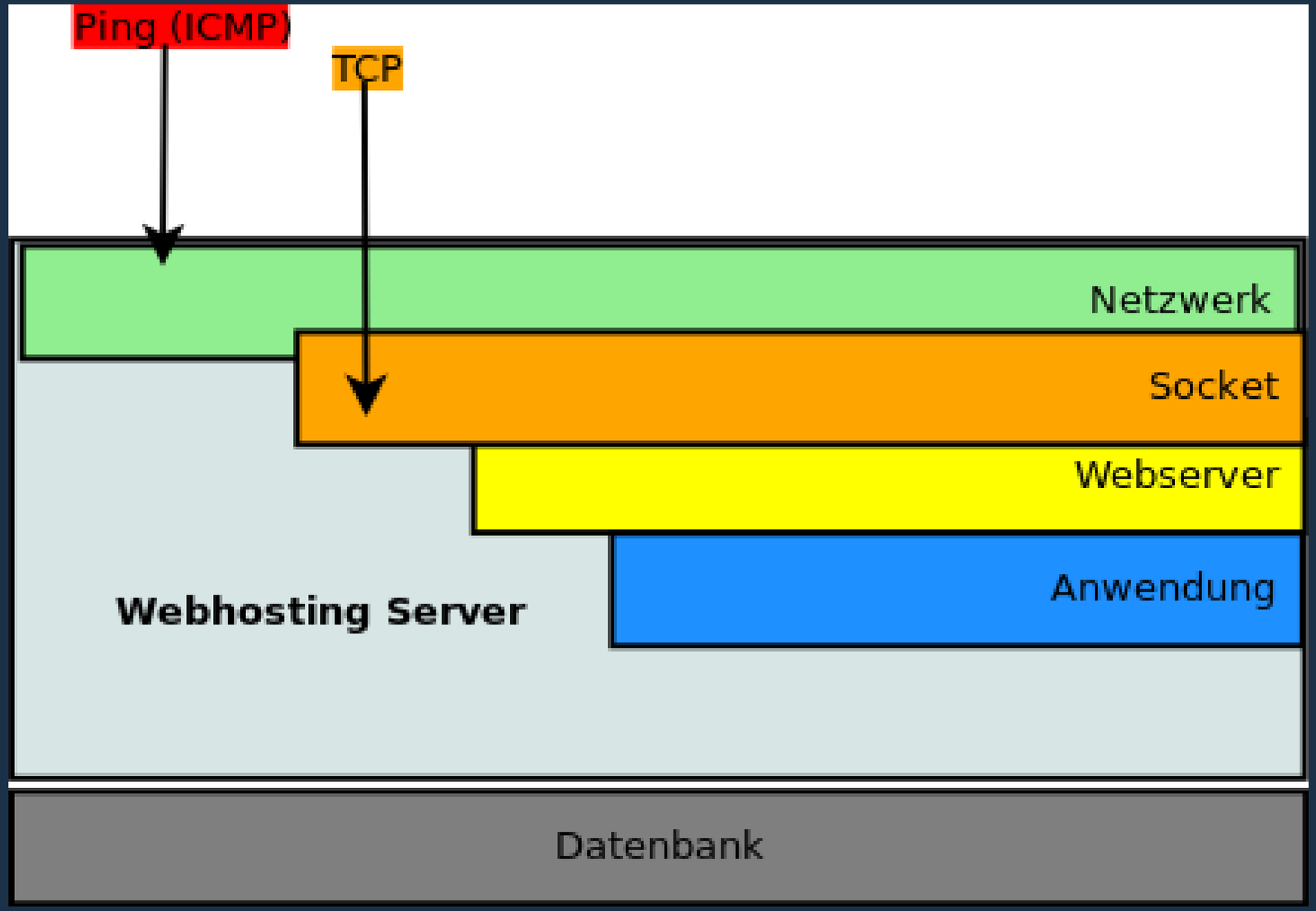
Den richtigen Check verwenden



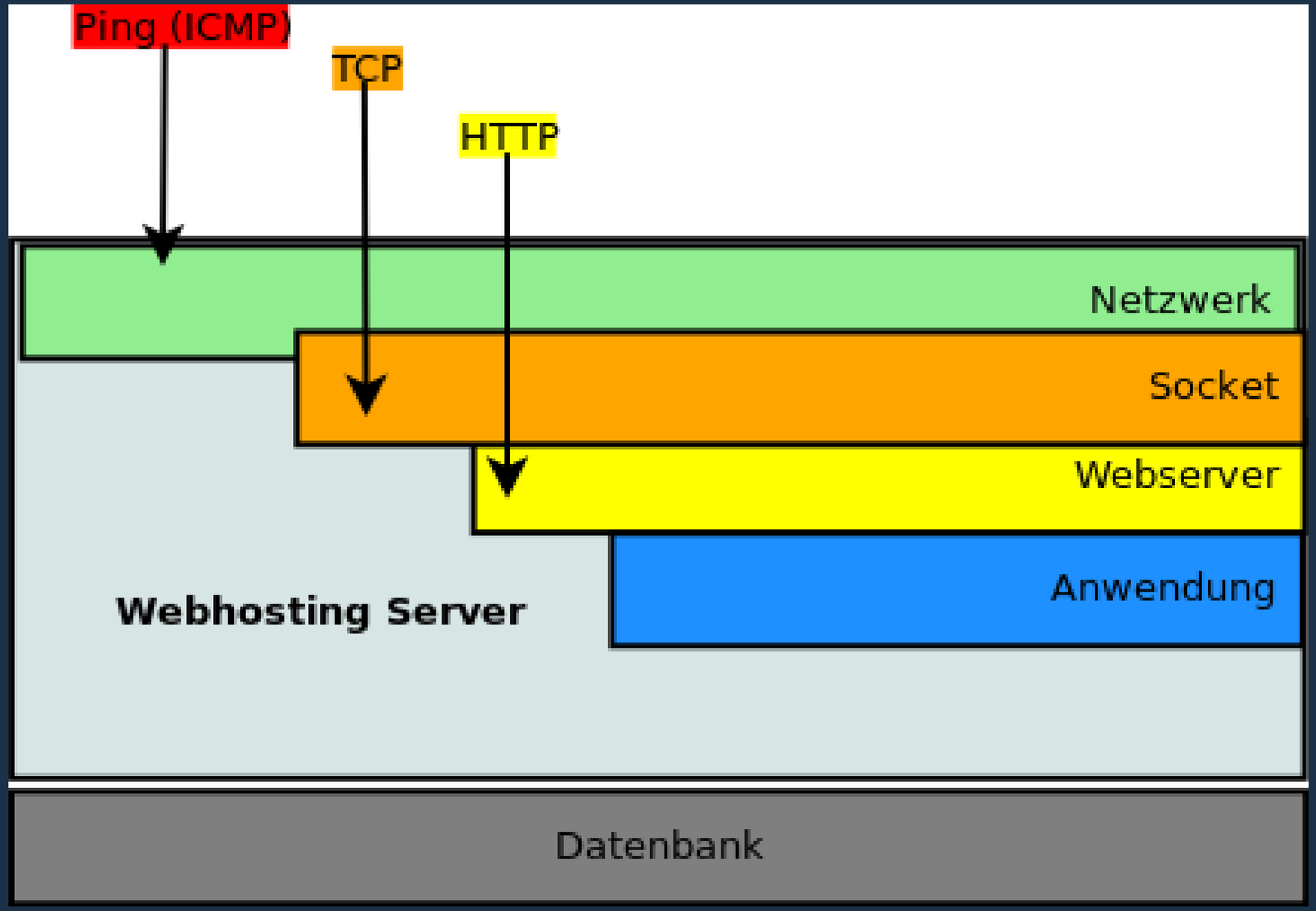
Den richtigen Check verwenden


Ping (ICMP)
Netzwerk
Socket
Webserver
Anwendung
Webhosting Server
Datenbank

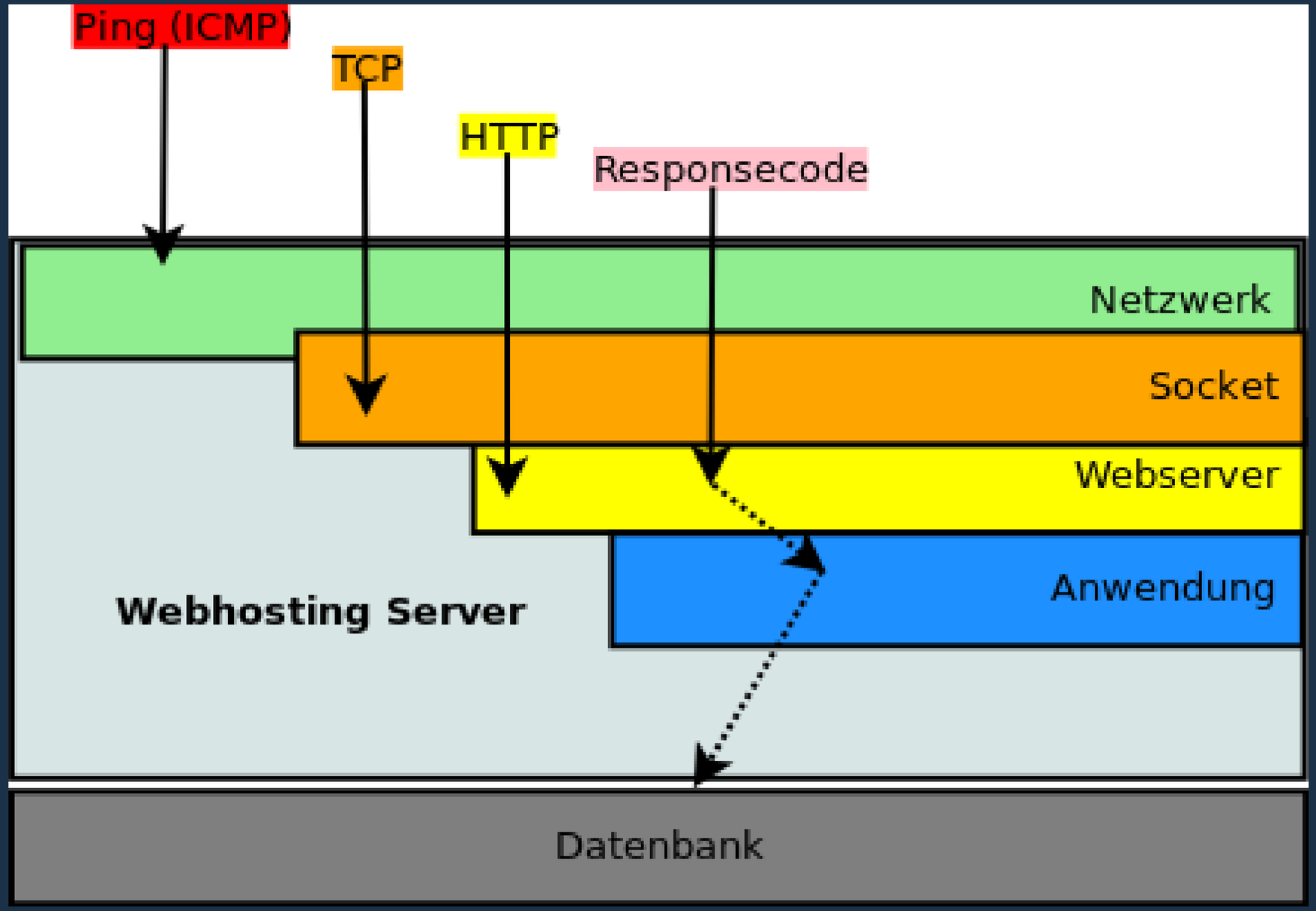
Den richtigen Check verwenden


Ping (ICMP)
TCP
Netzwerk
Socket
Webserver
Anwendung
Webhosting Server
Datenbank

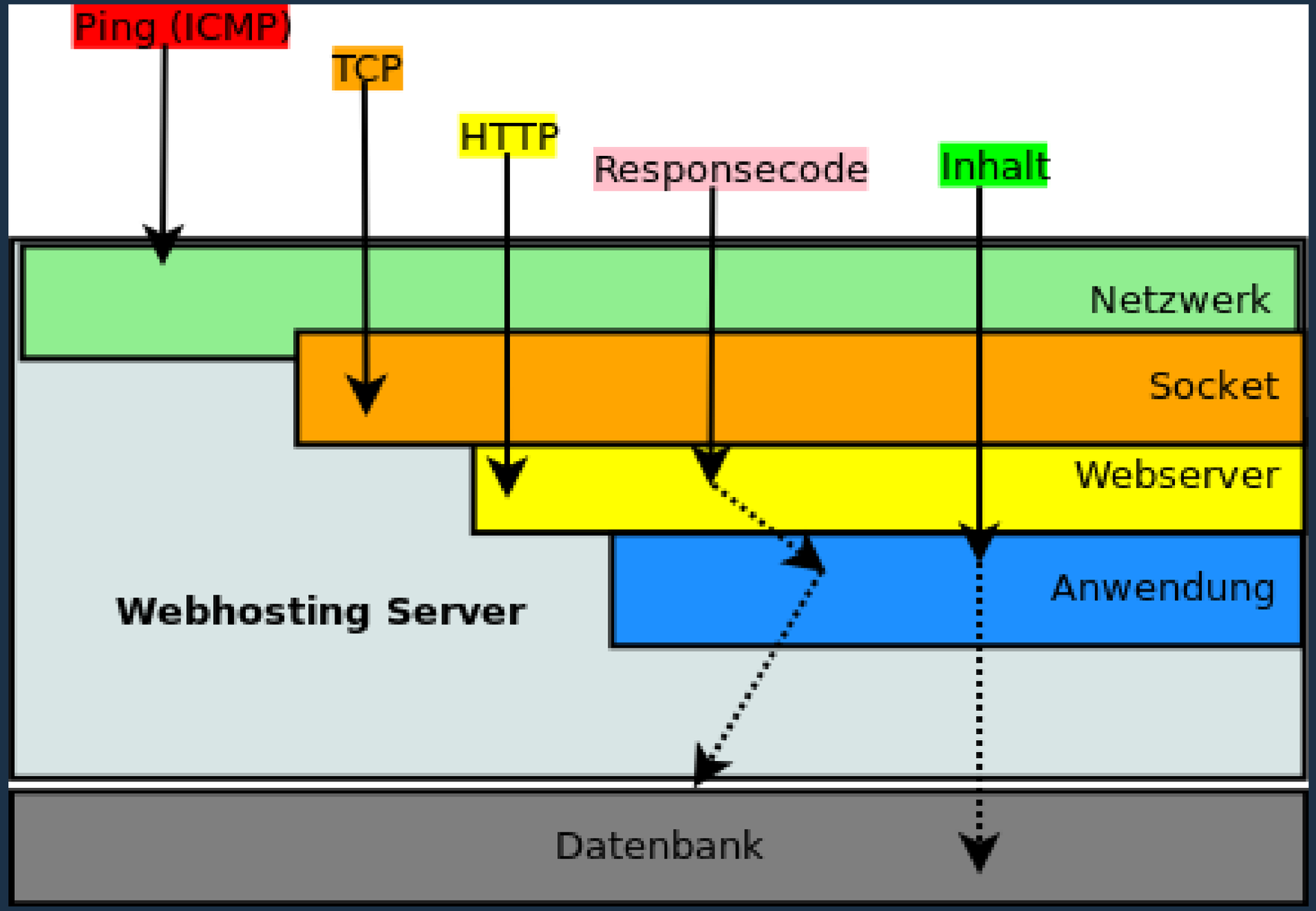
Den richtigen Check verwenden


Ping (ICMP)
TCP
HTTP
Netzwerk
Socket
Webserver
Anwendung
Webhosting Server
Datenbank

Den richtigen Check verwenden


Ping (ICMP)
TCP
HTTP
Responsecode
Netzwerk
Socket
Webserver
Anwendung
Webhosting Server
Datenbank

Den richtigen Check verwenden


Ping (ICMP)
TCP
HTTP
Responsecode
Inhalt
Netzwerk
Socket
Webserver
Anwendung
Webhosting Server
Datenbank

Den richtigen Check verwenden

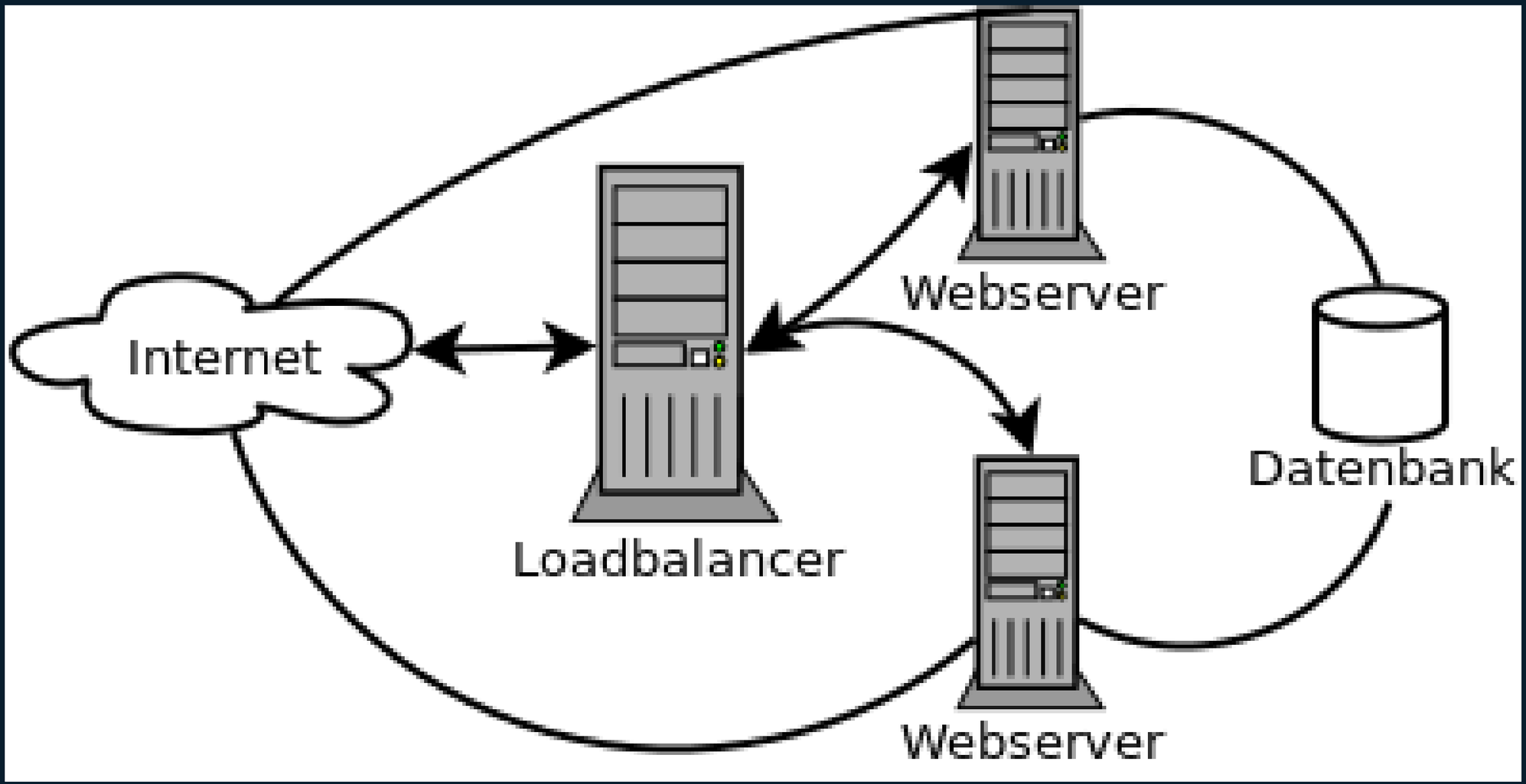
Unser richtiger Check sollte also...

- eine korrekte HTTP Antwort erwarten
- den HTTP Response Code 200 erwarten
- die Antwort innerhalb einer begrenzten Zeit erwarten
- einen Inhalt erwarten der von der Anwendung ausgeliefert werden muss

Noch mehr HTTP checks

- DNS
- Auslieferung mit SSL
- SSL Umleitung
- Auswerten von Anwendungsmetriken

Hosting Setup einer Webseite



Service Check Design

Am Beispiel von Nagios / Icinga

```
define service {
  check_command                   check_http
  check_interval                  5
  retry_interval                  1
  max_check_attempts              3
  check_period                    24x7
  notification_interval           1440
  contact_groups                  admins
  host_name                       makandra.de
  service_description             check_http_makandra.de
  ...
}
```

HTTP Monitoring für 100 Webseiten

Alle Hosts und Services sollten als Template darstellbar sein

- Je Allgemeingültiger der Servicecheck ist, desto besser
- Erreichbarkeit aller Hosts muss nach Möglichkeit gleich gelöst sein

Host Template

```
define hostgroup {
    hostgroup_name          webhosts
}
```

```
define host {
    name                    webhost
    hostgroups              webhosts
    check_command           check-host-alive
    notification_options    d,u,r
    contact_groups          admins,admins_p
    notification_interval   60
    notification_period     24x7
    max_check_attempts      3
    check_interval          3
    retry_interval          1
    register                0
}
```

Service Template

```
define command {
    command_line      $USER1$/check_http -H $HOST$ -e $ARG1$ -t $ARG2$
    command_name      check_http
}
```

```
define service {
    service_description     check_http
    check_command           check_http!200!3
    hostgroup_name          webhosts
    max_check_attempts      3
    check_interval          3
    retry_interval          1
    check_period            24x7
    notification_options    w,c,r,u,f
}
```

Host Definition

```
define host {
  host_name                www.makandra.de
  use                      webhost
}
```

```
define host {
  host_name                www.heise.de
  use                      webhost
}
```

```
define host {
  host_name                www.luga.de
  use                      webhost
}
```

Anwendung auf andere Service Checks

```
define hostgroup {
    hostgroup_name          physical_host
}
```

```
define service {
    service_description     check_disk
    check_command           check_disk!10%!5%
    hostgroup_name          physical_hosts
    max_check_attempts      3
    check_interval          10
    retry_interval          5
    check_period            24x7
    notification_options    w,c,r,u,f
}
```

Anwendung auf andere Service Checks

```
define hostgroup {
    hostgroup_name          postgresql_server
}
```

```
define service {
    service_description     check_postgres_query_time
    check_command           check_pg_query_time!200!50
    hostgroup_name          postgresql_server
    max_check_attempts      3
    check_interval          10
    retry_interval          5
    check_period            24x7
    notification_options    w,c,r,u,f
}
```

```
define host {
  host_name             postgres1.makandra.de
  use                   physical_hosts
  hostgroups            postgresql_server,physical_hosts
}
```

Ausblick auf große und komplexe Infrastrukturen

- Größere Infrastruktur führt zu komplexerem Monitoring
- Komplexeres Monitoring führt zu Fehleranfälligkeit
- Bei makandra im April 2017:
- > 4000 Service Checks
- > 190 Hosts
- Aufzeichnung vieler Performancedaten über 2 Jahre hinweg
- Stetiger Wechsel wenn Hosts kommen und gehen
- Eine Person für Monitoring abstellen oder automatisieren

Vielen Dank fürs zuhören!

maxandra › sucht immer nach Interessenten für anspruchsvollen Serverbetrieb!