

FH JOANNEUM - University of Applied Sciences

GreenWeb Prototype

Prototype-Entwicklung für Energie- und Performance-Messungen

Bereichsübergreifende Projektarbeit (BUEPR)

Verfasser:

Colin Jochum, Tom Kleinhapl, Marcel Kahr

Betreuer:

DI Georg Mittenecker

Graz, 27.01.2020

Inhaltsverzeichnis

1	Kurzfassung.....	4
2	Abkürzungsverzeichnis	5
3	Einleitung.....	5
3.1	Zielsetzung.....	5
3.2	Struktur und Aufbau.....	5
4	Allgemeines	6
5	Lösungsansätze von Institutionen für Green Web/IT	7
6	Infrastruktur.....	9
7	Hardware	10
7.1	DESKTOP-PC G1	10
7.2	DESKTOP-PC G2	10
7.2.1	VM1 – Greta Debian	10
7.2.2	VM2 – Greta Windows 10.....	10
7.2.3	VM3 – Greta Ubuntu 2.....	11
7.2.4	VM4.....	11
7.2.5	VM5 – Greta Ubuntu.....	11
7.3	HP-Server G6	11
7.4	Linksys Router	11
7.5	TP LINK HS110	11
8	Software.....	13
8.1	Open Broadcaster Software (OBS)	13
8.2	NGINX	13
8.2.1	Allgemein	13
8.2.2	NGINX für RTMP Streaming	14
8.2.3	NGINX Installation incl. RTMP Modul auf CENTOS 7 Fehler! Textmarke nicht definiert.	
8.3	ffmpeg	18
8.4	Snapd.....	21
8.5	Apache.....	22
8.6	JMeter	22
8.7	Intel Power Gadget	25
8.8	HP ILO	25
8.9	Loadtest.....	26

9	Messungstools	27
10	Messergebnisse	27
11	Energieverbrauch bei der Übertragung über das Internet	49
12	Probleme & Lessons Learned.....	50
13	Empfehlung und Ausblick	51
14	Zusammenfassung / Conclusio	52
15	Projektantrag	53
16	Meilensteine	55
17	Projektplan.....	56
18	Gantt Chart.....	57
19	Stundenerfassung	58
20	Tätigkeitsbericht	59
21	Zugangsdaten BÜPA.....	66
22	IP Konzept	66
23	Literaturverzeichnis	68
24	Abbildungsverzeichnis	69

1 Kurzfassung

2 Abkürzungsverzeichnis

3 Einleitung

In der Lehrveranstaltung „Bereichsübergreifende Projektarbeit“ kurz BUEPR, welche im 5. Semester des Studiengang Informationsmanagement absolviert werden muss, entschieden wir uns für das Thema GreenWeb Prototype. Dieses Thema wurde von Herrn DI Georg Mittenecker in Kooperation mit Herrn Gerhard Sprung, MSc, welcher das Thema GreenWeb Visualisierung betreut, erstellt. Dabei handelt es sich um die Entwicklung eines Prototyps zur Energie- und Performance-Messungen. Nach dem Kickoff-Meeting und ausführlicher Absprache im Projektteam, kamen wir zum Entschluss, Energieverbrauchsmessungen sowohl serverseitig als auch clientseitig beim Streamen von Videos umzusetzen. Dabei wird der Energieverbrauch diverser Qualitätsstandards, Verschlüsselungstechniken, als auch der Zugriff auf dynamische und statische Webseiten betrachtet. Da es im Bereich GreenWeb zwei Projektgruppen gibt, tauschen wir die gemessenen Daten mit dem anderen Team aus, damit diese etwaigen Visualisierungen umsetzen können.

3.1 Zielsetzung

Die Zielsetzung der Projektarbeit ist es eine funktionierende Netzwerkinfrastruktur aufzubauen, um Stromverbrauchsmessungen durchführen zu können. Dabei werden zwei verschiedene CPU Generationen, ein HP-Server, sowie die eigenen Notebooks der Studenten verwendet, um aussagekräftige Ergebnisse zu erzielen. Die Systeme werden zum Teil physisch auf dem OS aufgesetzt, jedoch zum überwiegenden Teil in virtualisierter Form (VMs). Weiters soll anhand diverser Tools und Schnittstellen der Energieverbrauch ermittelt, ausgelesen und untereinander verglichen werden. Die einzelnen Tools bzw. Schnittstellen werden in einem späteren Kapitel genauer betrachtet. Bei den Messungen werden natürlich diverse Einstellungen, wie unterschiedliche Qualitätsstandards (1080p, 4k), Verschlüsselungstechniken oder Codecs, berücksichtigt. Durch diese Einstellungen soll auch veranschaulicht werden, welche Unterschiede diese bewirken und ob mehr oder weniger Energie dafür benötigt wird. Weiters sollen die Messergebnisse aussagekräftig und lesbar sein, damit das Partnerteam mit den Ergebnissen anschauliche Visualisierungen generieren kann. Neben der praktischen Umsetzung und Analyse geht es auch darum, durch Recherchen auf schon bekannte bzw. in Aussicht befindliche Lösungsansätze von Institutionen wie Facebook und Co. für GreenWeb/IT zu finden und zu verweisen.

3.2 Struktur und Aufbau

Anhand der vordefinierten Meilensteine wird nach dem Projektstart und dem offiziellen Kickoff-Meeting zuerst der Aufbau der Netzwerkinfrastruktur und die Installation diverser Softwarekomponenten wie Windows Server 2019, Native Linux und andere Linux Distributionen für die virtuellen Maschinen, durchgeführt. Weiteres wird nach einem

geeigneten Webserver sowie etwaige Tools zum Messen des Energieverbrauchs recherchiert. Diese Tools werden auf ihre Kompatibilität für die etwaigen in Einsatz befindlichen System getestet, um damit aussagekräftige Messungen durchführen zu können. Nach erfolgreichem Aufbau der Infrastruktur und testen diverser Tools werden erste Probetests und Messungen durchgeführt. Weiters wird ein Plan erstellt, in dem eingetragen wird, welche Systeme wie und mit welchem Tool untereinander verglichen werden. Nach erfolgreicher Dokumentation der Messergebnisse werden wir ein finales Abschlussmeeting mit unserem Projektbetreuer ansetzen. Die Messergebnisse werden mit dem Partnerteam geteilt und eine Gesamtdokumentation des Projektes wird verfasst. Zu guter Letzt erfolgt die Projektpräsentation sowie die Abgabe des Projektes.

4 Allgemeines

Greenpeace veröffentlichte eine Studie namens „Clicking Green“ und verdeutlichte, dass wenn das Internet ein Land wäre, den sechstgrößten Energieverbrauch der Welt hätte.

Die größten Stromverbraucher können grundsätzlich in vier große Gruppe unterteilt werden:

1. PCs, Laptops, Smartphones, Tablets, Spielekonsolen
2. Server in Daten- und Rechenzentren
3. Mobilfunkstationen und Router
4. Herstellung der Hardware für die Punkte Eins, Zwei und Drei

Laut Google soll das Suchen nach „Wieviel Strom verbraucht einmal „Googlen“?“, rund 0,3 Wattstunden verbrauchen. Das bedeutet, dass mit rund 200 Suchanfragen ein Hemd gebügelt oder mit 3000 Suchanfragen ein Eimer Wasser zum Kochen gebracht werden kann.

Die digitalen Technologien verursachen rund vier Prozent der gesamten Treibhausgasemission, was sogar mehr ist, als die Luftfahrt produziert.

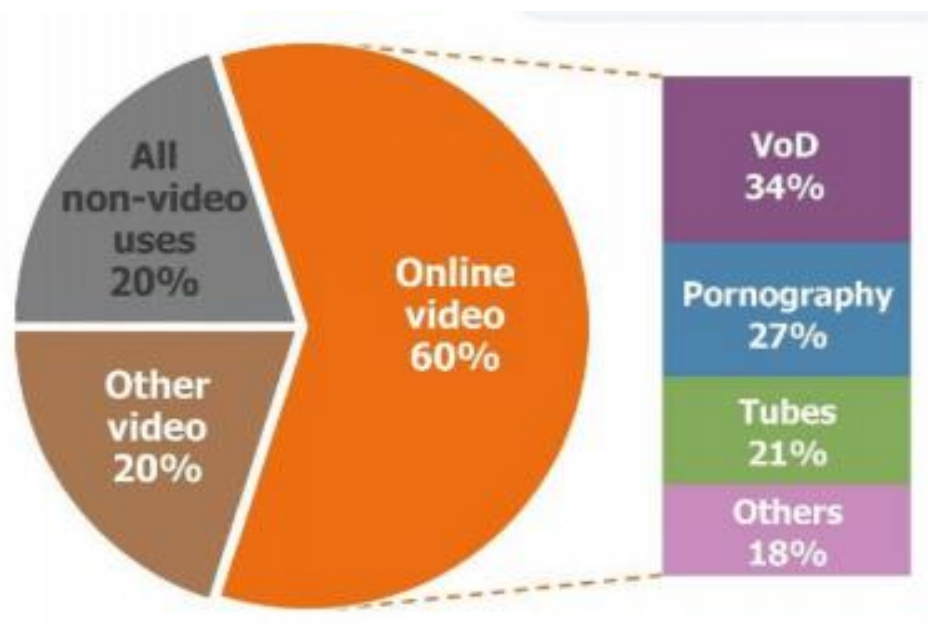
Rund 80% des weltweiten digitalen Datenverkehrs entsteht beim Übertragen von Videos. Dieser Datenverkehr kann in drei große Bereiche eingegliedert werden. Der größte Teil ist das Streamen von Onlinevideos. Das Übertragen von Onlinevideos (60% des Weltdatenverkehrs) machte im Jahr 2018 rund 1,05 Zettabytes auf. Das sind rund $1,127429 \times 10^9$ Terabytes, was für Menschen nahezu unvorstellbar ist. Dabei entstanden rund 306 Millionen Tonnen an CO₂, was rund 20 Prozent der Treibhausgasemission im Bereich der digitalen Technologie ausmacht. Weltweit gesehen sind das trotzdem nur gut ein Prozent der gesamten Treibhausgasemission.

Der Bereich des Online Videos kann in vier weitere Teilbereiche unterteilt werden. 34 Prozent wird von Video on Demand, das Streamen von Filmen und Videos über Netflix und Amazon Prime, verursacht. Weitere 27 Prozent fallen auf das Streamen von pornografischen Inhalten. Der dritte Teilbereich bezeichnet sich als „Tube“ und beinhaltet unterschiedliche Videostreaming Inhalte, dominierend zu 95 Prozent von YouTube. Zu guter Letzt

verursachen rund 18 Prozent des Online Video Streamings Soziale Netzwerke wie Facebook, Instagram, Snapchat oder Twitter.

Auf den Bereich „Andere Videos“ fallen rund 20 Prozent und beinhaltet alles rund um Live Fernsehen, Live Videos (Skype, „Camgirls“ etc.) oder auch Video Überwachung.

Nur rund 20 Prozent des gesamten weltweiten Datenverkehrs wird nicht für Videos verwendet. Dazu gehören Webseiten, E-Mails, Sofortnachrichten (WhatsApp), Onlinespeicher diverser Fotos und Daten und Firmennetzwerke.



5 Lösungsansätze von Institutionen für Green Web/IT

Heutzutage setzen vor allem große Unternehmen wie Apple, Google, Facebook und Co. in zahlreichen Projekten auf Nachhaltigkeit und erneuerbare Energien. Dabei wird versucht der fortschreitenden Klimaerwärmung und Umweltverschmutzung entgegenzuwirken sowie den Energieverbrauch zu minimieren und Energiequellen effizienter zu nutzen.

Apple ist Spitzenreiter im Bereich Green IT und betreibt schon heute seine weltweiten Einrichtungen aus CO₂ emissionsfreien Strom. Nicht nur der Stromverbrauch der Einrichtungen steht im Mittelpunkt sondern auch der Einsatz von recycelten und nicht gesundheitsschädlichen Stoffen in ihren Produkten. Dabei wird vor allem auf Quecksilber, PVC oder Arsen vermieden.

Neben Apple ist auch Google unter den Top drei der grünsten Unternehmen. Seit 2007 nutzt Google bereits erneuerbare Energien für den Betrieb ihrer Rechenzentren. Ein interessantes Projekt von Google ist Sunroof. Es ermöglicht dem Kunden mit Hilfe von Google Maps einen persönlichen Solarplan für das eigene Dach zu erstellen.

Das drittgrünste Unternehmen ist Facebook. Das Unternehmen möchte seine Treibhausgasemission von 2017 bis 2010 um rund 75 Prozent reduzieren und ab 2020 ihre gesamten Rechenzentren mit Energie aus 100 Prozent erneuerbarer Energie betreiben.

Aus der unterhalb ersichtlichen Grafik sind alle fünfzehn weltweit verteilten Rechenzentren von Facebook sowie ihre erneuerbaren Energiegewinnungsquellen ersichtlich.

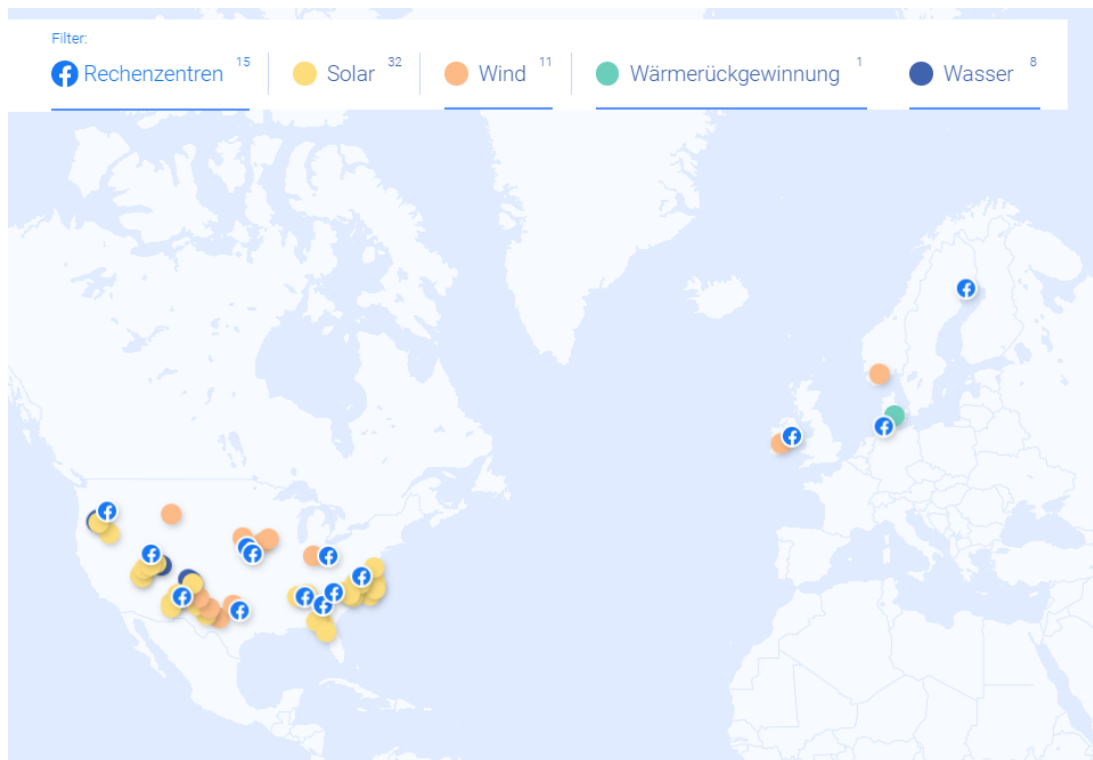
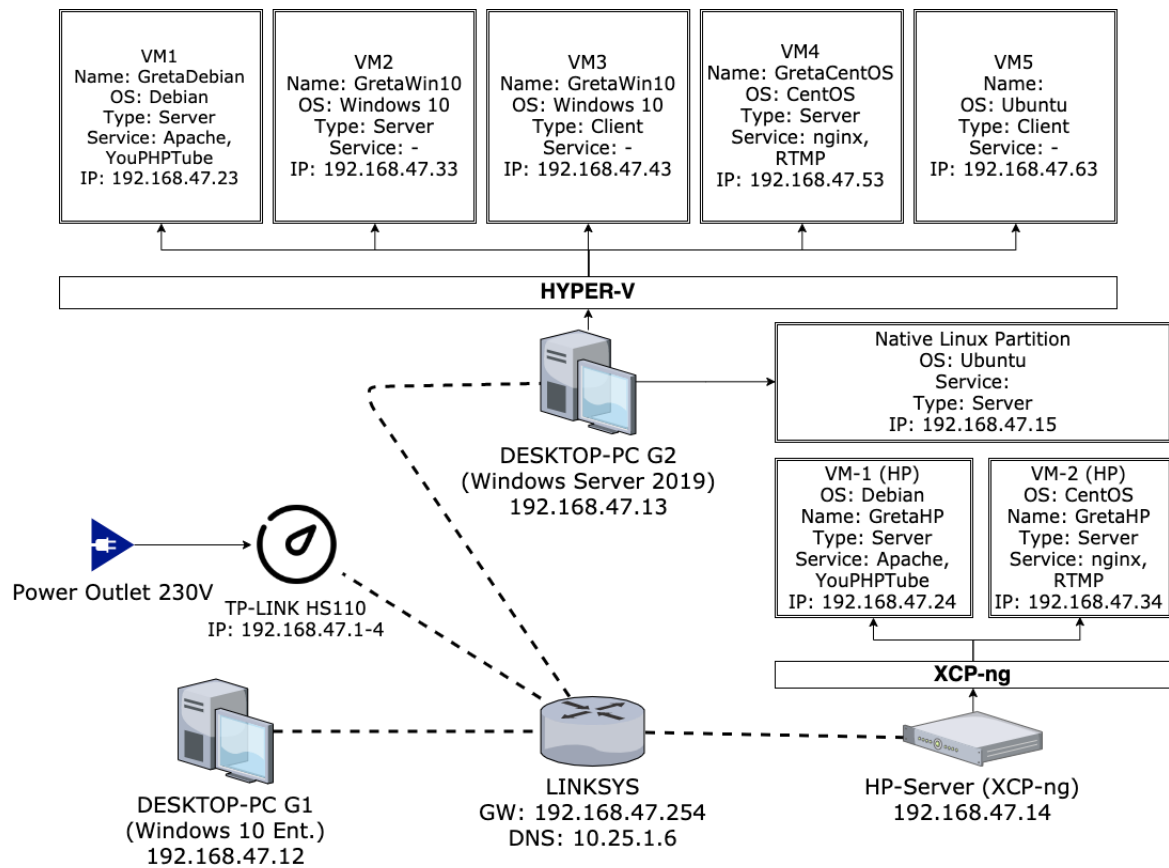


Abbildung 1: Übersichtskarte Erneuerbare Energien und Rechenzentren

Facebook nutzt beispielsweise in Dänemark für ihre Rechenzentren eine sogenannte Wärmerückgewinnung Methode. Hiermit wird die überschüssige Wärme der Server in ein lokales Fernwärmesystem weitergeleitet.

Amazon ist im Bereich Green IT noch etwas im Rückstand. Amazon hat sich bis zum Jahr 2040 das Ziel gesetzt, seine CO₂ Werte auf Null zu setzen. Hauptaugenmerk ist dabei der Transportsektor. Das Unternehmen möchte durch Alternativlösungen wie Elektrifizierung der Fahrzeuge oder die Auslieferungen in Ballungszentren mit Fahrrädern oder Prime Air Drohnen. Nicht nur im Transportsektor, sondern auch in der Cloud möchte Amazon klimaneutral arbeiten. Seit 2018 konnte die Energie für Amazon Web Service aus 50 Prozent erneuerbarer Energien gewonnen werden. Zur Minimierung des Trinkwasserverbrauchs für die Kühlung der Server in den Rechenzentren setzt Amazon auf eine sogenannte Verdunstungskühlung.

6 Infrastruktur



7 Hardware

Da momentan sehr viel Unruhe und Tumulte im Bereich Klimaschutz und Klimaerwärmung vorherrschen, einigten wir uns all unsere PCs und VMs nach der Klimaaktivsten Greta Thunberg zu benennen.

Da wir hauptsächlich mit den eigenen Laptops arbeiteten, nutzten wir Putty um über SSH auf die VMs zugreifen zu können. Weiters nutzten wir ein sehr mächtiges Tool namens WinSCP. Eine Windows Client, der das Secure Copy Protokoll implementiert, welches ebenfalls mit SSH Verbindungen arbeitet. Es ermöglicht es sich auf diverse Server zu verbinden und per Drag and Drop direkt Dateien von einem Server auf den andren Server zu verschieben.

7.1 DESKTOP-PC G1

Der Desktop-PC HP Elite Desk ist unser Upstream Client und wird rein zum Streamen und Aufnehmen des Video Samples „Big Buck Bunny“ mit Hilfe von OBS Studio verwendet. Auf diesem PC wurde Windows 10 Enterprise installiert.

7.2 DESKTOP-PC G2

Der Desktop-PC HP Elite Desk mit der CPU Intel Core i7 – 6700 wird zur Bereitstellung diverser Serverumgebungen. Dabei wird zwischen zwei Partitionen unterschieden. Die eine Partition hostet die native Linux Distribution Ubuntu, welcher zu Beginn der BÜPA für Testzwecke von diversen Measurement Tools war. Auf der zweiten Partition wurde Windows Server 2019 installiert, auf welcher die einzelnen VMs laufen.

7.2.1 VM1 – Greta Debian

Auf der VM1 wurde die Linux Distribution Debian installiert. Diese virtuelle Maschine wird als Server ausgeführt und hostet unseren NGINX-Server welcher als Videostreaming Server eingesetzt wird.

7.2.2 VM2 – Greta Windows 10

Auf der zweiten virtuellen Maschine wurde Windows 10 installiert. Hier wurde der Apache Webserver zum Hosten der statischen und dynamischen Website installiert. Weiters wurde das XAMPP Control Panel installiert, mit dem sich die MySQL Datenbank sowie der Apache Server starten lässt.

7.2.3 VM3 – Greta Ubuntu 2

Auf der dritten VM wurde die Linux Distribution Ubuntu installiert, welche als Webserver zum Hosten der statischen und dynamischen Website NGINX verwendet wird. Als Datenbank wird wiederum MySQL eingesetzt.

7.2.4 VM4

7.2.5 VM5 – Greta Ubuntu

Auf der fünften virtuellen Maschine wurde wiederum die Linux Distribution Ubuntu installiert, welche den selbst erstellten YouTube Clone hostet. Dieser wird für Video-on-Demand Testzwecke eingesetzt.

7.3 HP-Server G6

Auf unserem HP-Server ProLiant DL380 G6 wurden zwei VMs installiert. Die VM1 hostet das Linux OS Debian, worauf der Webserver Apache sowie der YouTube Clone installiert wurden. Die zweite virtuelle Maschine basiert auf der Linux Distribution CentOS worauf der Webserver NGINX und RTMP installiert wurde.

7.4 Linksys Router

Der Linksys Router wurde mit Open WRT konfiguriert. Außerdem wurde die Vergabe von IP Adressen für externe Geräte mit Hilfe eines DHCP-Servers geregelt. Allen physischen Desktop PCs, Server und VMs wurde eine statische IP Adresse zugeteilt. Der DHCP Server wurde benötigt, damit die TP Links konfiguriert und ins Netzwerk eingebunden werden konnten.

7.5 TP LINK HS110

Eine hardwareseitige Messmethode, die für unser Projekt ausgewählt wurde, ist der TP-Link HS110. Die TP-Links wurden mit Hilfe eines Smartphones und der APP Kasa Smart konfiguriert. Damit können die TP-Links auch ein- und ausgeschaltet werden und bekommen einen Überblick des Stromverbrauchs. Dieser eignet sich besonders gut für unsere Messungen da er vom Unternehmen softScheck, welches sich auf Software Schwachstellen bzw. Sicherheitslücken spezialisiert hat, analysiert wurde und durch eine Schwachstelle ein vollständiger Zugriff auf das Gerät möglich ist.

Die genaue Analyse der Sicherheitslücke wurde von softScheck genauestens auf ihrer Webseite dokumentiert: <https://www.softscheck.com/en/reverse-engineering-tp-link-hs110/>

SoftScheck hat zudem das Python 2 Script öffentlich zugänglich gemacht um die Daten sehr einfach und komfortabel auszulesen. GitHub Repository:

<https://github.com/softScheck/tplink-smartplug>

Durch diese Schwachstelle können Befehle im JSON Format an das Gerät gesendet und empfangen werden.

TP-LINK HS110 Commands

on	Turn on the plug
off	Turn off the plug
info	Return device information
time	Return system time
reboot	Reboot device
reset	Reset the device to factory settings
energy	Return realtime voltage/current/power

Als Grundlage für unser Script wurde das Projekt von softScheck verwendet welches in weiterer Folge auf die neueste Python v3 aktualisiert wurde. Zusätzlich wurden zusätzliche Komponenten implementiert welche ein Logging...

8 Software

8.1 Open Broadcaster Software (OBS)

Die Open Broadcaster Software (OBS) wird verwendet, um Inhalte auf dem Bildschirm des Users über das Internet in Echtzeit zu übertragen. Die Datenübertragung hierfür erfolgt über das Real Time Messaging Protokoll, welches auf der Empfängerseite von Streamingdiensten empfangen und verarbeitet. Die Zuordnung der Daten sowie die Authentifizierung erfolgt hierbei über sogenannte Streaming Schlüssel.

Als Encoder stehen in OBS x264, Intel Quick Sync Video und Nvidia NVENC zur Verfügung, welche in weiterer Folge den Stream im H.264 Format kodieren. Für die Audioübertragung kann auf die MP3 oder AAC Kodierung zurückgegriffen werden.

Die Open Source Broadcasting Software OBS stellt für uns ein geeignetes Tool dar, da es möglich ist Video Streaming Workloads in unterschiedlichen Qualitätsstufen und Kodierungen zu generieren und deren Einfluss auf den Stromverbrauch zu messen.

OBS wurde auf dem DESKTOP-PC G1 installiert da dieser als Streaming Client definiert ist. Der Stream wird in weiterer Folge an den DESKTOP-PC G2 (VM4) geschickt auf welchem der Streaming Server nginx (incl. RTMP Extension) installiert wurde.

8.2 NGINX

8.2.1 Allgemein

NGINX wird als Open Source Projekt betrieben und kann für eine Vielzahl von Anwendungen verwendet werden. Diese Flexibilität war einer der Gründe warum wir uns für diesen entschieden haben. Ressourcenschonen, hohe Performance, und steigende Verwendung im Business Bereich waren weitere Gründe, um NGINX anstelle von Apache zu verwenden. Zudem verfügt NGINX über eine große Community was es uns erleichtert den RTMP Streaming Server zu konfigurieren.

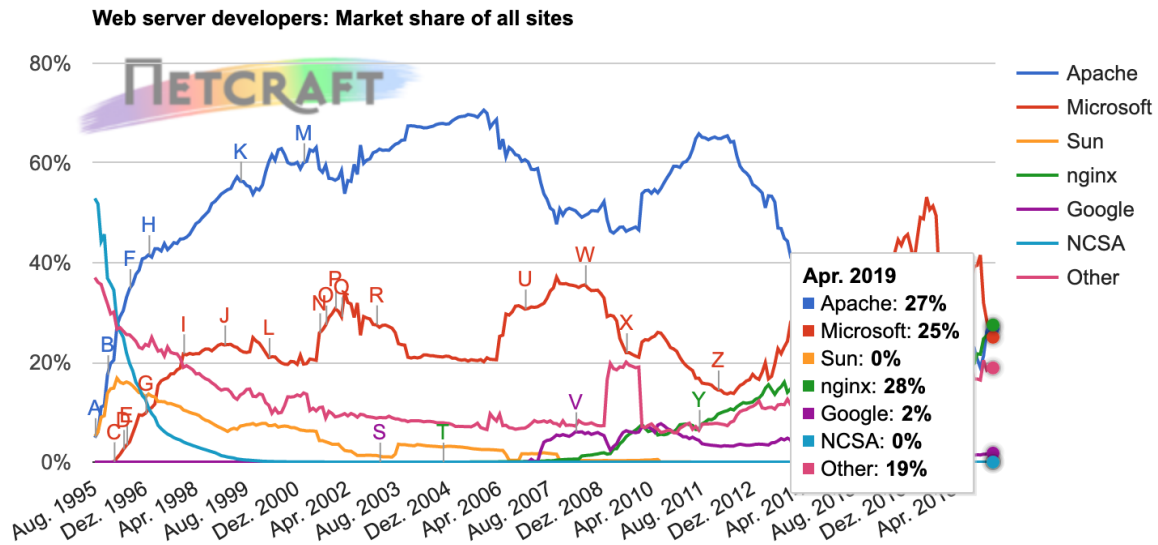


Abbildung 2: <https://news.netcraft.com/archives/2019/04/22/april-2019-web-server-survey.html>

8.2.2 NGINX für RTMP Streaming

In unserem Project wurde NGINX mit dem RTMP Modul auf der VM4 des DESKTOP-PC G2 installiert und fungiert hier als Streaming Server, welcher den Stream des Clients entgegennimmt und an die Zuseher (Clients) verteilt.

Auf dem NGINX RTMP Server werden zwei „use-cases“ behandelt.

- Zum einen wird der Stream welcher an den Server geschickt wird direkt an die Zuseher verteilt. Diese Verteilung erfolgt ohne jegliche Veränderung des Videos.
- Zum anderen wird der Stream mithilfe von ffmpeg eine niedrigere Qualitätsstufe gebracht. Diese Konfiguration erfolgt direkt in der NGINX Konfigurationsdatei.

8.2.3 NGINX Installation incl. RTMP Modul

Centos:

Für die installation von nginx mit dem zugehörigen RTMP Modul wurde das Tutorial von Samuyi mit dem Titel: „How to Setup Nginx for hls video streaming on Centos 7“ verwendet. Da das RTMP Modul nicht standardmäßig in nginx verfügbar ist muss das ganze Programm einschließlich gewünschter Module neu kompiliert werden.

<https://dev.to/samuyi/how-to-setup-nginx-for-hls-video-streaming-on-centos-7-3jb8>

Hier werden nochmal die grundlegenden Installationsschritte zusammengefasst:

Folgende Module müssen auf CentOS vorhanden bzw. installiert sein, um einen reibungslosen Installationsprozess zu gewährleisten.

- git, wget, gcc, gcc-c++, perl, gd, gd-devel, perl-ExtUtils-Embed, geoip, geoip-devel, tar

CENTOS aktualisieren & Pakete Installation

```
> sudo yum update
> sudo yum install epel-release
> sudo yum install git wget gcc gcc-c++ tar gd gd-devel perl-ExtUtils-Embed geoip
geoip-devel
```

Pearl Regular Expression (PCRE) Installation

```
> wget ftp.pcre.org/pub/pcre/pcre-8.42.tar.gz
> tar -zxf pcre-8.42.tar.gz
> rm -rf pcre-8.42.tar.gz
> cd pcre-8.42
> ./configure
> make
> sudo make install
```

ZLIB (v1.2.11) compression Library Installation

```
> wget http://zlib.net/zlib-1.2.11.tar.gz
> tar -zxf zlib-1.2.11.tar.gz
> rm -rf zlib-1.2.11.tar.gz
> cd zlib-1.2.11
> ./configure
> make
> sudo make install
```

OPEN SSL (v.1.0.2q) Installation

```
> wget http://www.openssl.org/source/openssl-1.0.2q.tar.gz
> tar -zxf openssl-1.0.2q.tar.gz
> rm -rf openssl-1.0.2q.tar.gz
> cd openssl-1.0.2q
> ./config
> make
> sudo make install
```

NGINX-RTMP-MODULE Open Source Projekt von <https://github.com/arut/nginx-rtmp-module>

```
> git clone https://github.com/arut/nginx-rtmp-module
```

NGINX (v1.14.2) installation

```
> wget https://nginx.org/download/nginx-1.14.2.tar.gz
> tar zxf nginx-1.14.2.tar.gz
> rm -rf nginx-1.14.2.tar.gz
> cd nginx-1.14.2
```

NGINX Konfigurationsoptionen mit den zugehörigen Modulen

```
> ./configure --add-module=../nginx-rtmp-module \
--sbin-path=/usr/sbin/nginx \
--lock-path=/var/run/nginx.lock \
--conf-path=/etc/nginx/nginx.conf \
--pid-path=/run/nginx.pid \
--with-pcre=../pcre-8.42 \
--with-zlib=../zlib-1.2.11 \
--with-openssl=../openssl-1.0.2q \
--error-log-path=/var/log/nginx/error.log \
--http-log-path=/var/log/nginx/access.log \
--user=nginx \
--group=nginx \
--with-http_auth_request_module \
--with-http_degradation_module \
--with-http_geoip_module \
--with-http_gunzip_module \
--with-http_gzip_static_module \
--with-http_image_filter_module \
--with-http_mp4_module \
--with-http_perl_module \
--with-http_realip_module \
--with-http_secure_link_module \
--with-http_slice_module \
--with-http_ssl_module \
--with-http_stub_status_module \
--with-http_v2_module \
--with-stream_ssl_module \
--with-stream \
--with-threads \
--prefix=/etc/nginx
```

NGINX kompilieren

```
> make
> sudo make install
```

NGINX USER ERSTELLEN / LOG DIRECTORY

```
> sudo useradd --system --home /var/lib/nginx --shell /sbin/nologin --comment
"nginx system user" nginx
> sudo mkdir /var/log/nginx && sudo chown nginx:nginx /var/log/nginx
```

NGINX service file

```
> sudo vim /lib/systemd/system/nginx.service
```

[Unit]

Description=nginx - high performance web server

Documentation=<https://nginx.org/en/docs/>

After=network.target remote-fs.target nss-lookup.target

Wants=network-online.target

[Service]


```
Type=forking
PIDFile=/run/nginx.pid
ExecStartPre=/usr/bin/rm -f /run/nginx.pid
ExecStartPre=/usr/sbin/nginx -t -c /etc/nginx/nginx.conf
ExecStart=/usr/sbin/nginx -c /etc/nginx/nginx.conf
ExecReload=/bin/kill -s HUP $MAINPID
KillSignal=SIGQUIT
TimeoutStopSec=5
KillMode=process
PrivateTmp=true
[Install]
WantedBy=multi-user.target
```

RELOAD AND RESTART NGINX

```
> sudo systemctl daemon-reload
> sudo systemctl start nginx
> sudo systemctl enable nginx
```

Video Stream Files directory

```
> sudo mkdir -p /var/www/hls/live
```

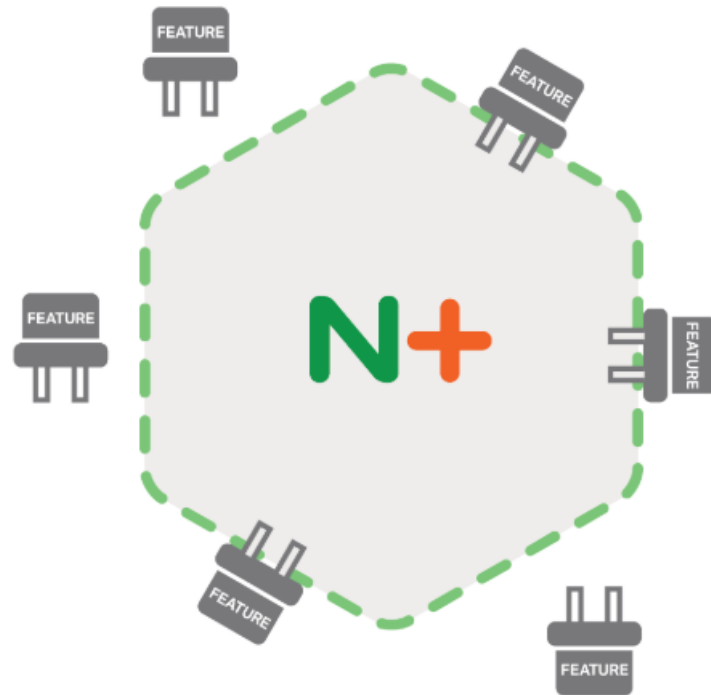
NGINX CONFIG FILE /etc/nginx/nginx.conf

...

Ubuntu Server:

Da es nicht möglich war die über Snap installierte Version von ffmpeg aus der nginx Konfiguration auszuführen wurde das nginx Setup auf eine Ubuntu Server VM verschoben. Ein großer Vorteil bei der Installation auf Ubuntu ist, dass das RTMP Modul in Aptitude, dem standardmäßigen Paketmanager auf Debian basierenden Linux Distributionen, verfügbar ist. Nginx unterstützt seit Version 1.9.11 dynamische Module. Das ermöglicht es Features in form von Modulen in Nginx zu aktivieren ohne es neu kompilieren zu müssen. Diese Module müssen entweder selbst kompiliert werden oder falls sie in Aptitude verfügbar sind einfach installiert werden. Das RTMP Module ist unter dem Namen libnginx-mod-rtmp im Repository zu finden. Ist das Paket einmal installiert muss nur mehr in der Nginx Konfiguration darauf verwiesen werden:

```
load_module "modules/ libnginx-mod-rtmp";
```



<https://docs.nginx.com/nginx/admin-guide/dynamic-modules/dynamic-modules/>

8.3 ffmpeg

Bei ffmpeg oder auch „Fast Forward MPEG“ genannt, handelt es sich um ein vielfach genutztes Multimedia-Framework und Befehlszeilenprogramm mit zahlreichen Funktionen. Es bietet Möglichkeiten beinahe alles zu dekodieren, kodieren, transkodieren, streamen oder zu filtern. Es unterscheidet sich von anderen GUI Programmen dahingehend, da es die WYSIWYG-Methode (was Sie sehen, ist was Sie bekommen) verwenden.

<https://snapcraft.io/install/ffmpeg/centos>

<https://riptutorial.com/de/ffmpeg/example/24782/was-ist-ffmpeg->

ffmpeg liest eine beliebige Anzahl von Dateien ein und schreibt eine beliebige Anzahl von Ausgangsdateien, die angegeben werden in einen einfachen Ausgabe-URL.

<https://ffmpeg.org/ffmpeg.html#Description>

Zum Encoden wird folgender ffmpeg Befehl als Vorlage verwendet:

```
ffmpeg -i rtmp://localhost:1935/live/ -r 60 -vf scale=1920:1080 -c:v  
libx264 -preset medium -b:v 6M -minrate 6M -maxrate 6M -bufsize 12M  
-b:a 320k -f flv rtmp://localhost:1935/mobile/;
```

Zum besseren Verständnis wird nachfolgend auf diverse Parameter eingegangen.

- i -> Input
- r -> Frames per Second
- vf scale -> Resolution
- c:v -> Video Kodierung
- b:v -> Video Bitrate
- preset -> Voreinstellung für das Verhältnis von Kodierungsgeschwindigkeit zur Komprimierung
- minrate -> Minimum bitrate
- maxrate -> Maximum bitrate
- bufsize -> Twice the maxrate
- b:a -> Audio Bitrate
- f Output Format

Preset / Voreinstellung

Das Attribut Preset umfasste eine Sammlung von Optionen, welches ein bestimmtes Verhältnis von Kodierungsgeschwindigkeit zur Komprimierung darstellt. Im Allgemeinen heißt das, mit einer langsameren Voreinstellung erzielt man eine bessere Komprimierung der Qualität. Aus der unten angegebenen Grafik können die unterschiedlichen Kodierungsgeschwindigkeiten entnommen werden. Falls kein Preset definiert wird, verwendet ffmpeg standardmäßig die Kodierungsgeschwindigkeit mittel.

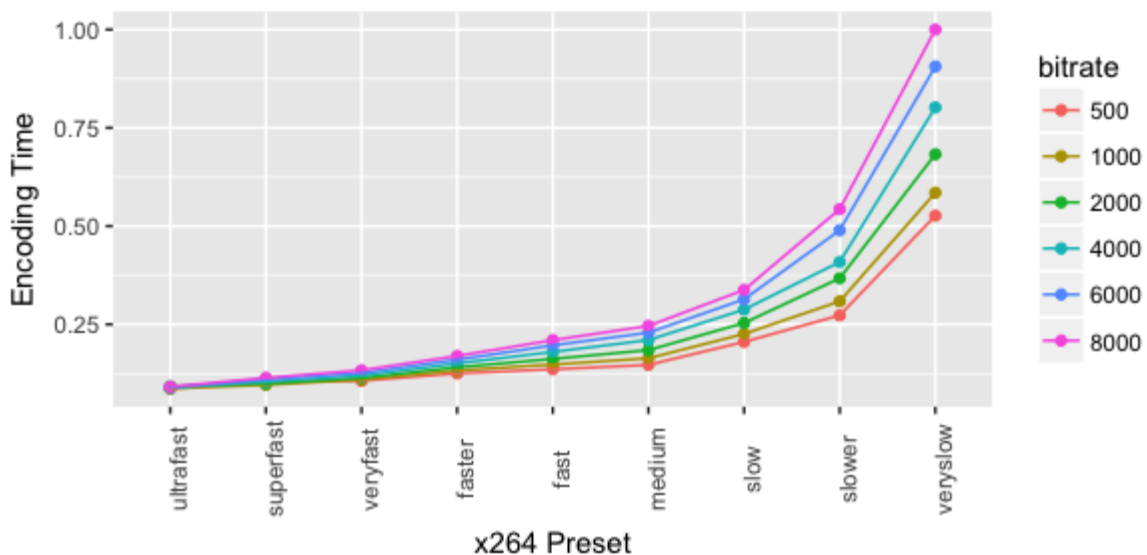


Abbildung 3: Kodierungszeitbeispiel eines 1080p-Videos

Die Kodierungszeit hängt von der Qualität des Quellmaterials, von der Ziel-Bitrate sowie von der Hardwarekonfiguration ab. Grundsätzlich gilt, je höher die Bitrate, desto mehr Zeit wird beim Kodieren benötigt.

Von mittel bis langsam erhöht sich die benötigte Zeit um ca. 40 Prozent. Wenn man stattdessen langsamer kodiert, wird etwa 100 Prozent mehr Zeit benötigt (das heißt, es dauert doppelt so lange). Im Vergleich zu mittel, sehr langsam erfordert 280 Prozent der ursprünglichen Codierungszeit, mit nur minimalen Verbesserungen gegenüber langsamer in Bezug auf Qualität.

Mit schnell spart man etwa 10 Prozent der Kodierungszeit und bei schneller rund 25 Prozent. Bei ultraschnell spart man rund 55 Prozent, jedoch mit einer sehr viel geringeren Qualität.

<https://trac.ffmpeg.org/wiki/Encode/H.264#Listpresetsandtunes>

Parameter	Description	Example Value
b:v	Video Bitrate	150k
b:a	Audio Bitrate	192k
ac	Set Number of Audio Channels	2
crf	Constant Rate Factor	23
qscale	Constant Quality Scale. {1,30}	20
prefwidth	Preferred Width	640
prefheight	Preferred Height	480
vcodec	Video Codec	libx264
acodec	Audio Codec	libfaac
preset	Provides a certain encoding speed to compression ratio	medium
threads	Set the Thread Count	2
b	Set Bitrate (Output, Audio, Video)	200k
vpre	Loads vpre preset file (Legacy)	
fpre	Loads fpre preset file (Legacy)	
fps	Desired Output Framerate	30
ab	Audio Bitrate	192k
ar	Set Audio Sampling Rate (in Hz)	44100
setpts	Change the PTS (presentation timestamp) of the input video frames	10
scaledownonly	Prevents scaling width and height greater than original video size	true

<https://entermediadb.org/knowledge/9/avconvffmpeg-parameters/>

NGINX Konfiguration mit ffmpeg hineintun

8.4 Snapd

Bei Snapp Apps (snaps) handelt es sich um ein sogenanntes Paketformat, welches zusätzlich zur normalen Paketverwaltung installiert und verwendet werden kann. Dieses Paketformat war ursprünglich für das Internet of Things und für den Einsatz auf Servern und Clouddiensten entwickelt worden. Ein Vorteil von Snaps ist, dass sie bereits alle Dateien und Abhängigkeiten wie z.B. diverse Konfigurationsdateien und Bibliotheken mit sich bringen. Die Laufzeitumgebung der snaps bezeichnet man als Core. Für die Verwaltung der snaps wird der Hintergrunddienst snapd verwendet. Es können unterschiedliche snaps im Snap Store heruntergeladen werden.

Snapd wurde benötigt, da ffmpeg für CentOS nur bis zur Version 2 supportet wird. Doch mit Hilfe von Snapd konnte wir die aktuelle 4. Version von ffmpeg installieren.

Um Snaps verwenden zu können, müssen diese zuerst aktiviert werden. Da Snap im EPEL-Repository (Extra Packages for Enterprise Linux) verfügbar ist, kann dieses mit dem nachfolgenden Befehl hinzugefügt werden.

Befehl: **\$ sudo yum install epel-release**

Nun kann Snap mit dem Befehl **\$ sudo yum install snapd** installiert werden.

Nach der Installation muss die Systemeinheit aktiviert werden.

Befehl: **\$ sudo systemctl enable --now snapd.socket**

Im nächsten Schritt muss ein Support mit Hilfe einer symbolischen Verknüpfung zwischen /var/lib/snapd/snap und /snap erstellt werden.

Befehl: **\$ sudo ln -s /var/lib/snapd/snap /snap**

Nachdem snap richtig installiert wurde muss das System neu gestartet werden, damit die Pfade von snap aktualisiert werden. Nach dem Neustart kann nun mit Hilfe von snap FFmpeg installiert werden.

Befehl: **\$ sudo snap install ffmpeg**

<https://wiki.ubuntuusers.de/snap/>

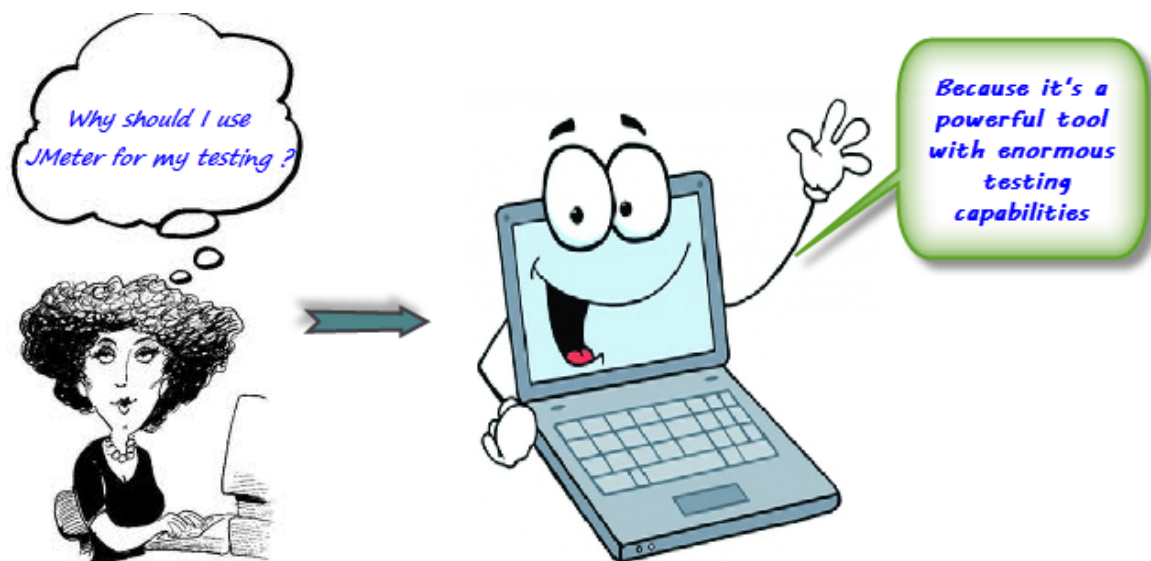
<https://snapcraft.io/install/ffmpeg/centos>

8.5 Apache

Apache wird verwendet um den Stromverbrauch hinsichtlich statischer bzw. dynamischer Webseiten verwendet. Da das Projektteam auf die dynamische Webseite auf Apache getestet hat wurde eine Virtuelle Maschine mit

8.6 JMeter

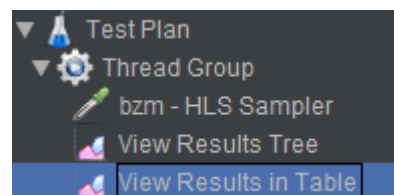
Apache JMeter ist eine Java Open-Source-Software. Mit Hilfe von JMeter können Leistungstests von Webanwendungen sowie Analysen von zahlreichen Diensten durchgeführt werden. Weiters ist es möglich einen Lastentest von mehreren gleichzeitigen Benutzerzugriffen und diverse andere Einstellungen durchzuführen, um beispielsweise 1000 http Requests auf einen Streaming-Server oder eine Website zu testen. Weitere mögliche Requests sind FTP Requests, JUnit Requests, Java oder HTTP2 Requests.



Zum Testen von Webapplikationen wird zunächst ein Testplan erstellt. Dieser besteht immer aus einem Threadgroup und diversen Samplern. Im Threadgroup lassen sich die Anzahl an Threads, die Ramp-up Period und die Loop Count definieren. Die Anzahl an Threads ist die Anzahl an Usern die später auf die definierte Webseite zugreifen wird. Die Ramp-up Period bestimmt die Dauer bis alle User aktiv sind. Ein Threadgroup von 30 Usern mit einer Ramp-up Period von 120 würde also alle vier Sekunden einen neuen Thread starten. Dieses Setting ist vor allem dann nützlich, wenn sehr viele Threads auf die Webseite zugreifen, die den Webserver ohne Ramp-up überlasten würden. Die Art des Zugriffes auf

den Webserver wird durch den Sampler definiert. Ein einfacher GET request wird über einen HTTP Request Sampler gesendet. Da es sich in unserem Fall um HLS handelt, benötigen wir einen HLS Sampler. Dafür muss der Jmeter Plugins Manager installiert werden, ein JAR File welches in den lib/ext Folder kopiert wird. Im Plugins Manager lässt sich dann der von Blazemeter entwickelte HLS Sampler installieren. Diesen hängen wir an unseren Testplan und geben ihm den Pfad zum HLS Playlist File (.m3u8).

Zuletzt finalisieren wir den Testplan mit Listnern, mittels welchen wir die Resultate des Tests sehen können.



Beim Arbeiten mit HLS Samplern ist der die große menge an RAM zu beachten. Die Standardeinstellung für Jmeter sind nur 512 MB RAM. Um mehr RAM für die Java Applikation zu reservieren muss im Launchscript (jmeter.bat oder jmeter.sh) die Java Heap Size vergrößert werden. Bei einer Threadgroup Größe von 500 Usern werden zirka 20GB benötigt:

```
Set HEAP= -Xms1g -Xmx20g;
```

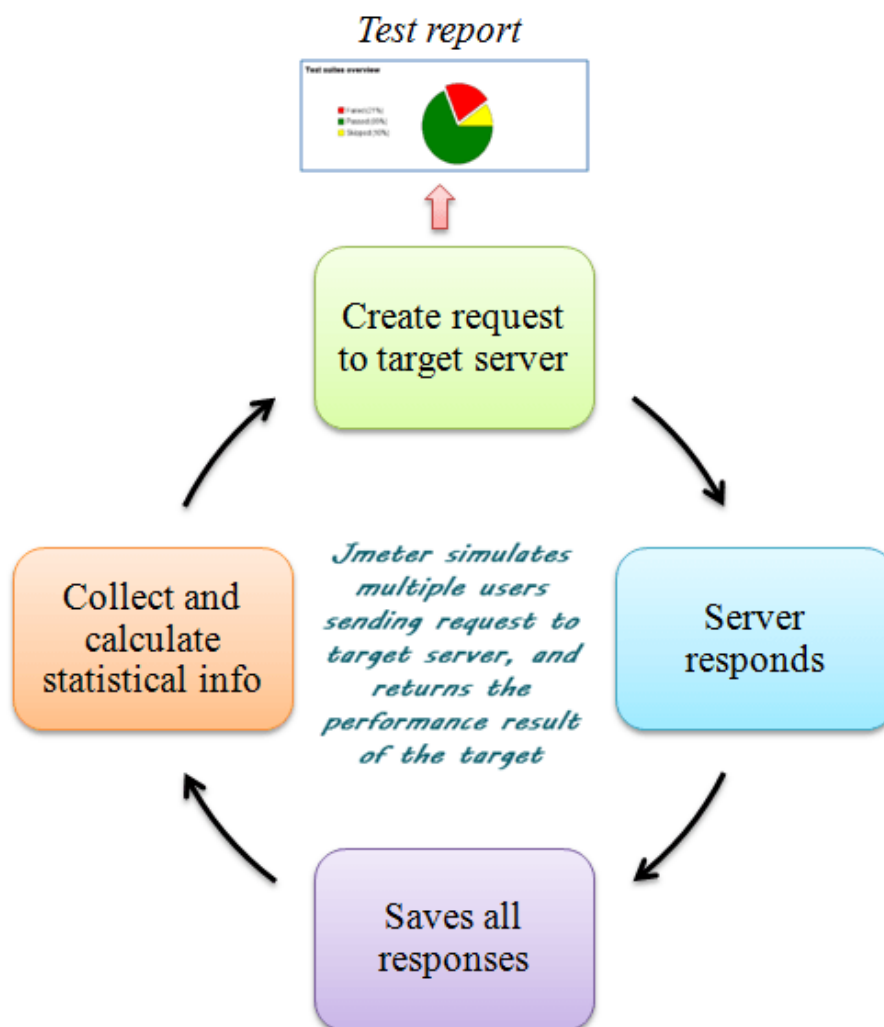
Im Listener view Result in Table werden beim Ausführen den Testplans die empfangenen Responses aufgelistet. Bei HLS video sind die zum einen GET requests für das Playlist File

(media playlist) und GET requests für die im Playlistfile verwiesenen HLS Chunks (media segment).

42	17:30:32.923	Thread Group 1-41	bzm - HLS Sampler - media playlist	23	✓	449
43	17:30:32.045	Thread Group 1-42	bzm - HLS Sampler - media playlist	22	✓	449
44	17:30:32.064	Thread Group 1-43	bzm - HLS Sampler - media playlist	23	✓	449
45	17:30:32.085	Thread Group 1-44	bzm - HLS Sampler - media playlist	25	✓	449
46	17:30:32.105	Thread Group 1-45	bzm - HLS Sampler - media playlist	22	✓	449
47	17:30:32.126	Thread Group 1-46	bzm - HLS Sampler - media playlist	23	✓	449
48	17:30:32.144	Thread Group 1-47	bzm - HLS Sampler - media playlist	24	✓	449
49	17:30:32.164	Thread Group 1-48	bzm - HLS Sampler - media playlist	35	✓	449
50	17:30:32.200	Thread Group 1-49	bzm - HLS Sampler - media playlist	28	✓	449
51	17:30:32.204	Thread Group 1-50	bzm - HLS Sampler - media playlist	24	✓	449
52	17:30:31.282	Thread Group 1-3	bzm - HLS Sampler - media segment	1225	✓	9414582
53	17:30:31.793	Thread Group 1-2	bzm - HLS Sampler - media segment	1065	✓	8311210
54	17:30:31.373	Thread Group 1-8	bzm - HLS Sampler - media segment	1859	✓	9414582
55	17:30:31.458	Thread Group 1-12	bzm - HLS Sampler - media segment	1855	✓	9414582
56	17:30:31.438	Thread Group 1-11	bzm - HLS Sampler - media segment	1929	✓	9414582
57	17:30:31.294	Thread Group 1-4	bzm - HLS Sampler - media segment	2130	✓	9414582
58	17:30:31.413	Thread Group 1-10	bzm - HLS Sampler - media segment	2103	✓	9414582
59	17:30:31.395	Thread Group 1-9	bzm - HLS Sampler - media segment	2261	✓	9414582

https://jmeter.apache.org/usermanual/test_plan.html

<https://www.blazemeter.com/blog/the-new-hls-plugin-for-jmeter-the-complete-guide/>

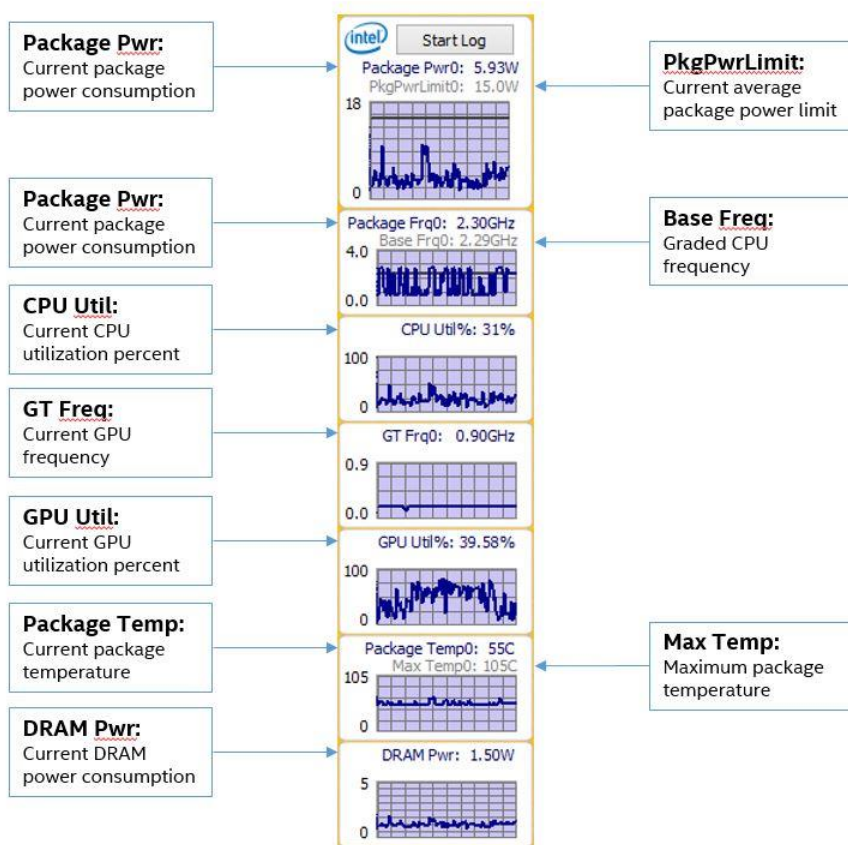


<https://www.guru99.com/introduction-to-jmeter.html>

8.7 Intel Power Gadget

Beim Intel Power Gadget handelt es sich um ein softwarebasiertes Tool von Intel. Es dient zur Überwachung des Stromverbrauchs der Intel Prozessoren und unterstützt die Generationen zwei bis zehn. Dabei werden in Echtzeit mithilfe des Energiezählers im Prozessor der Stromverbrauch in Watt ermittelt. Das Power Gadget unterstützt nur Core und keine Atom Prozessoren von Intel. Betriebssysteme wie Windows, MacOS sowie auch Linux unterstützen dieses Tool. Diverse Plattformen wie Notebooks, Desktops und Server unterstützen das Power Gadget jedoch nur am Parent und nicht in virtuellen Umgebungen.

Mit Hilfe des Start Log Buttons kann der Energieverbrauch für einen zuvor definierten Zeitraum aufgenommen werden. Dabei wird eine CSV-Datei mit den einzeln gemessenen Komponenten erstellt.



<https://software.intel.com/en-us/articles/intel-power-gadget>

8.8 HP ILO

Beim HP ILO (Integrated Lights-Out) handelt es sich um ein proprietäres Management-System zur Administration von HP Servern. In diesem Fall handelt es sich um das ILO2. Die

Firmware zur Konfiguration des ILOs wird beim Booten des Servers beim Drücken der F8 taste aufgerufen. Hier lässt sich das ILO aktivieren, User anlegen und die Schnittstelle genauer konfigurieren. Das ILO wird entweder über die eigens dafür verbaute Ethernet Port mit dedizierter NIC angesprochen oder als "shared Ethernet Port" über das Ethernet Port des Motherboards.

Ist das ILO einmal aktiviert stellt es ein Webinterface zur Verfügung. Das Interface des ILO2 wird von vielen derzeitigen Browsern nicht mehr unterstützt, wir konnten es nur mit Internet Explorer und Firefox (dank exzellenter Rückwärtskompatibilität) aufrufen. Das Webinterface bietet einiges an Funktionalität mit sich, die für unser Projekt interessanten Powermonitoring Tools befinden sich allerdings hinter einer weiteren Paywall in der Form einer Advanced License Key. Glücklicherweise können diese wiederverwendet werden, Zugriff zu den Erweiterten Features kann man daher durch eine einfache Google Suche erlangen. Im Webinterface lässt sich nun ein 24 Stunden Rückblick über den Stromverbrauch ausgeben.

Zusätzlich gibt es eine Python API, über welche der Stromverbrauch ebenfalls ausgelesen werden kann. Für den Zugriff über die API-Schnittstelle wurde ein einfaches Python-Skript geschrieben, welches den aktuellen Stromverbrauch im Sekundentakt ausgibt. Da sich der Wert im ILO nicht oft ändert und es in der Dokumentation des ILO2 nicht erläutert wird wurde das ILO nicht für Messungen herangezogen. Es ermöglichte allerdings die Genauigkeit der smarten Steckdosen zu überprüfen.

8.9 Loadtest

Hierbei handelt es sich um ein Einfaches, auf Linux basierendes, Datengenerierungs-Tool, welches durch einen einfachen Befehl hundert bis tausend http Requests generieren kann.

<https://www.npmjs.com/package/loadtest>

Bsp.: `loadtest -c 10 --rps 200 http://mysite.com/`

- -c steht für concurrency
- -rps steht für requests per second

XAMPP

XCP-ng Center

9 Messungstools

- Smart Meter
- Intel Power Gadget
- ILO-Schnittstelle



10 Messergebnisse

Bei den Messungen wurden unterschiedliche Testszenarien durchgeführt. Beim ersten Szenario wurden mit ffmpeg ein Userzugriff simuliert. Im zweiten Szenario versuchten wir mit Hilfe von JMeter Traffic zu generieren, um Multiuserzugriffe auf den Streamingserver zu simulieren. Im dritten Testszenario beziehen wir uns auf die Webseitenmessungen des Partnerprojektteams.

Die Prozessor Power (CPU) des Intel Powergadget Tools errechnet sich aus der IA Energy (Energie der CPU Kerne), der GT Energy (Energie der Prozessorgrafikkarte) und andere Komponenten.

Bei der Mittelwertberechnung werden die ersten vier Sekunden und die letzten vier Sekunden des Streams weggelassen, da beim Starten und Beenden die größten Fluktuationen (Spikes) entstehen.

Szenario 1 - ffmpeg

Bei Szenario eins wurden mit Hilfe des ffmpeg Befehls die Encoder Einstellungen im NGINX Konfigurationsfile für das Beispielvideo „Big Buck Bunny“ durchgeführt. Mit Hilfe von OBS Studio wurde das Video an den NGINX Server weitergeleitet. Dabei wurde der Energieverbrauch mit Hilfe des Intel Power Gadgets und des TP-Links für den HP Server ProLiant und den Desktop-PC G2 gemessen.

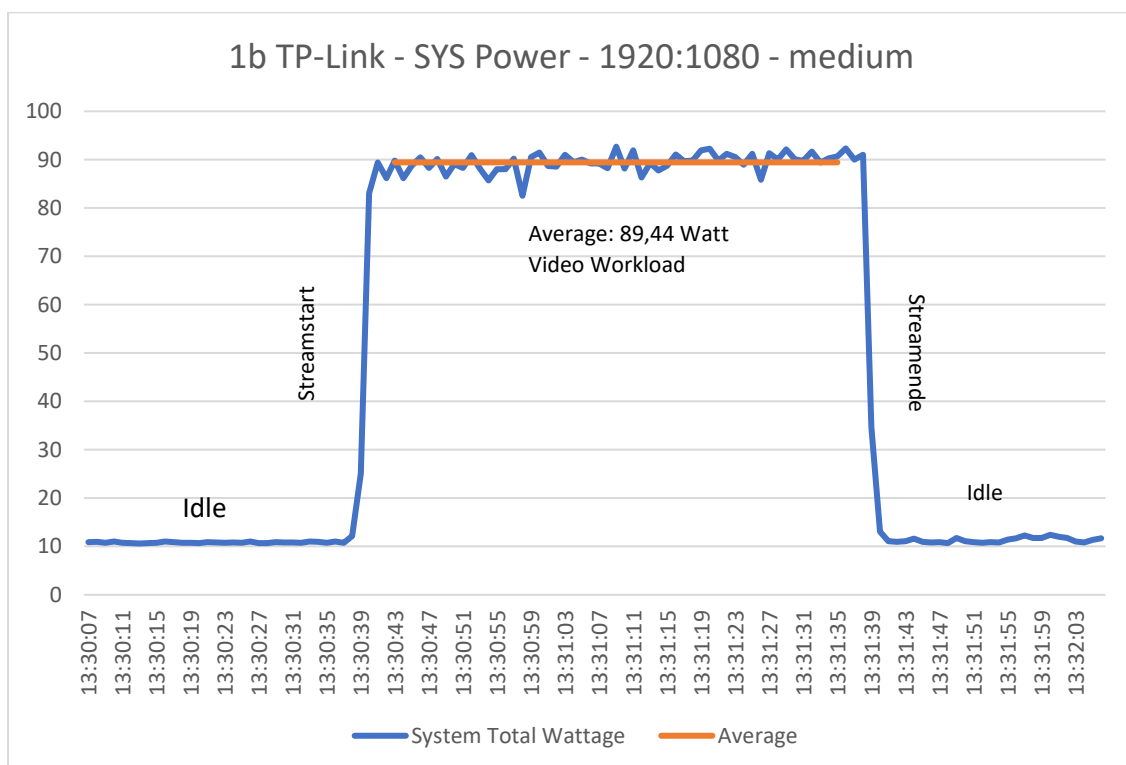
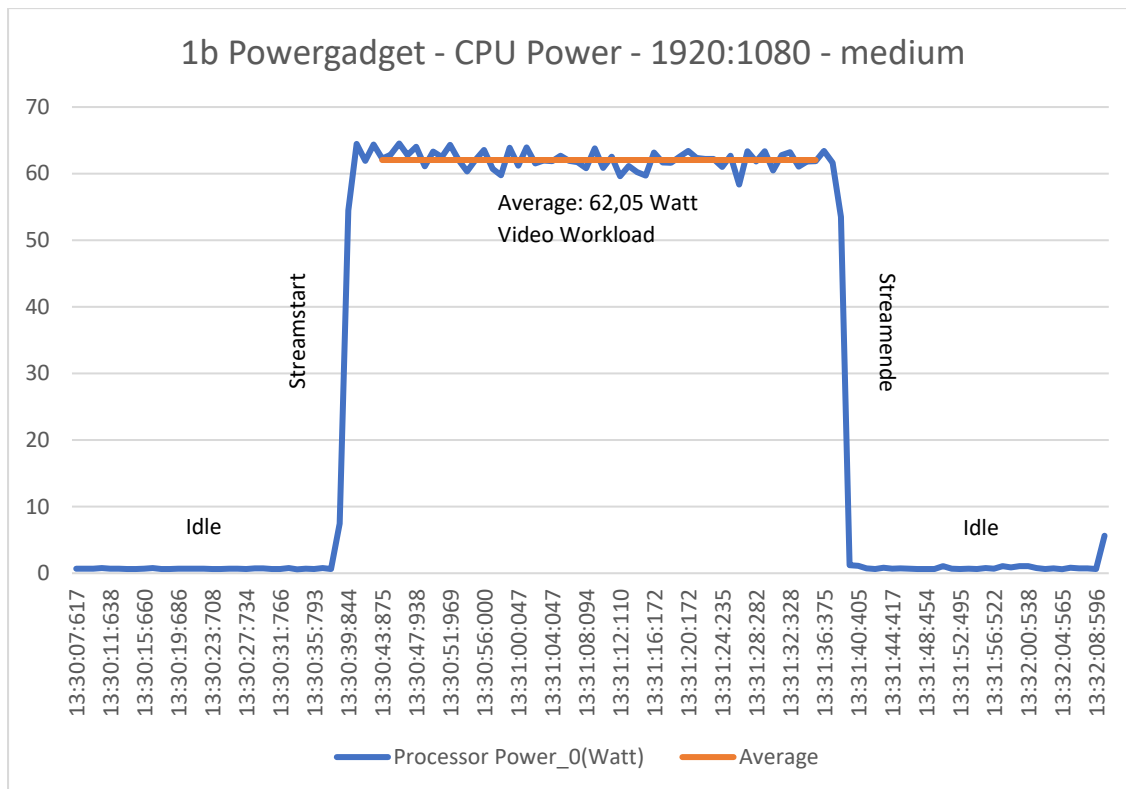
Die unterhalb ersichtliche Tabelle zeigt die durchgeführten Messungen am Desktop PC G2. Dabei wurden jeweils zwei Messungen (A und B) der gleichen Einstellung durchgeführt und bei der Auswertung die aussagekräftigeren Ergebnisse herangezogen. Die verwendeten Auflösungen sind 1920:1080 mit einer Bitrate von 6000 Kilobits pro Sekunde, 1280:720 mit einer Bitrate von 4500 Kilobits und 640:360 mit 3000 Kilobits pro Sekunde. Als

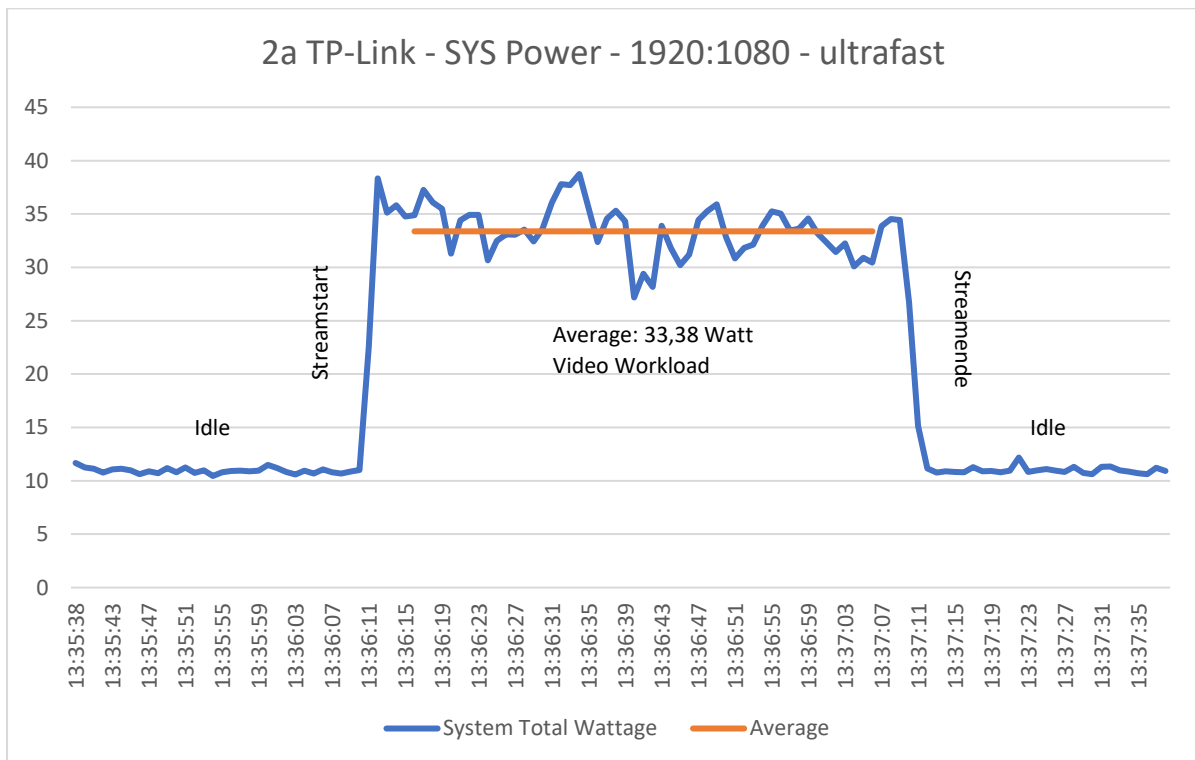
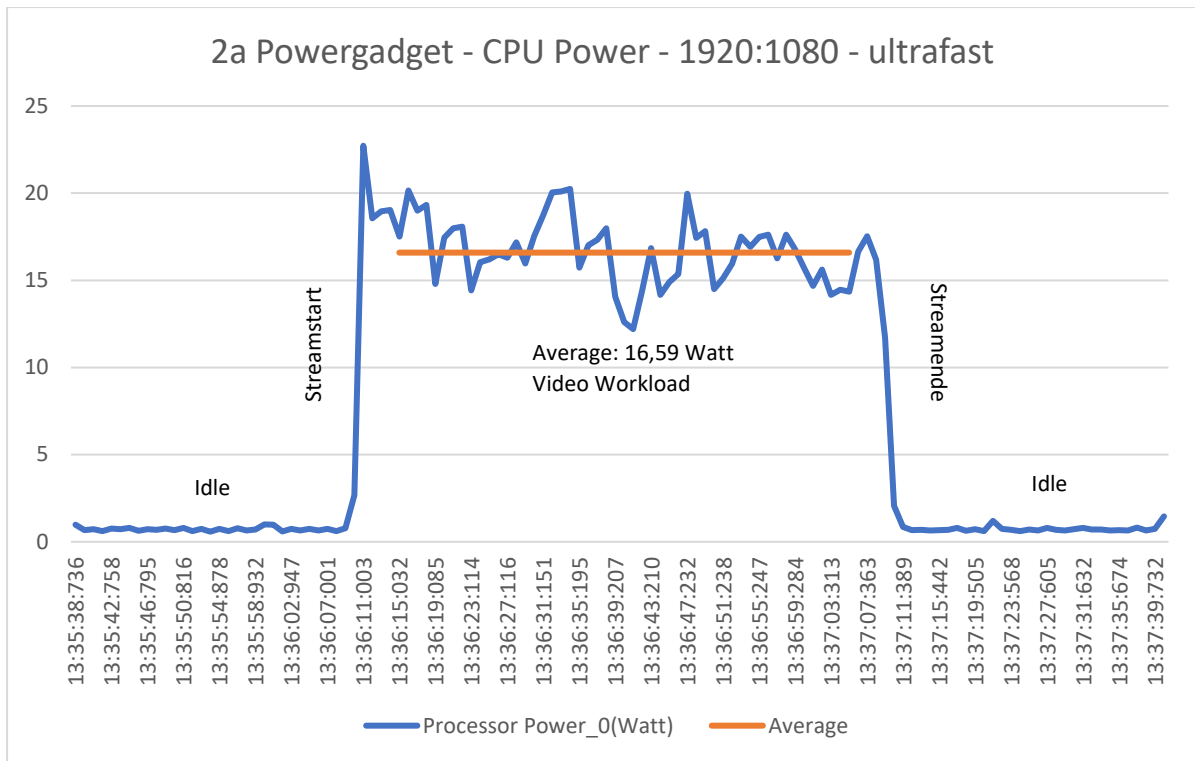
Orientierungshilfe für die zu verwendete Auflösung und Bitrate wurde Twitch herangezogen. Bei den diversen Auflösungen wurden jeweils drei Messungen mit unterschiedlicher Preset-Einstellung (medium, ultrafast, slower) durchgeführt. Preset wurde bereits in den vorderen Kapiteln erklärt.

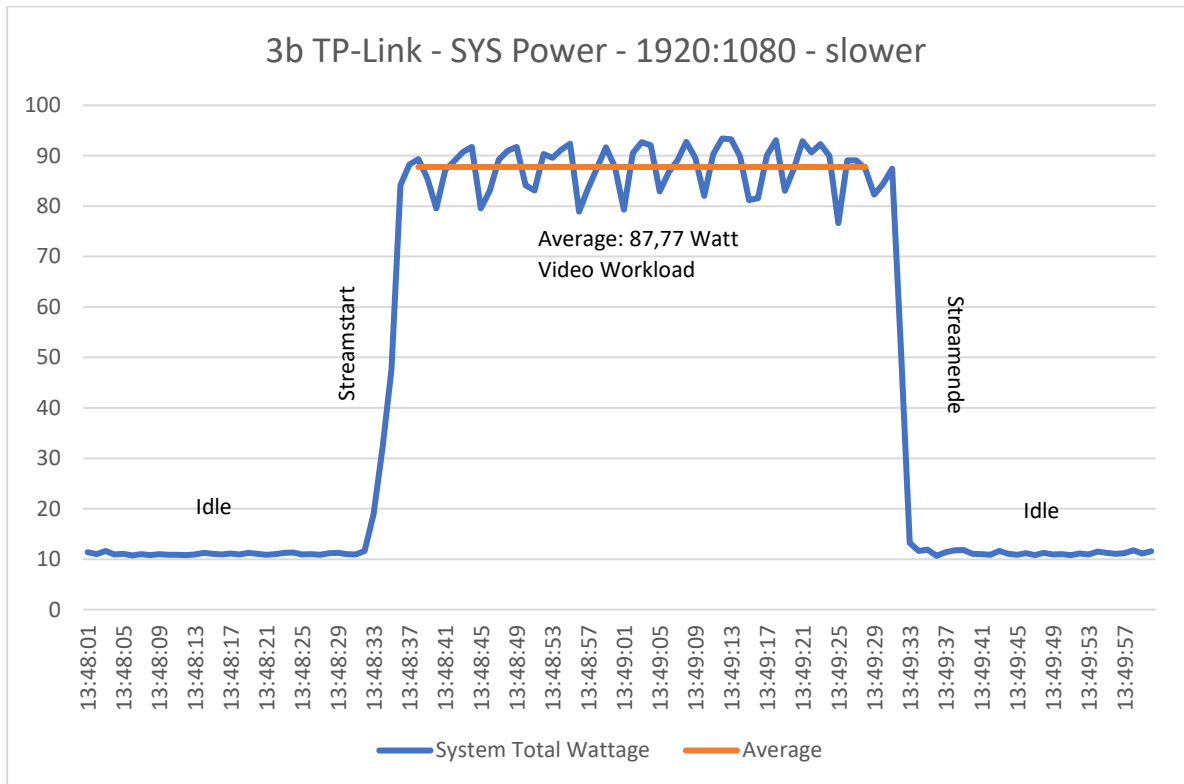
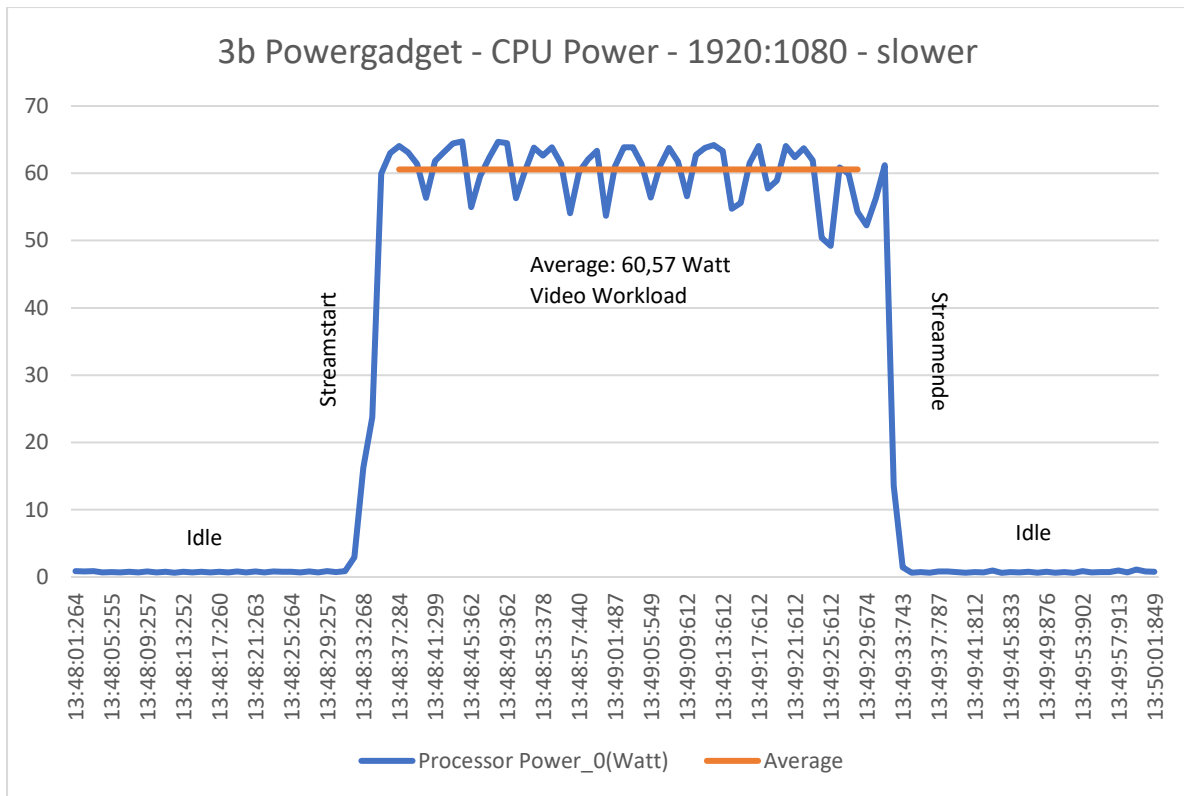
Die Messungen zeigten, dass es bei der Auflösung 1920:1080 nahezu keine großen Unterschiede bei der Preset Einstellung medium und slower gab, weder bei der System- noch bei der CPU Power. Bei der Preset Einstellung medium verbrauchte der PC durchschnittlich 89,44 Watt, davon 62,05 Watt die CPU. Bei der Einstellung slower verbrauchte der durchschnittlich 87,77 Watt und die CPU rund 60,57 Watt. Der größte Unterschied konnte jedoch beim Preset ultrafast ermittelt werden. Hierbei benötigte der der G2 gerade mal 33,38 Watt, davon 16,59 Watt die CPU.

Desktop PC G2								
-vf scale	-b:v	-b:v	FPS	-preset	libx264 -crf	SYS POWER [W] TP-Link	CPU POWER [W] Intel	Nr .
1920:1080	6000	CBR	60	medium	23	89.44 (1b)	62.05 (1b)	1
1920:1080	6000	CBR	60	ultrafast	23	33.38 (2a)	16.59 (2a)	2
1920:1080	6000	CBR	60	slower	23	87.77 (3b)	60.57 (3b)	3
1280:720	4500	CBR	60	medium	23	62.33 (4b)	40.36 (4b)	4
1280:720	4500	CBR	60	ultrafast	23	29.42 (5a)	13.91 (5a)	5
1280:720	4500	CBR	60	slower	23	86.94 (6b)	60.71 (6b)	6
640:360	3000	CBR	60	medium	23	31.85 (7b)	16.31 (7b)	7
640:360	3000	CBR	60	ultrafast	23	23.72 (8b)	9.83 (8b)	8
640:360	3000	CBR	60	slower	23	50.87 (9b)	31.92 (9b)	9

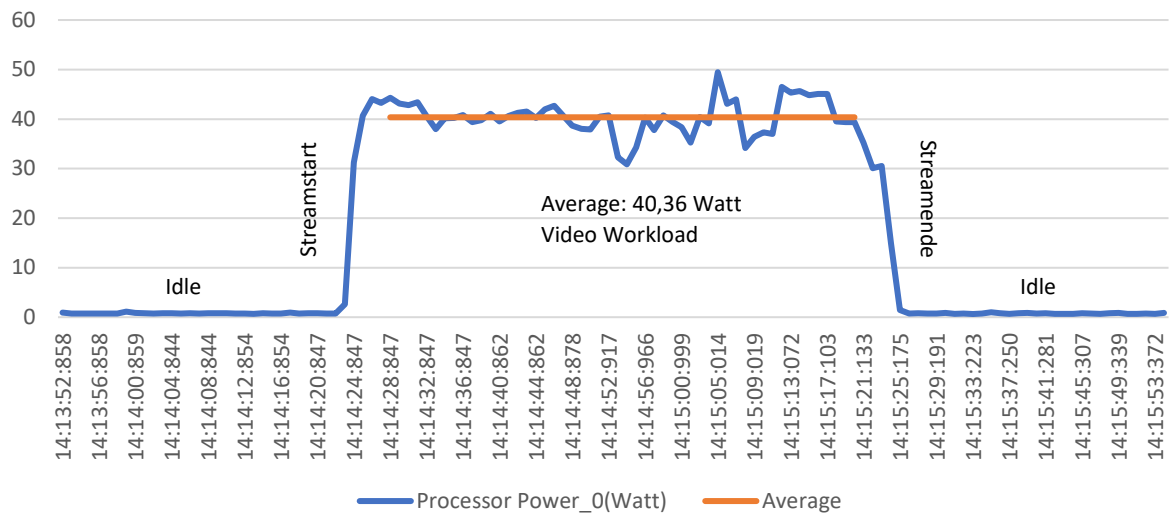
Nachfolgend wurden alle durchgeführten Messungen eingefügt.



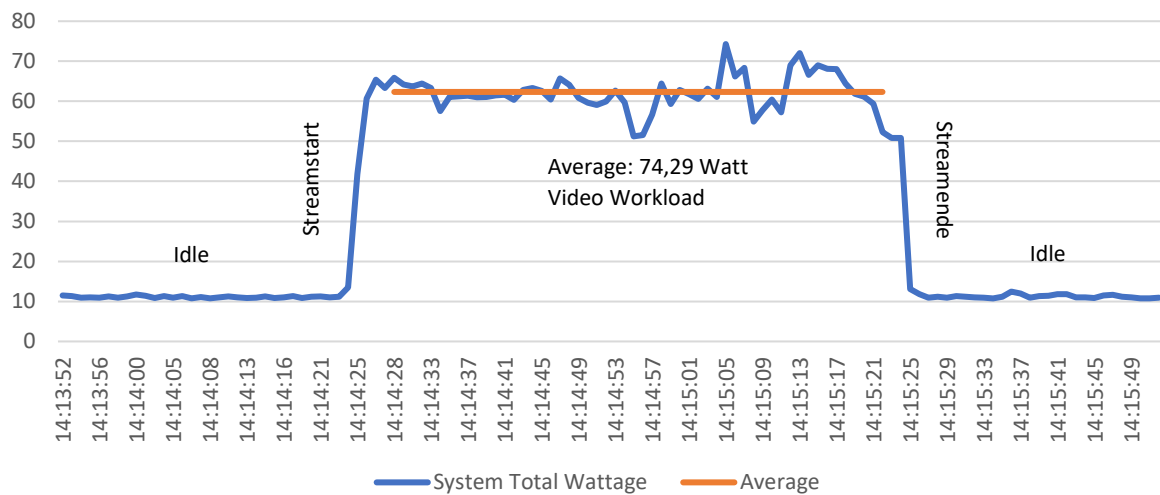


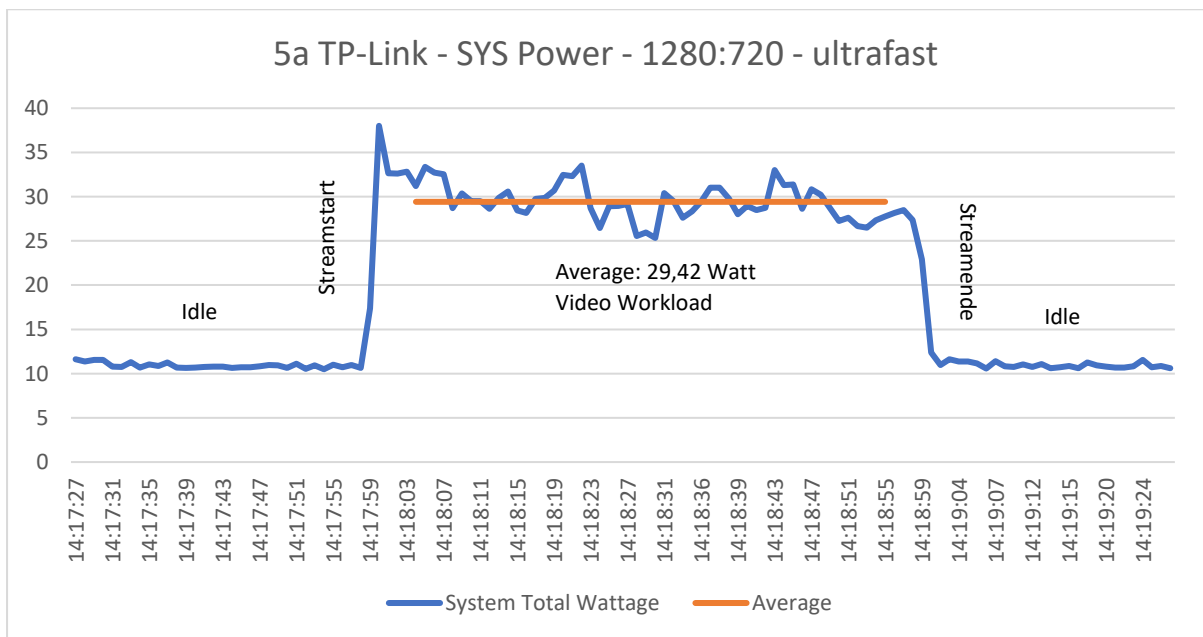
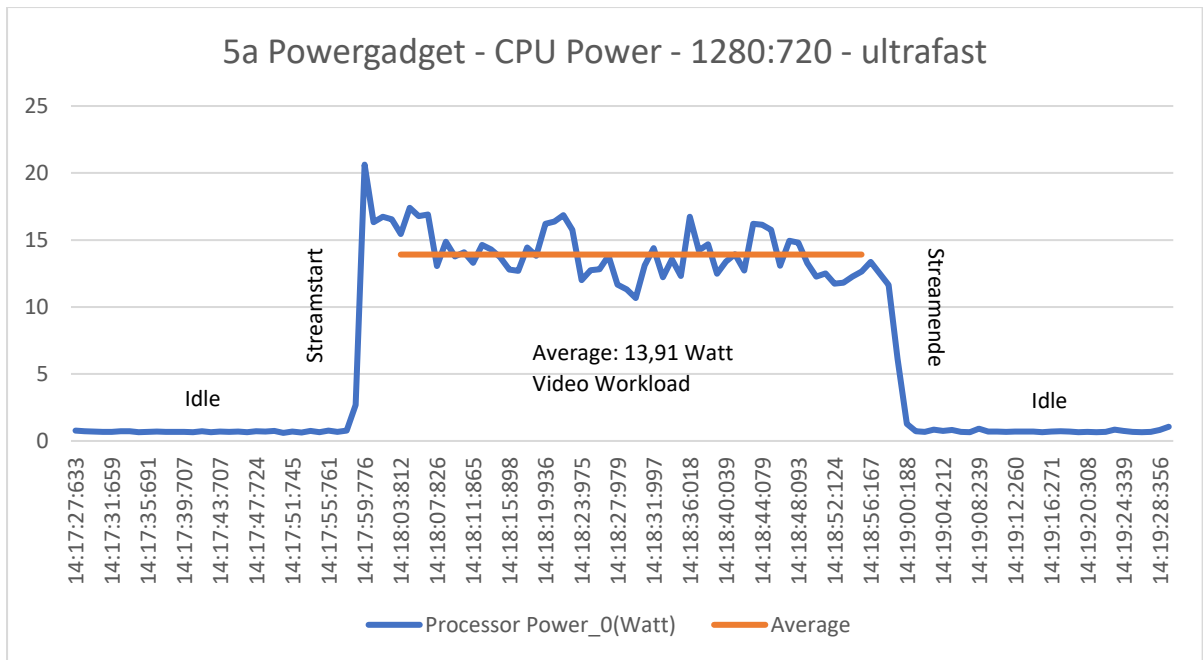


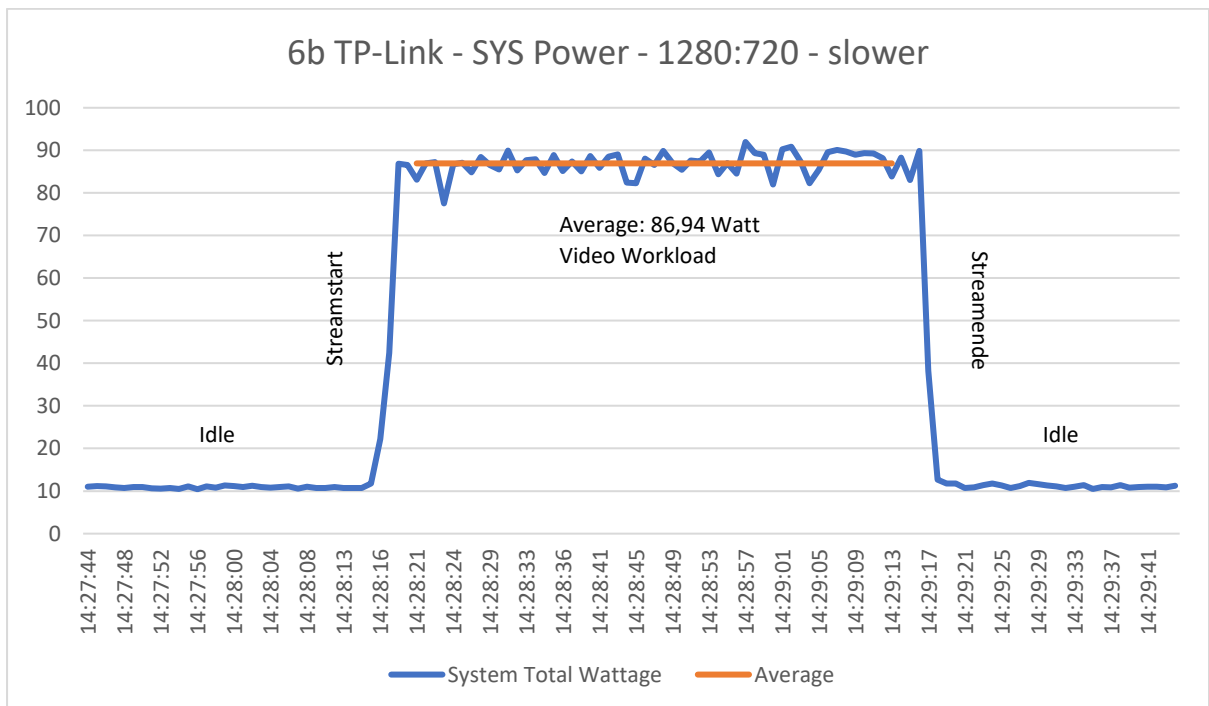
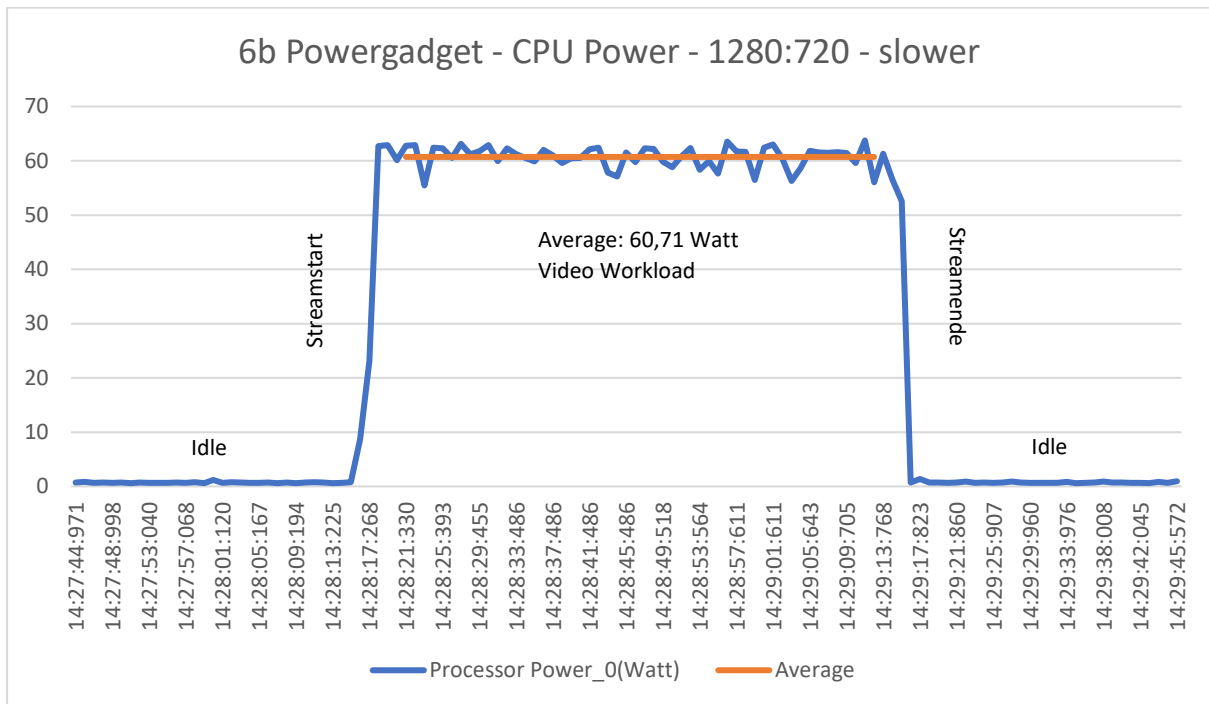
4b Powergadget - CPU Power - 1280:720 - medium

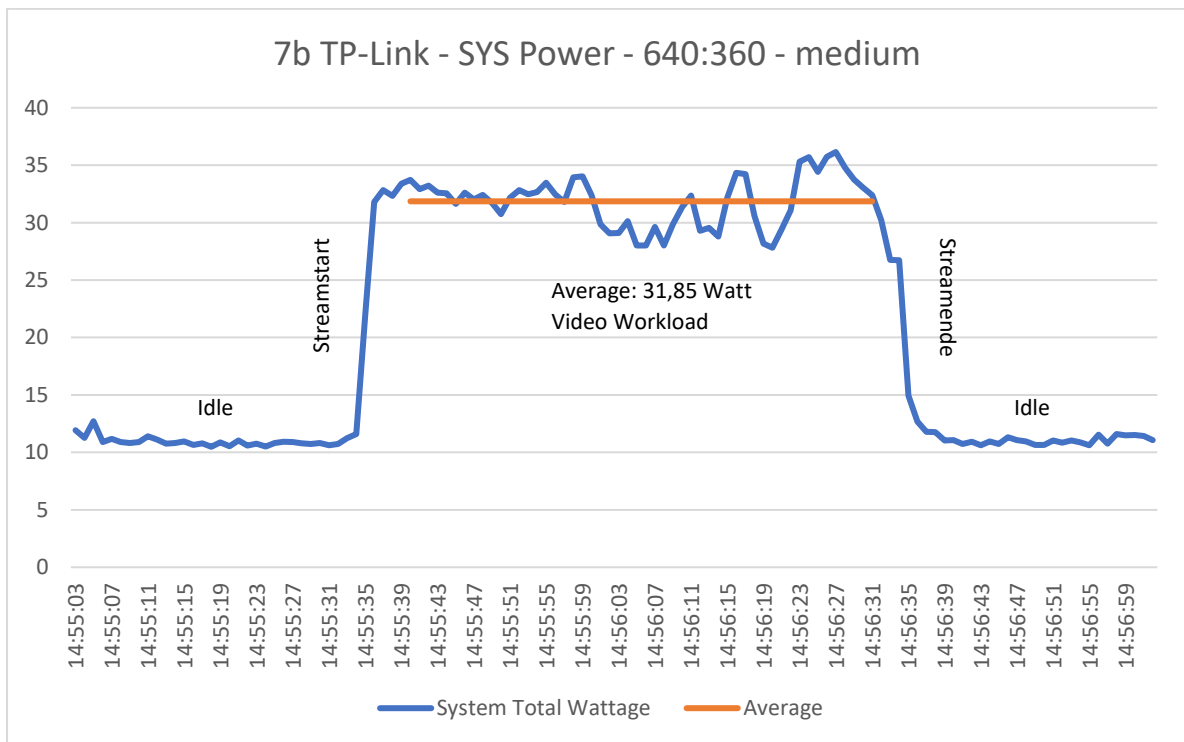
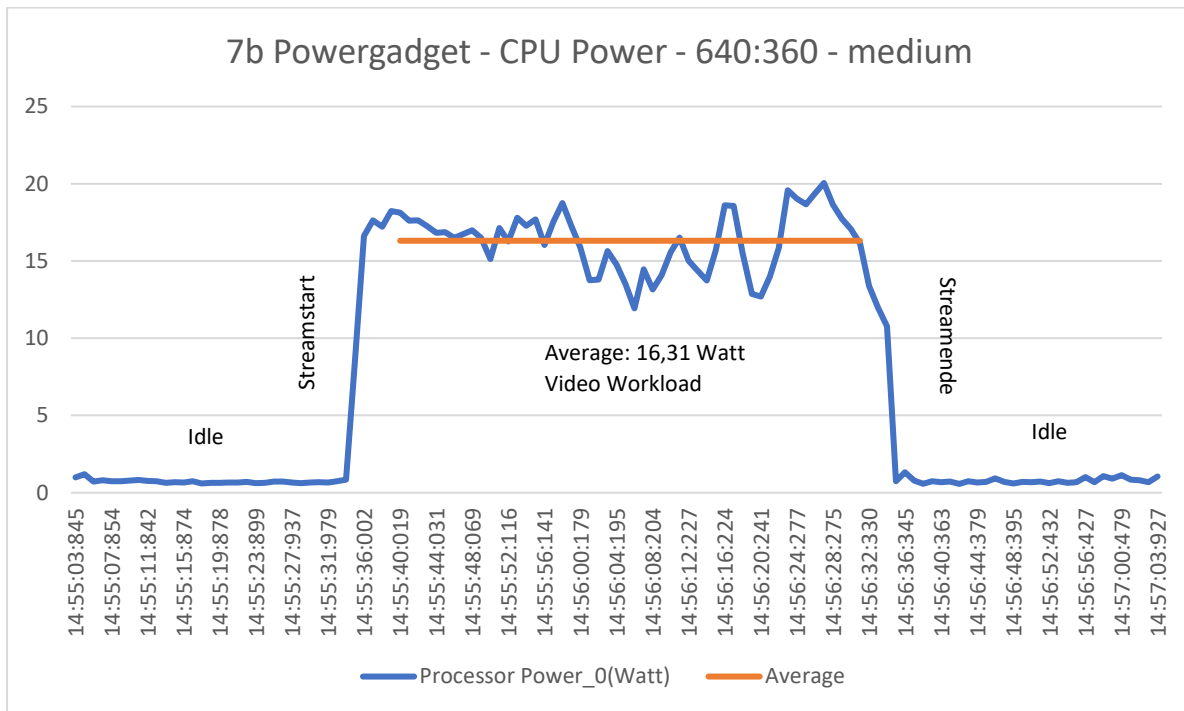


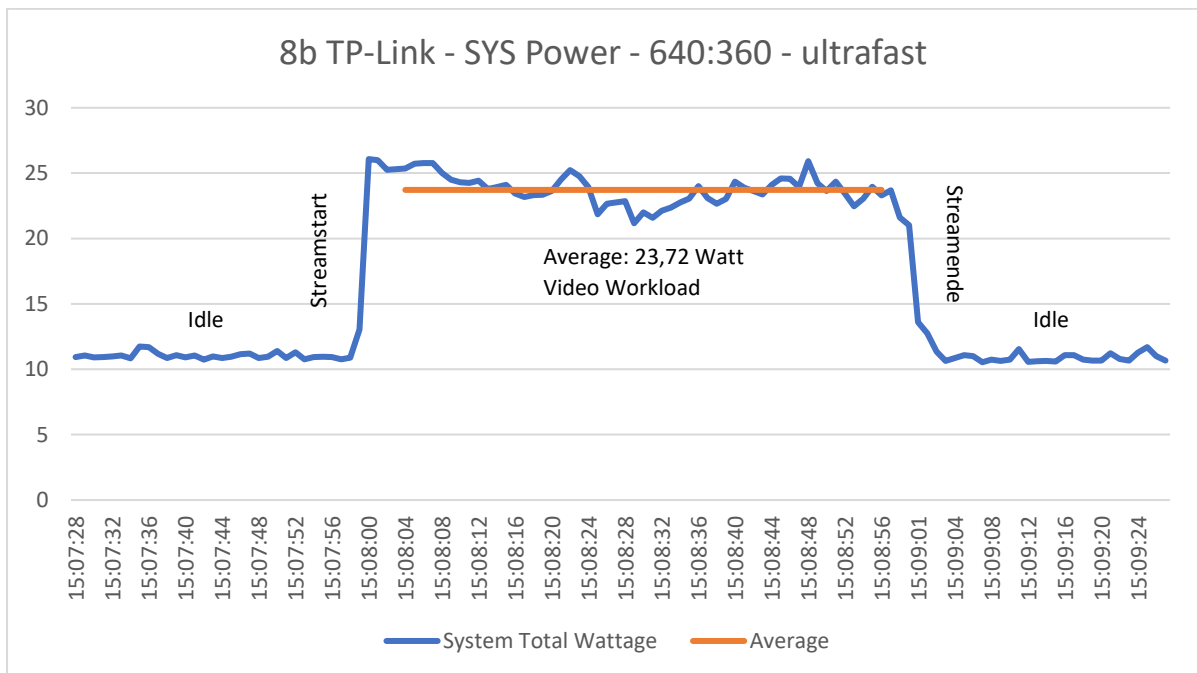
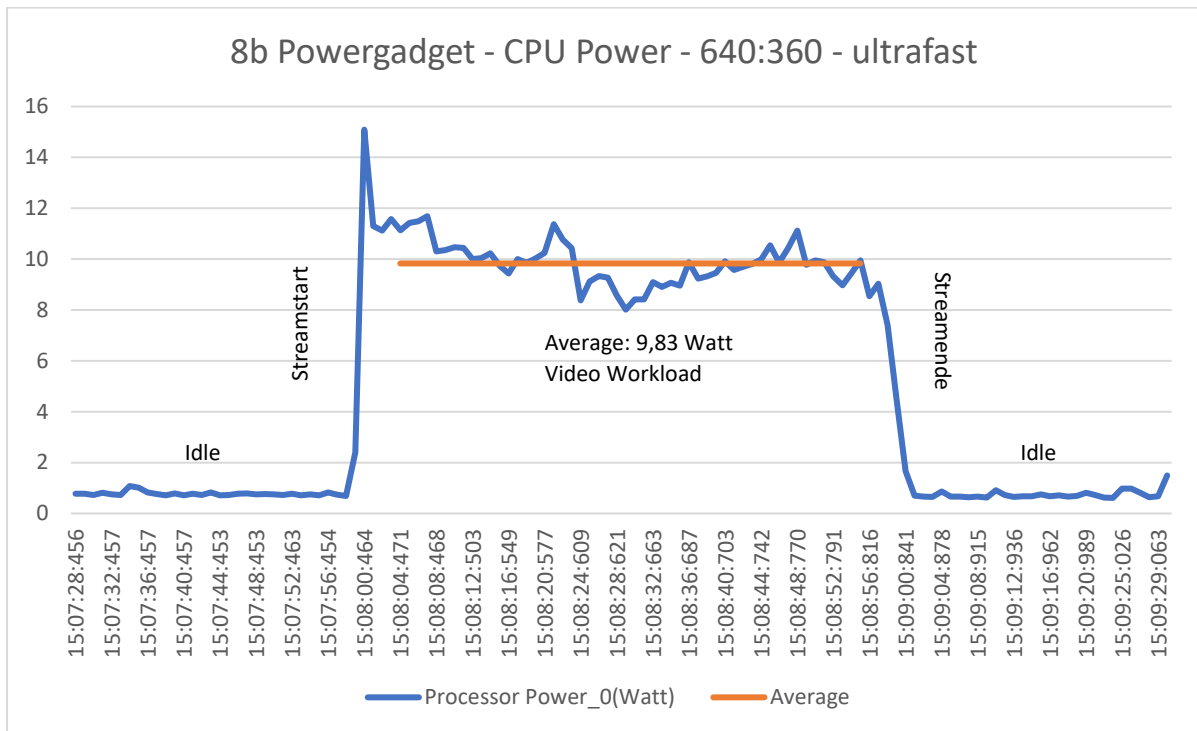
4b TP-Link - SYS Power - 1280:720 - medium

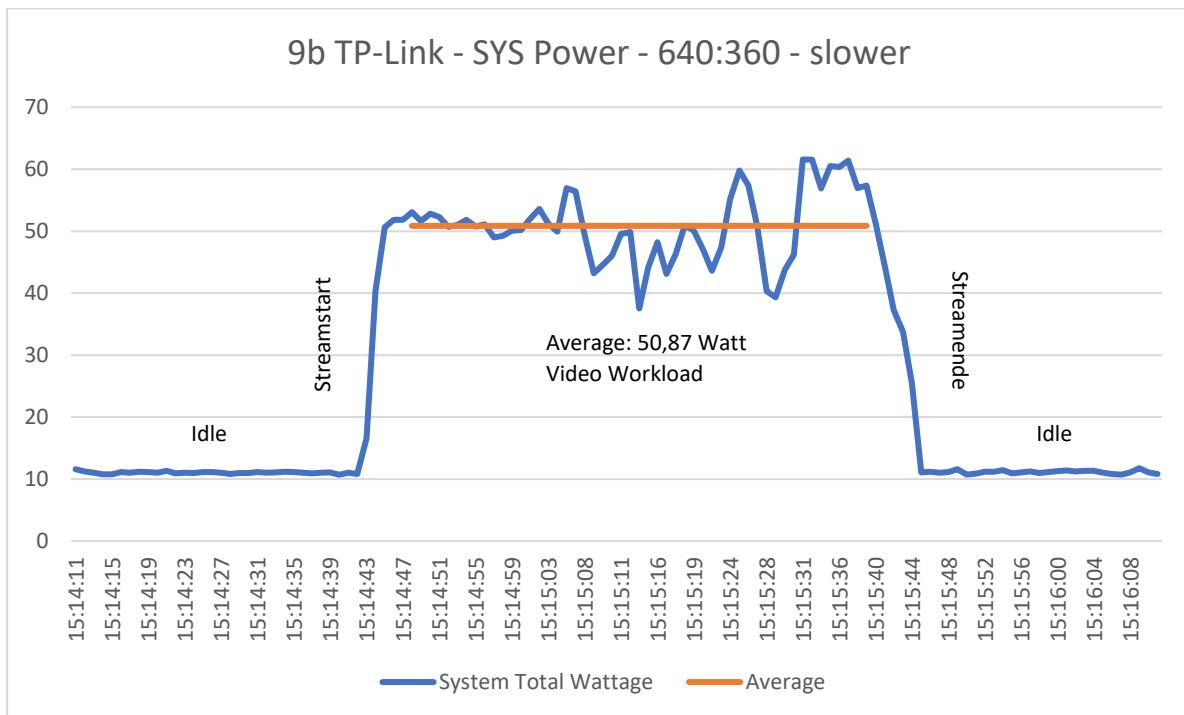
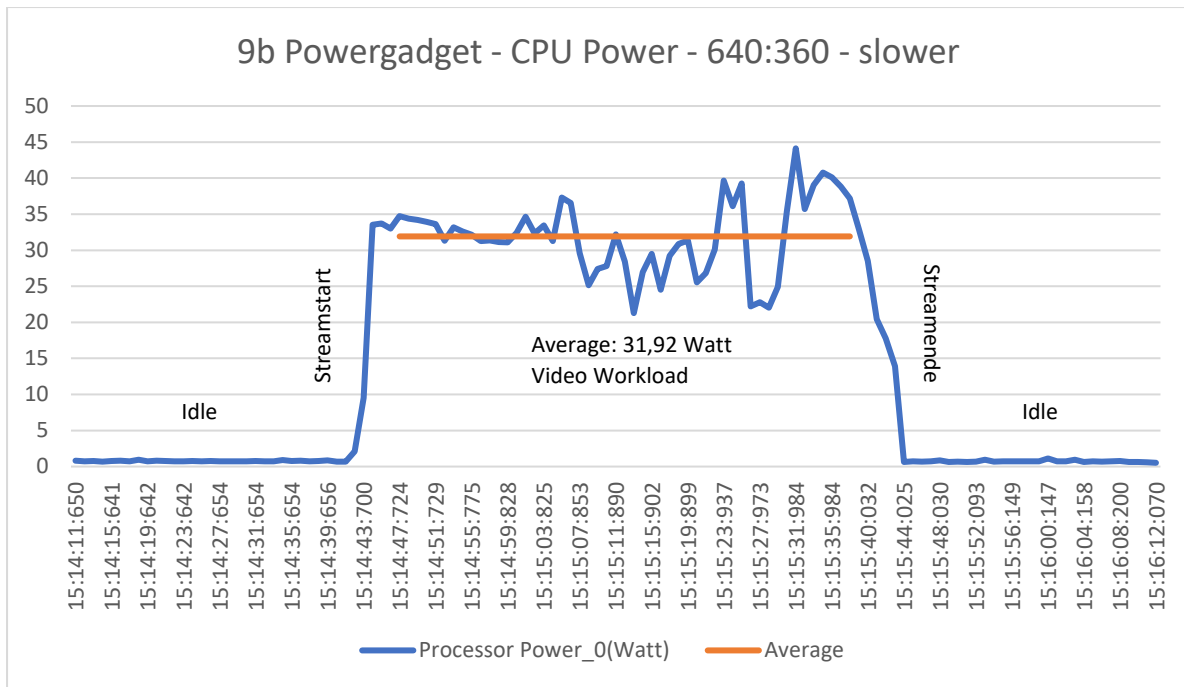




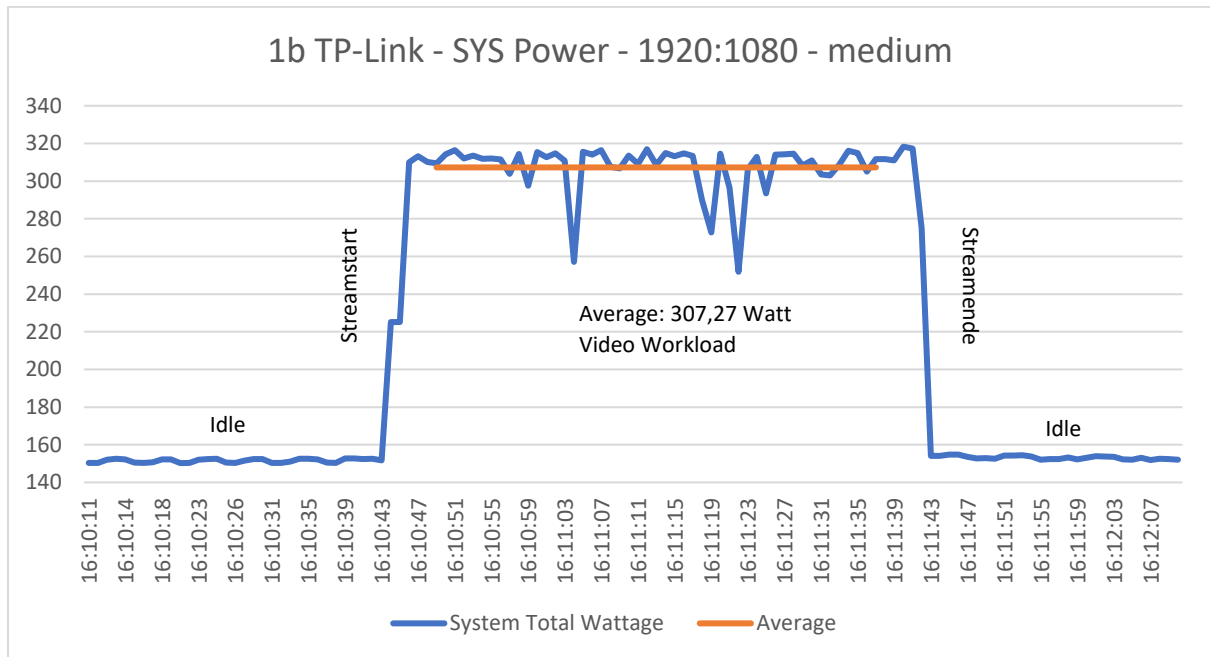




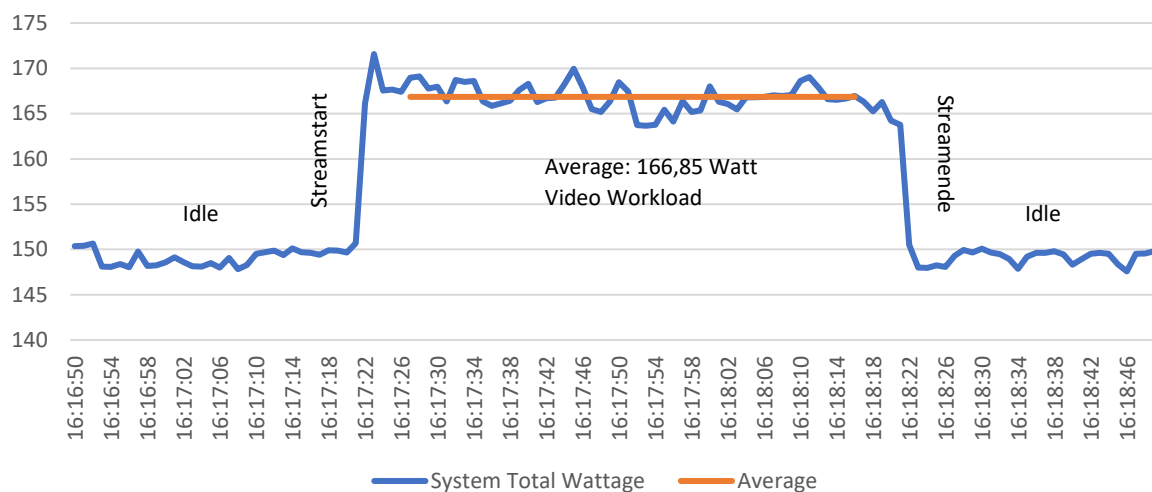




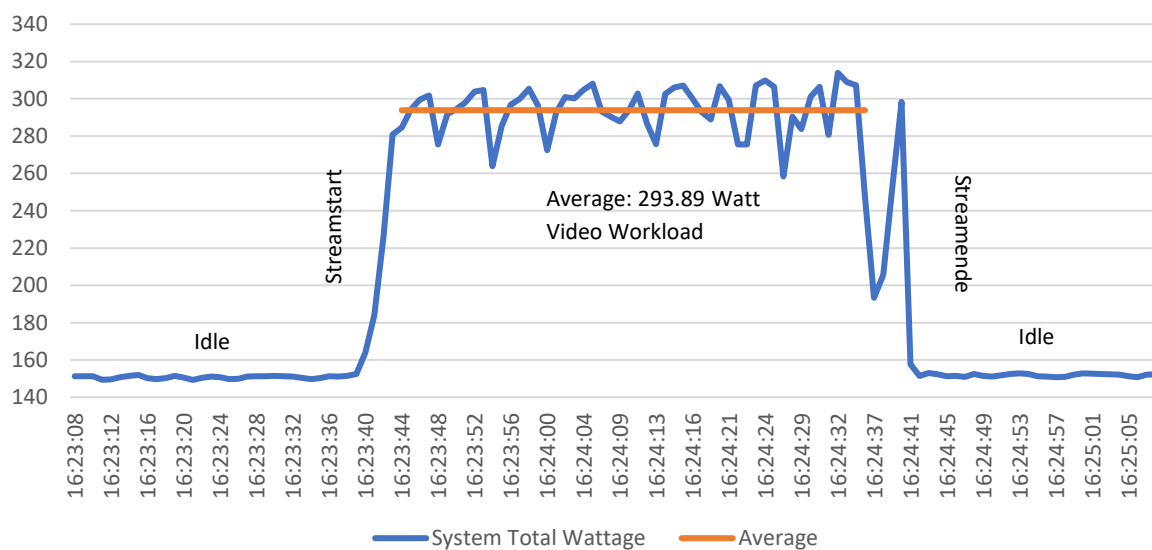
HP Server ProLiant G6							
-vf scale	-b:v	-b:v	FPS	-preset	libx264 -crf	SYS POWER [W] TP-Link	Nr.
1920:1080	6000	CBR	60	medium	23	307.27 (1b)	1
1920:1080	6000	CBR	60	ultrafast	23	166.85 (2b)	2
1920:1080	6000	CBR	60	slower	23	293.89 (3b)	3
1280:720	4500	CBR	60	medium	23	214.99 (4b)	4
1280:720	4500	CBR	60	ultrafast	23	164.07 (5b)	5
1280:720	4500	CBR	60	slower	23	283.33 (6b)	6
640:360	3000	CBR	60	medium	23	165.42 (7b)	7
640:360	3000	CBR	60	ultrafast	23	159.12 (8a)	8
640:360	3000	CBR	60	slower	23	186.82 (9b)	9



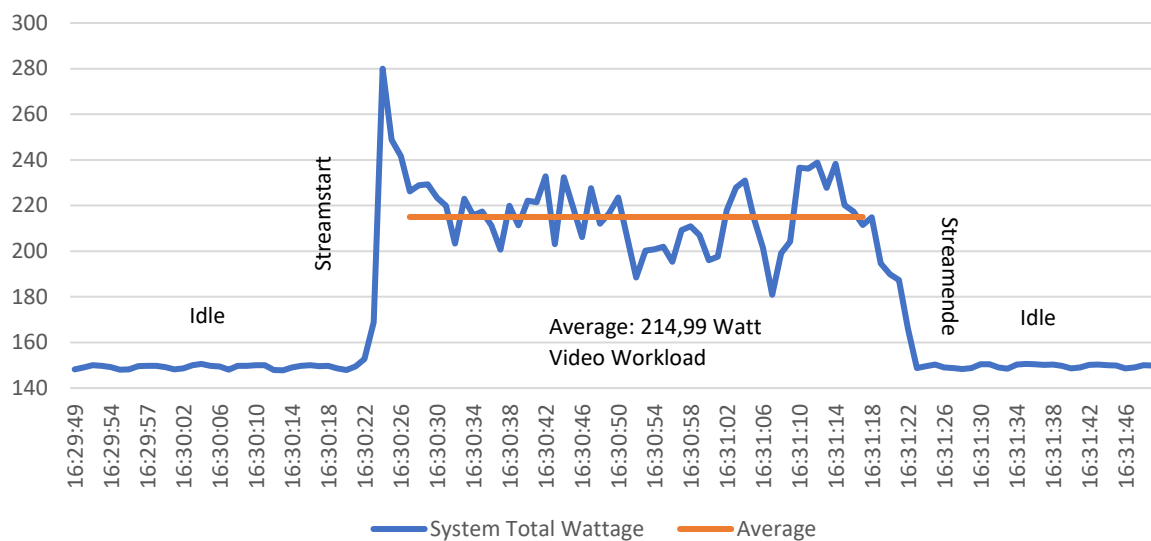
2b TP-Link - SYS Power - 1920:1080 - ultrafast



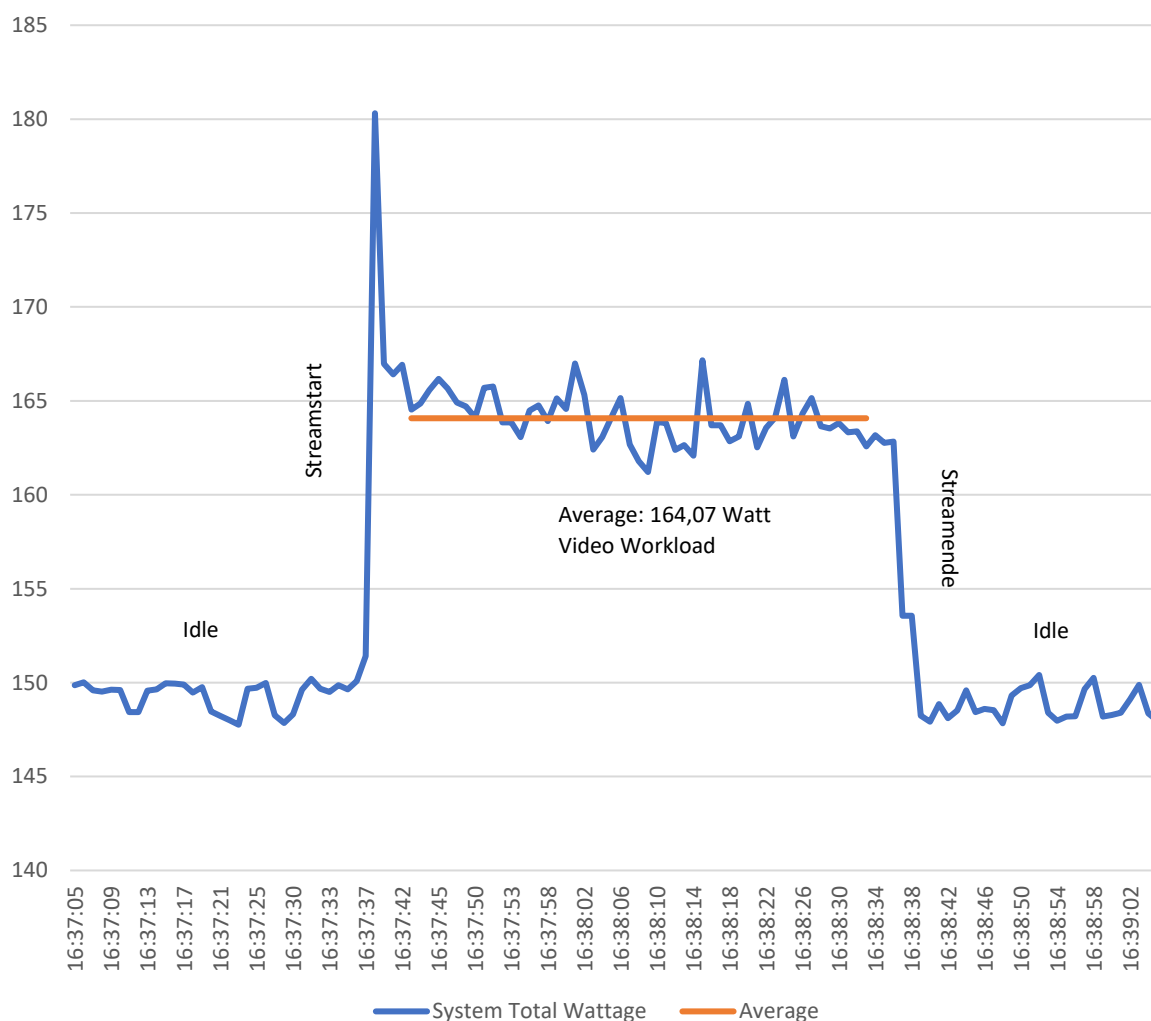
3b TP-Link - SYS Power - 1920:1080 - slower

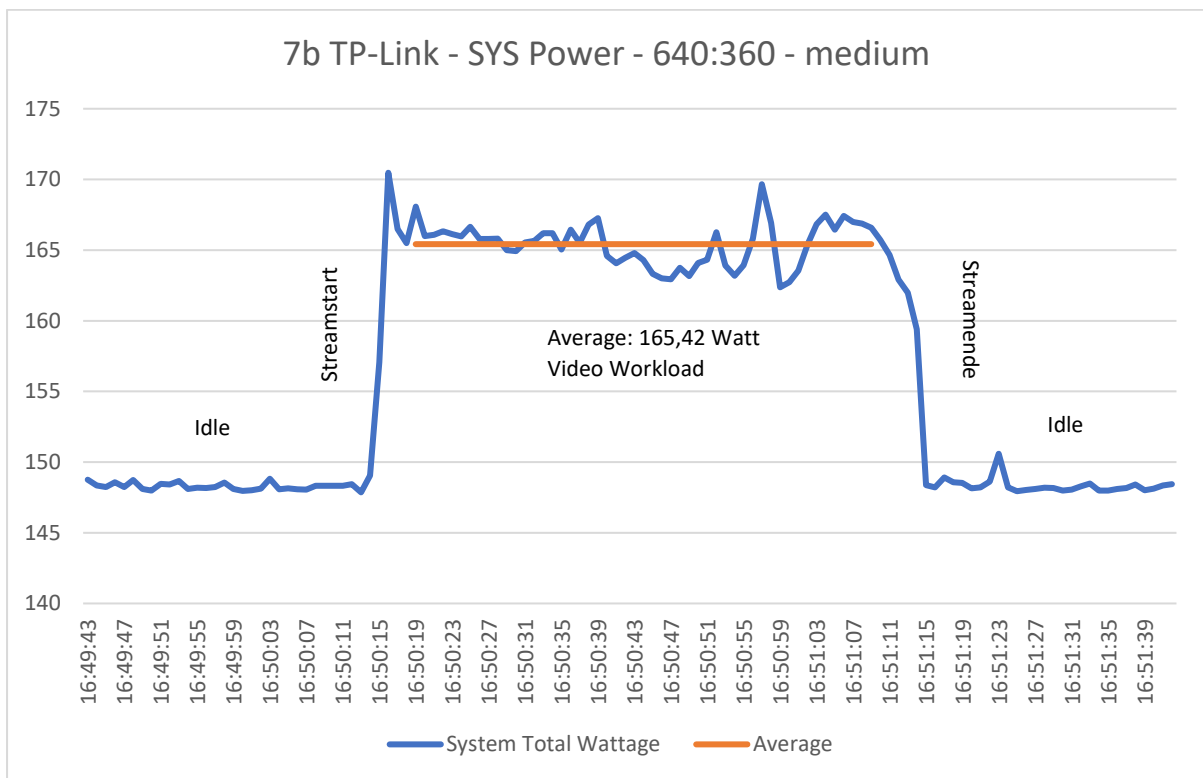
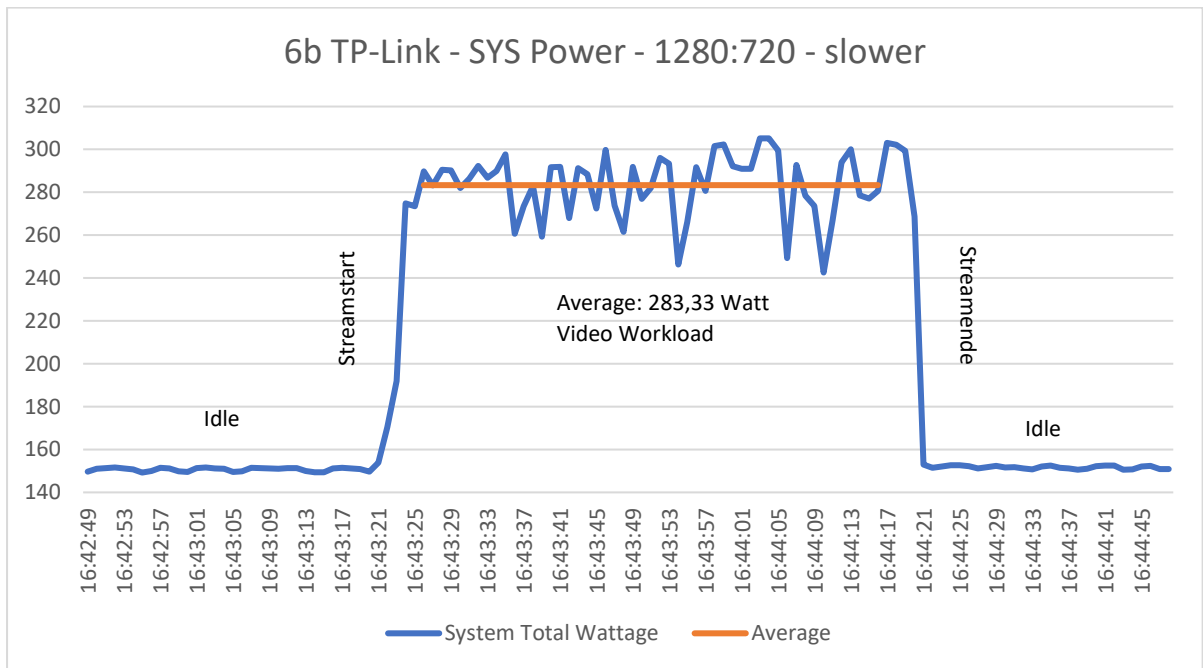


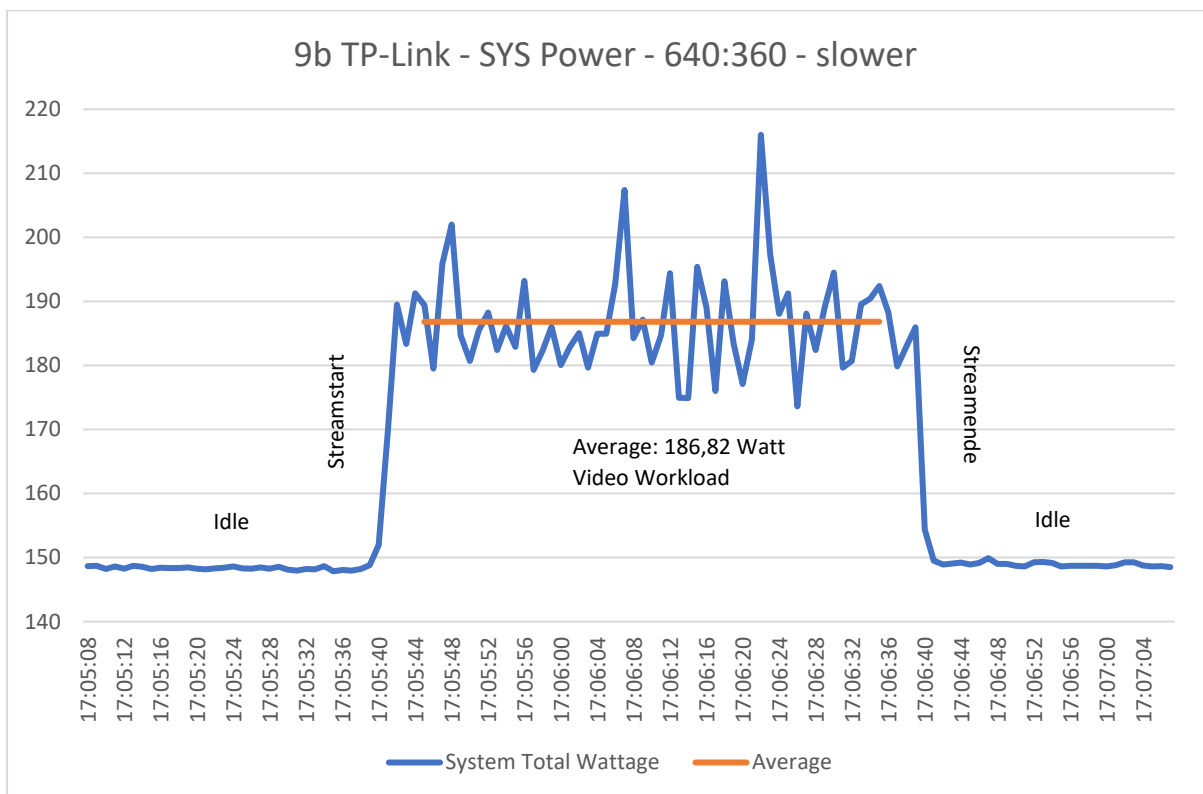
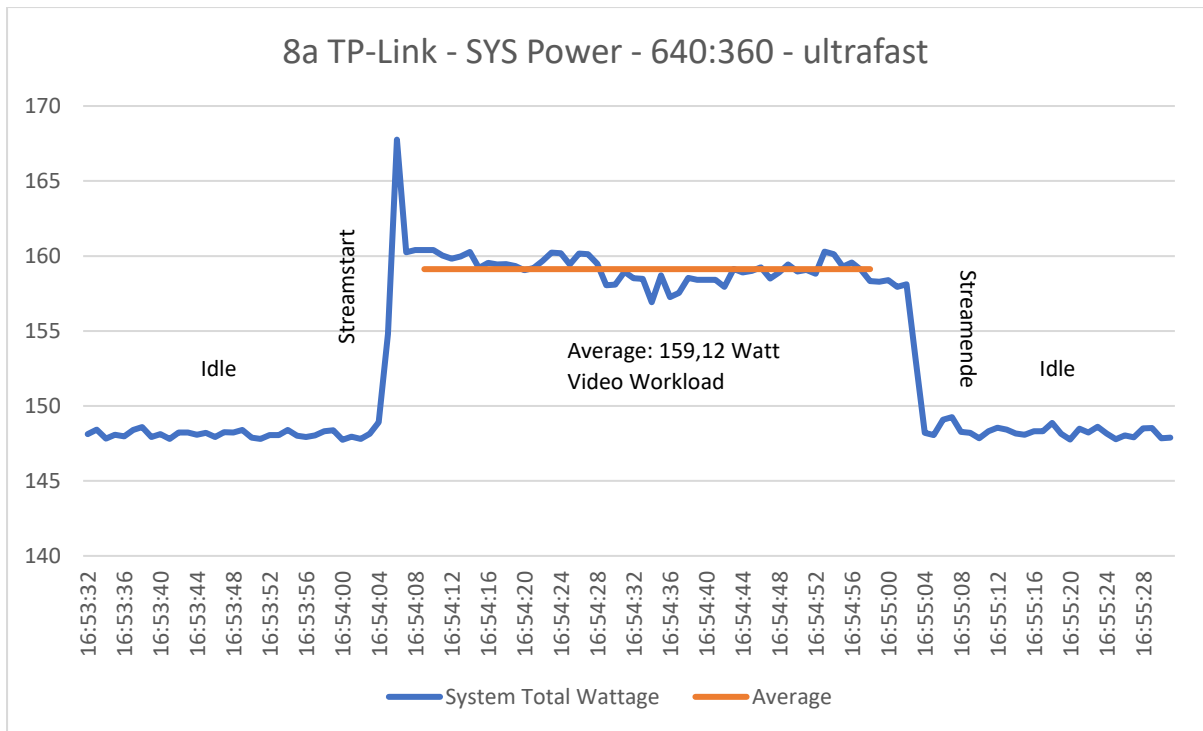
4b TP-Link - SYS Power - 1280:720 - medium



5b TP-Link - SYS Power - 1280:720 - ultrafast

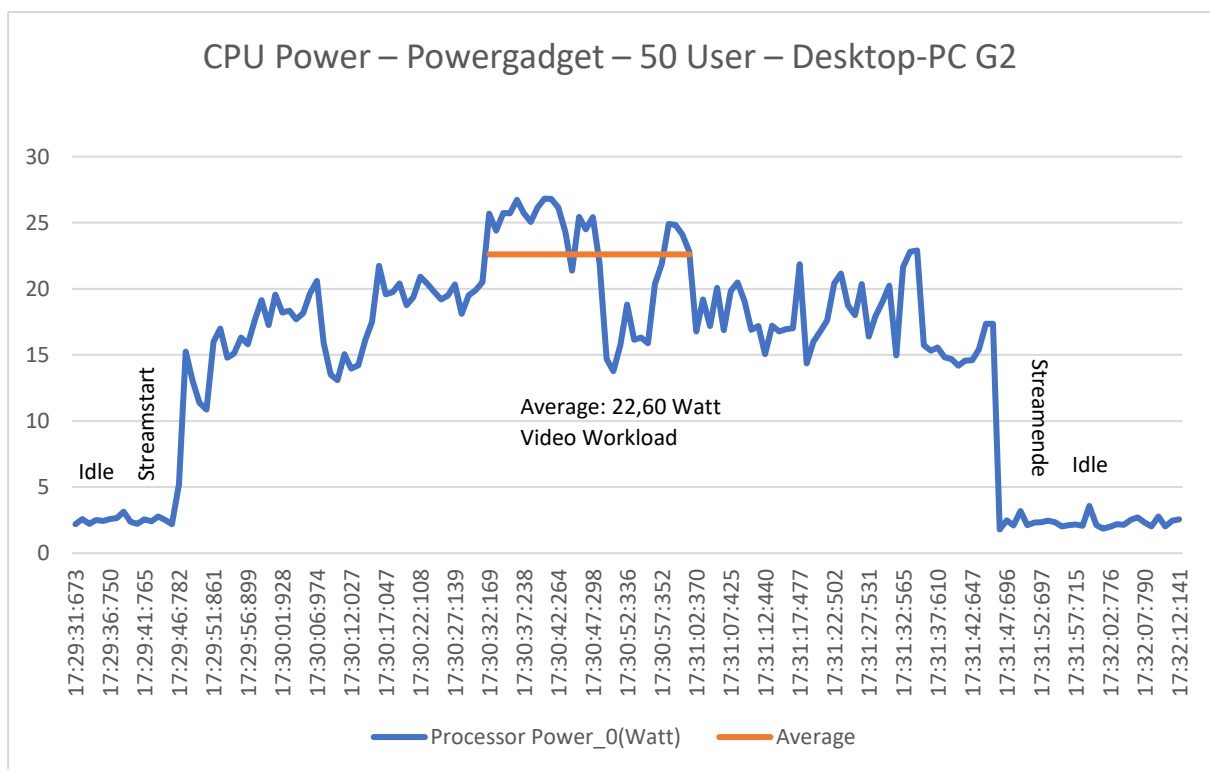


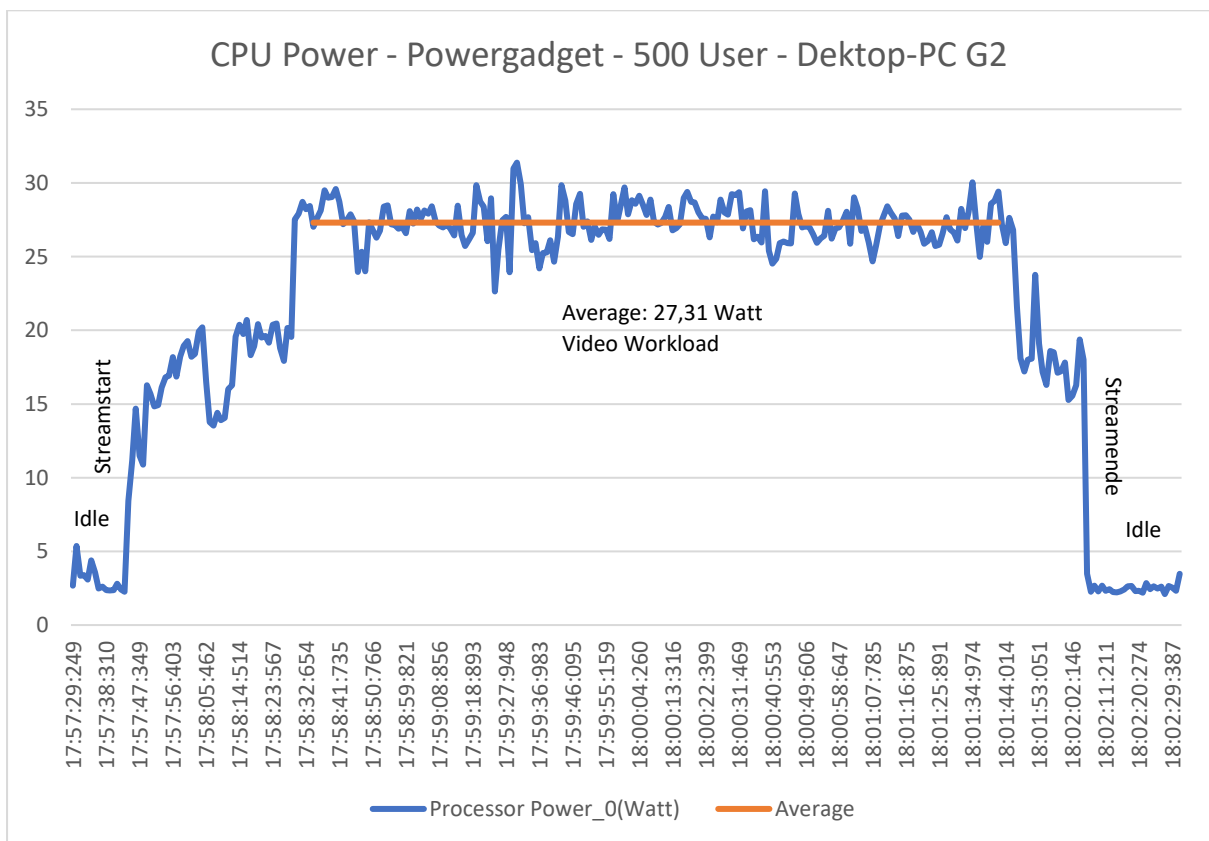
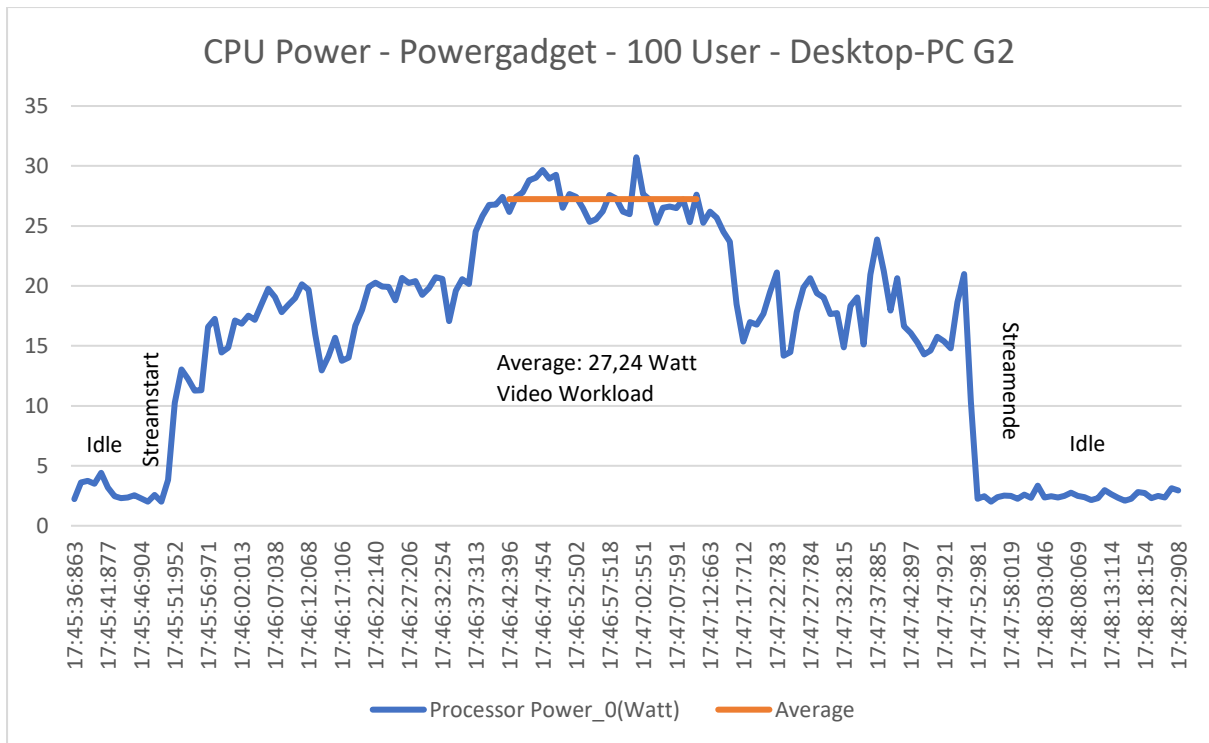


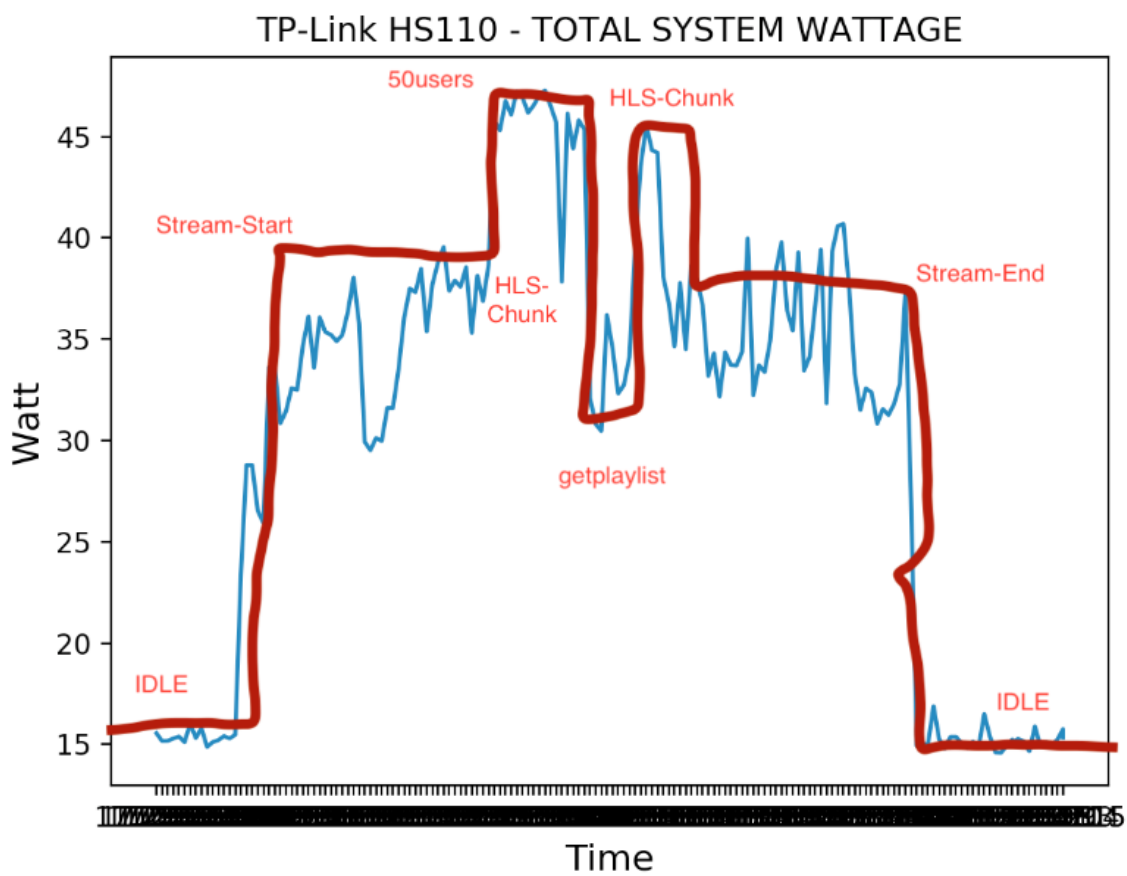
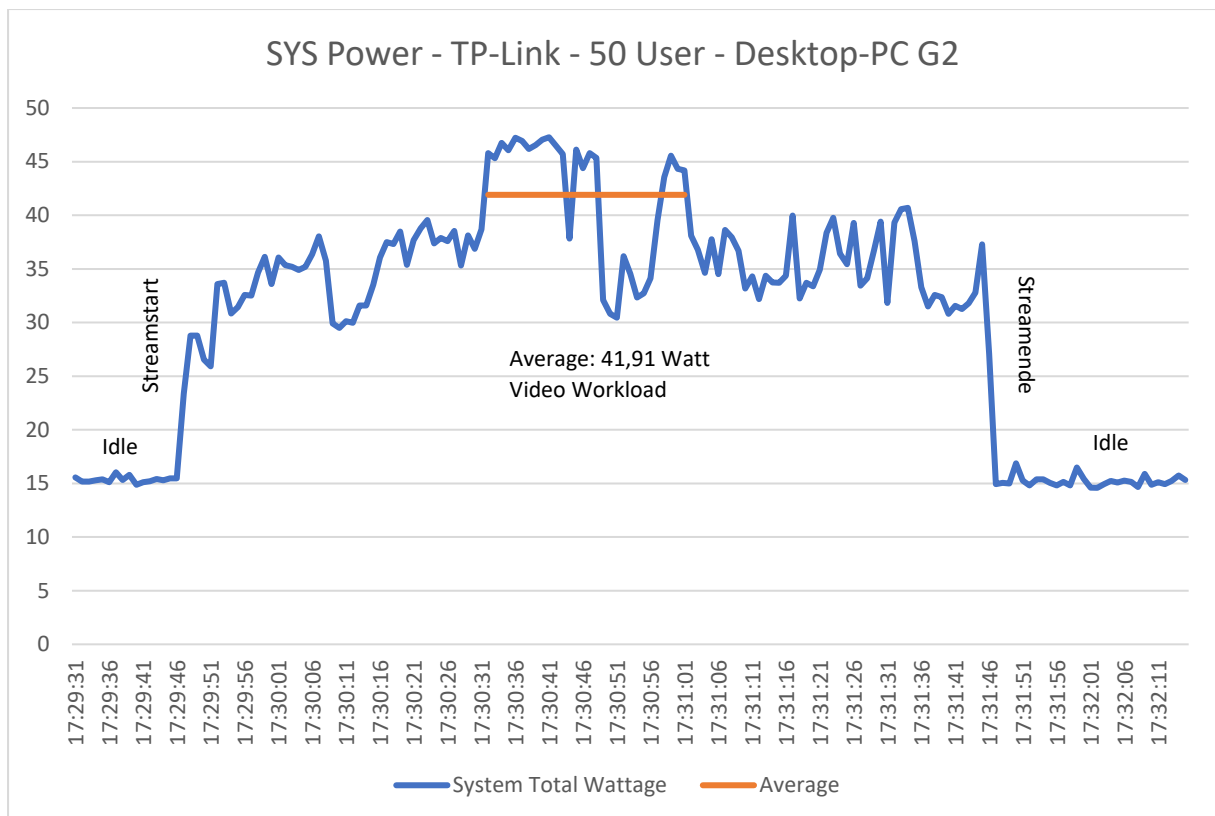


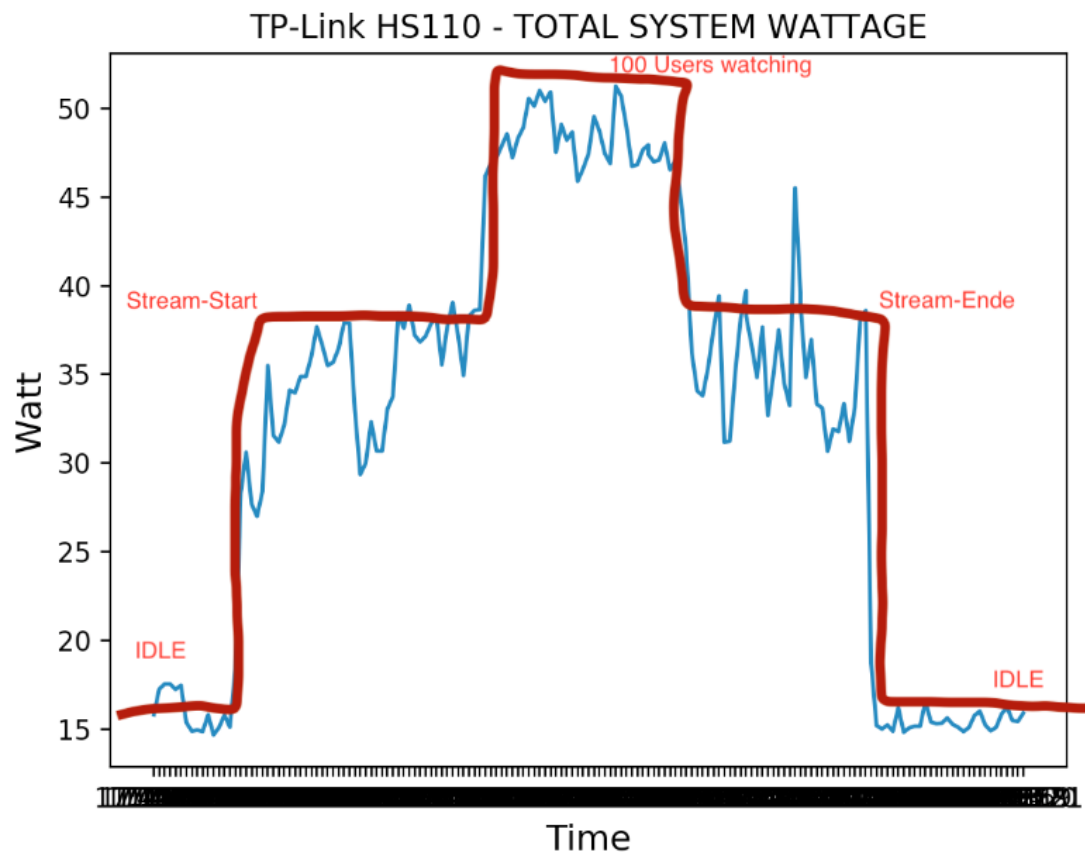
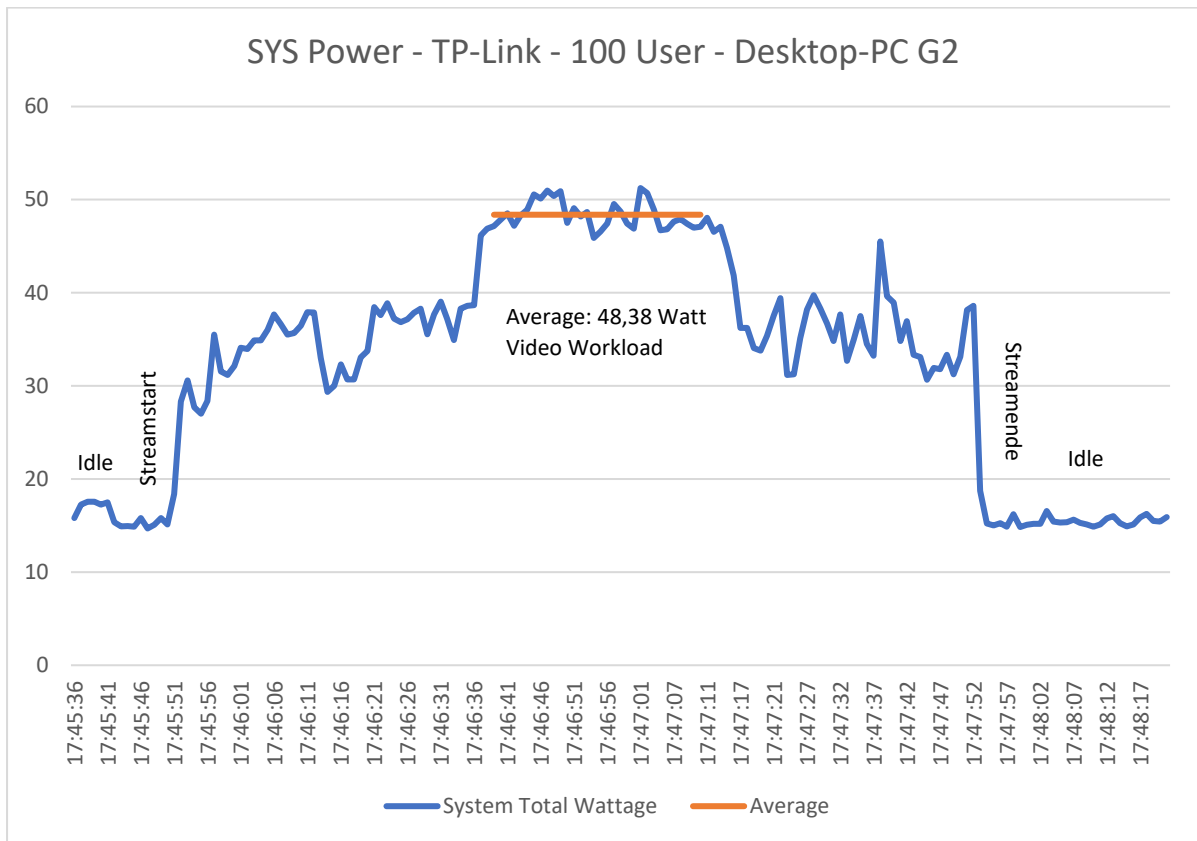
Szenario 2 - JMeter

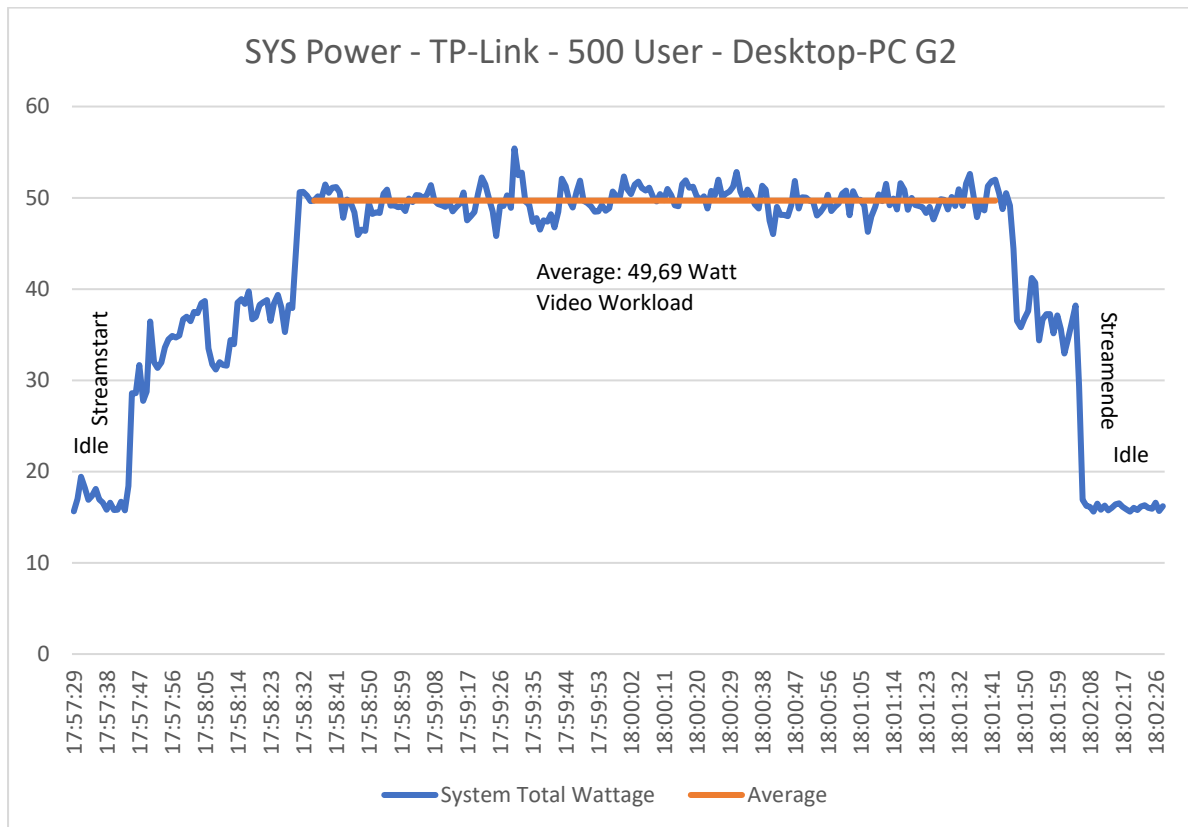
JMeter – Desktop-PC G2								
-vf scale	-b:v	-b:v	FPS	-preset	libx264 –crf	SYS POWER [W] TP-Link	CPU POWER [W] Intel	User
1920:1080	6000	CBR	60	ultrafast	23	41.91	22.60	50
1920:1080	6000	CBR	60	ultrafast	23	48.38	27.24	100
1920:1080	6000	CBR	60	ultrafast	23	49.69	27.31	500



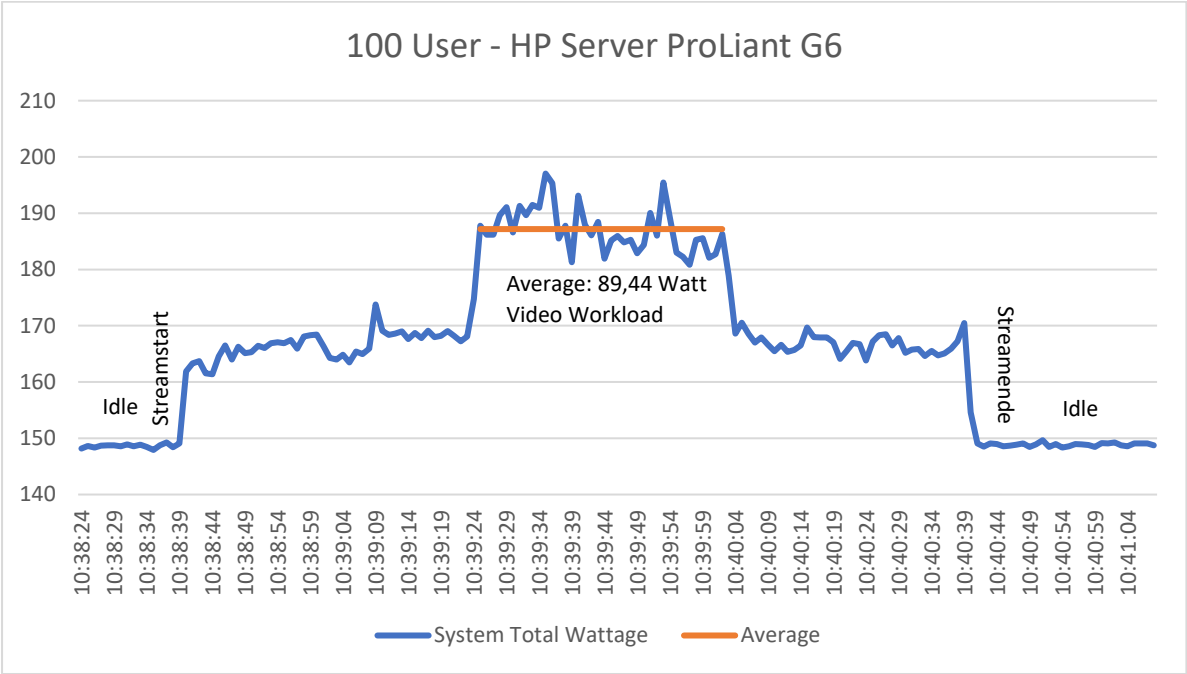
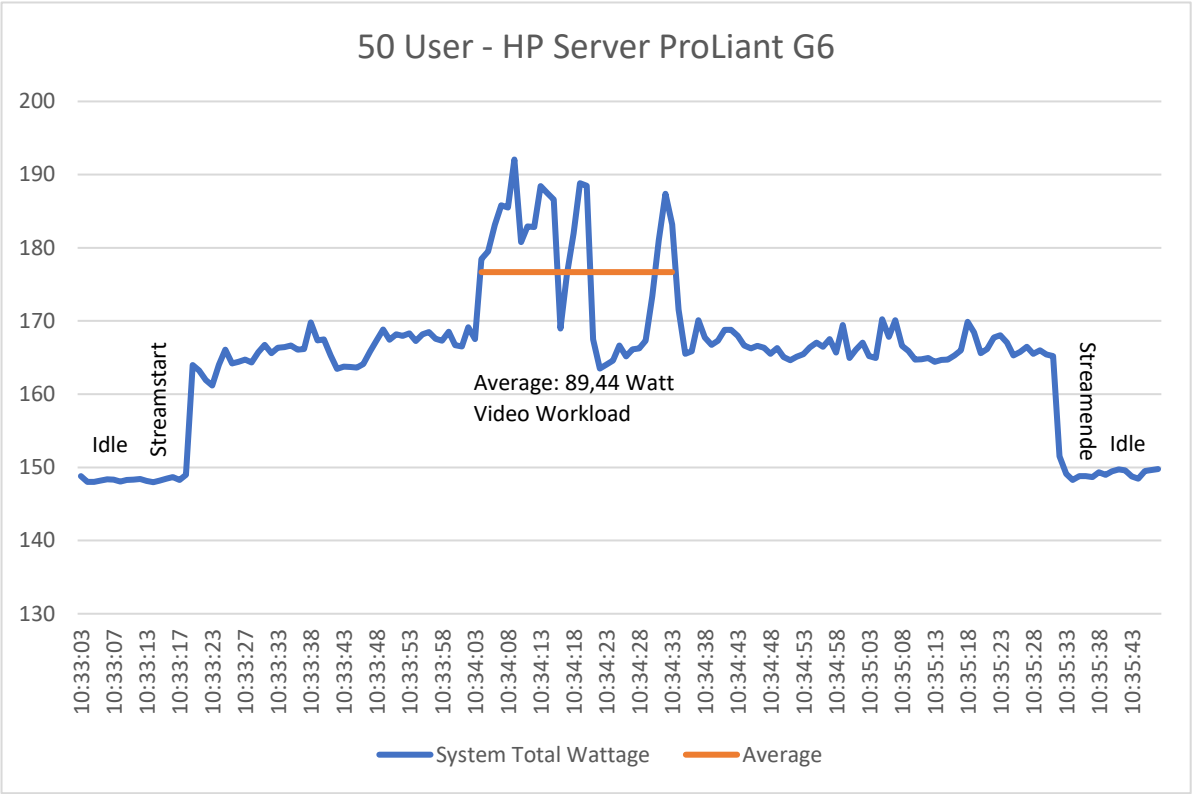


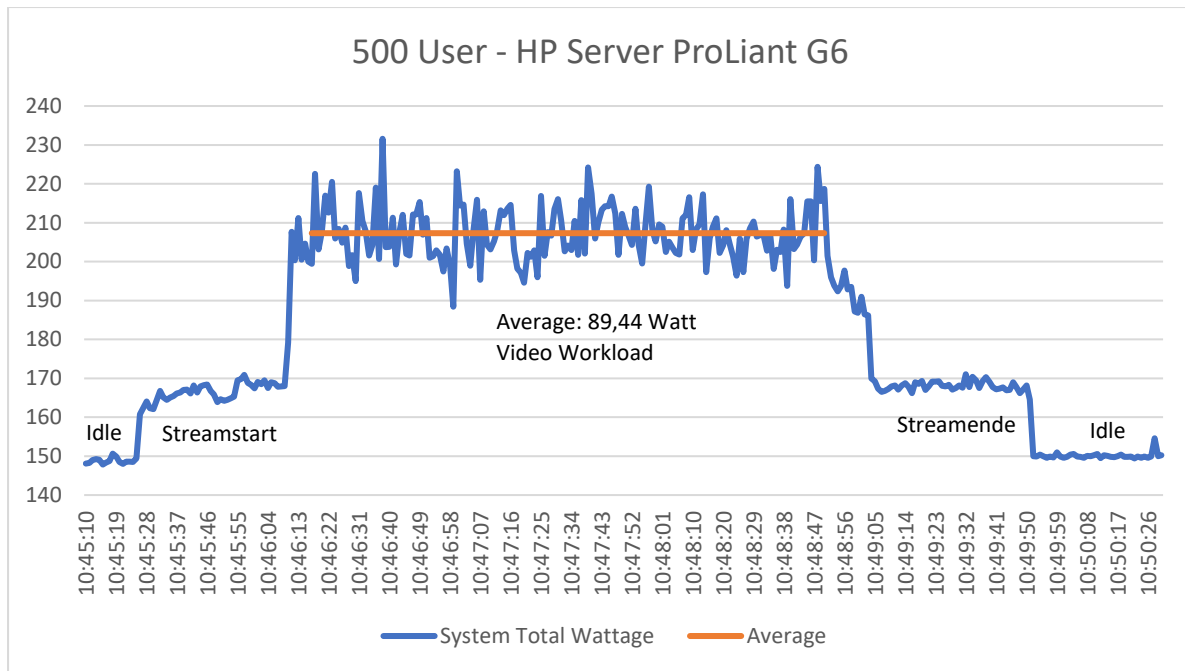






JMeter – HP Server ProLiant G6							
-vf scale	-b:v	-b:v	FPS	-preset	libx264 –crf	SYS POWER [W] TP-Link	User
1920:1080	6000	CBR	60	ultrafast	23	176.68	50
1920:1080	6000	CBR	60	ultrafast	23	187.18	100
1920:1080	6000	CBR	60	ultrafast	23	207.31	500





11 Energieverbrauch bei der Übertragung über das Internet

12 Probleme & Lessons Learned

13 Empfehlung und Ausblick

14 Zusammenfassung / Conclusio

15 Projektantrag





























Projektname:	GreenWeb Prototype Prototype-Entwicklung für Energie- und Performance-Messungen	
Projektnummer:	6	
Projektantrag		
Vorhaben / Projektziele:	Mit diesem Projekt sollen folgende Ziele erreicht werden: <div><div>1.</div><div>Recherche und Verweis auf etwaige bekannte Lösungsansätze für Green Web / IT</div></div> <div><div>2.</div><div>Aufbau einer Infrastruktur zur Messung und Datenergebung bezüglich des Vergleichs des Energieverbrauchs in folgenden Anwendungsbereichen:<div><div>a.</div><div>Video-Streaming Dienst (1080p, 4k, h265, h264)</div></div><div><div>b.</div><div>Energieverbrauch von Verschlüsselungstechniken</div></div><div><div>c.</div><div>Dynamische & Statische Webseiten</div></div><div><div>d.</div><div>Server / Client Energiemessungen</div></div></div></div> <div><div>3.</div><div>Kontaktaufnahme & Recherche bezüglich GreenWeb / GreenIT incl. externe Quellen z.B. Rechenzentrum</div></div> <div><div>4.</div><div>Datenerhebung damit diese vom anderen Team visuell aufbereitet werden können.</div></div>	
Meilensteine	<div><div>1.</div><div>Projektstart</div></div> <div><div>2.</div><div>Inoffizielles Kickoff-Meeting</div></div> <div><div>3.</div><div>Offizielles Kickoff-Meeting</div></div> <div><div>4.</div><div>Infrastruktur aufgebaut</div></div> <div><div>5.</div><div>Erste Messergebnisse</div></div> <div><div>6.</div><div>Finale Messergebnisse</div></div> <div><div>7.</div><div>Abschlussmeeting</div></div> <div><div>8.</div><div>Finale Dokumentation</div></div> <div><div>9.</div><div>Projektpräsentation</div></div> <div><div>10.</div><div>Projektabgabe</div></div> <div><div>11.</div><div>Projektende</div></div>	
Fertigstellungstermine	<div><div>10.10.19</div><div>Projektantrag</div></div> <div><div>10.10.19</div><div>Meilensteine definiert</div></div> <div><div>25.10.19</div><div>Infrastruktur aufbauen & einrichten</div></div> <div><div>22.11.19</div><div>Messungen durchführen</div></div> <div><div>15.12.19</div><div>Ergebnisse analysieren</div></div> <div><div>22.01.20</div><div>Projektdokumentation</div></div> <div><div>27.01.20</div><div>Projektpräsentation / Abgabe</div></div>	
Kosten	Grundsätzlich entsteht kein Kostenaufwand Mögliche Hardwarekosten: Energiemessgerät (z.B. Smart Meter)	

	<p>Lizenzkosten für Windows Server etc. werden von der FH-JOANNEUM gedeckt.</p> <p>Andere Kosten: Zeit, Tickets für öffentliche Verkehrsmittel</p>
Personenaufwand	Kostenlos
Sachmittel	<p>Server: HP ProLiant DL380 G6</p> <p>Geräte: HP EliteDesk 800 G1, HP EliteDesk 800 G2, Smartphone, Tablet, Smart Meter, Messgeräte zur Energiemessung, Peripheriegeräte, Netzwerkanbindung</p> <p>Software: Windows Server 2019, Hyper-V, Linux Distro, Open Source Software</p>
Risiko	Zeitmanagement, Überarbeitung, Inhaltliche Differenzen, keine aussagekräftigen Ergebnisse, Projektzielverfehlung
Projektorganisation	
Auftraggeber	FH Joanneum – DI Georg Mittenecker
Projektleiter	Colin Jochum
Projektmitglieder	Colin Jochum, Tom Kleinhapl, Marcel Kahr
Projektausschuss	DI Werner Fritz, DI Georg Mittenecker, Colin Jochum, Tom Kleinhapl, Marcel Kahr

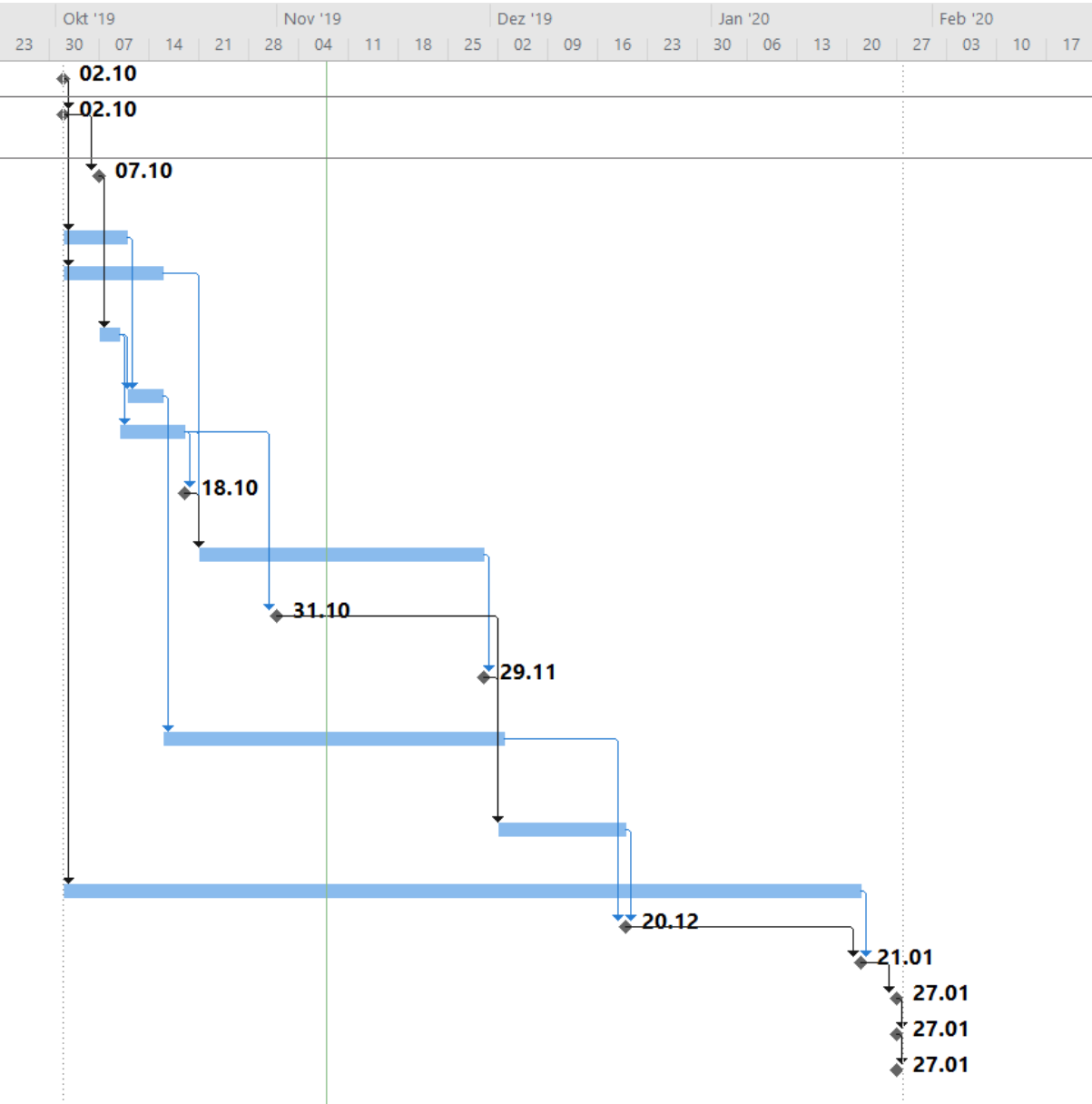
16 Meilensteine

02.10.19	Projektstart
02.10.19	Inoffizielles Kickoff-Meeting
07.10.19	Offizielles Kickoff-Meeting
09.10.19	Meilensteindefinition
10.10.19	Projektantrag
18.10.19	Infrastruktur aufgebaut
08.11.19	Erste Messergebnisse
22.11.19	Finale Messergebnisse
20.12.19	Abschlussmeeting
21.01.20	Finale Dokumentation
27.01.20	Projektpräsentation
27.01.20	Projektabgabe
27.01.20	Projektende

17 Projektplan

		Vorgan ▾	Vorgangsname ▾	Dauer ▾	Anfang ▾	Ende ▾	Vorgänger
1			Projektstart	0 Tage	Mit 02.10.19	Mit 02.10.19	
2			Inoffizielles Kickoff-Meeting	0 Tage	Mit 02.10.19	Mit 02.10.19	1
3			Offizielles Kickoff-Meeting	0 Tage	Mon 07.10.19	Mon 07.10.19	2
4			Projektantrag	7 Tage	Mit 02.10.19	Don 10.10.19	2
5			Recherche / Ideenfindung	10 Tage	Mit 02.10.19	Die 15.10.19	1
6			Zielvereinbarungen / Meilensteine	3 Tage	Mon 07.10.19	Mit 09.10.19	3
7			Arbeitspakete planen	3 Tage	Fre 11.10.19	Die 15.10.19	6;4
8			Infrastruktur aufbauen	7 Tage	Don 10.10.19	Fre 18.10.19	6
9			Infrastruktur aufgebaut	0 Tage	Fre 18.10.19	Fre 18.10.19	8
10			Messungen durchführen	30 Tage	Mon 21.10.19	Fre 29.11.19	8;9;5
11			Erste Messergebnisse	0 Tage	Don 31.10.19	Don 31.10.19	8
12			Finale Messergebnisse	0 Tage	Fre 29.11.19	Fre 29.11.19	10
13			Sonstige Informationen sammeln	34 Tage	Mit 16.10.19	Mon 02.12.19	7
14			Ergebnisse analysieren	14 Tage	Mon 02.12.19	Don 19.12.19	11;12
15			Projektdokumentation	80 Tage	Mit 02.10.19	Die 21.01.20	2
16			Ablussmeeting	0 Tage	Fre 20.12.19	Fre 20.12.19	14;13
17			Finale Dokumentation	0 Tage	Die 21.01.20	Die 21.01.20	15;16
18			Projektpräsentation	0 Tage	Mon 27.01.20	Mon 27.01.20	17
19			Projektabgabe	0 Tage	Mon 27.01.20	Mon 27.01.20	18
20			Projektende	0 Tage	Mon 27.01.20	Mon 27.01.20	19

18 Gantt Chart



19 Stundenerfassung

Code	Aufgaben	Verantwortlich	Start	Soll-Ende	Ist-Ende	Tage	Stunden geplant	Stunden benötigt	Priorität	Bemerkung	Status
1	Inoffizielles Kick-off-Meeting	Colin Jochum	02.10.19	02.10.19	02.10.19	1	1,25	1,25	Mittel		Abgeschlossen
2	Offizielles Kick-off-Meeting	Colin Jochum	07.10.19	07.10.19	07.10.19	1	1	1	Hoch		Nicht begonnen
3	Projektantrag	Colin Jochum	02.10.19	10.10.19	09.10.19	8	2,75	2,5	Hoch		In Bearbeitung
4	Recherche / Ideenfindung	Colin Jochum	02.10.19	15.10.19		13	10	8	Niedrig		Nicht begonnen
5	Zielvereinbarungen / Meilensteine	Colin Jochum	07.10.19	09.10.19	09.10.19	2	1	1	Niedrig		Nicht begonnen
6	Arbeitspakete planen	Colin Jochum	10.10.19	14.10.19	14.10.19	4	1	1	Niedrig		Nicht begonnen
7	Infrastruktur aufbauen	Colin Jochum	10.10.19	18.10.19	17.10.19	8	30	35,5	Niedrig		Nicht begonnen
8	Messungen durchführen		21.10.19	29.11.19		39	20		Niedrig		Nicht begonnen
9	Sonstige Informationen sammeln		15.10.19	29.11.19		45	8		Niedrig		Nicht begonnen
10	Ergebnisse analysieren		21.10.19	29.11.19		39	5		Niedrig		Nicht begonnen
11	Projektdokumentation		02.10.19	21.01.20		111	18	1	Niedrig		Nicht begonnen
12	Abschlussmeeting		20.01.20	20.01.20		1	1		Niedrig		Nicht begonnen
13	Projektpräsentation		27.01.20	27.01.20		1	0,5		Niedrig		Überfällig
14	Projektabgabe		27.01.20	27.01.20		1	0,5		Niedrig		Nicht begonnen
Gesamt			117				100	51,25			

20 Tätigkeitsbericht

	1. Treffen Aufbau Infrastruktur am 09.10.2019 (11 Uhr - 18:30 Uhr)
Colin	<ul style="list-style-type: none"> • Meilensteine überarbeitet und festgelegt • Projektantrag erstellt • PowerPoint Präsentation für PMGL → Projektcheck 1 erstellt <ul style="list-style-type: none"> ○ Projektantrag, Grundparameter, Problemfeldanalyse • 1. Aufsetzversuch von Windows Server 2019 auf Serverhardware Version 2009 → dies verursachte einige Probleme • Windows 10 Desktop-Client aufgesetzt • Windows Server 2019 auf Desktop-PC aufgesetzt <ul style="list-style-type: none"> ○ Hyper-V installiert ○ Einen virtuellen Debian Server aufgesetzt ○ Zwei virtuelle Windows Pro Versionen aufgesetzt • Testen auf Kompatibilität diverser Energie Messungstools <ul style="list-style-type: none"> ○ Intel Power Gadget und Joulemeter nur am Parent möglich, nicht in der VM • Recherche diverser Toolkomponenten → Smartmeter (TP-Link)
Tom	
Marcel	
	2. Treffen Aufbau Infrastruktur am 10.10.2019 (12:45 Uhr - 17 Uhr)
Colin	<ul style="list-style-type: none"> • Versuchter Aufbau einer Internetverbindung mit Hilfe von WLAN-Adaptern zum Server von Herrn Geister <ul style="list-style-type: none"> ○ Verbindung konnte nicht hergestellt werden, da kein passender Treiber im Internet gefunden wurde • Persönliche Kontaktaufnahme mit Herrn Geister bezüglich Internetzugriff <ul style="list-style-type: none"> ○ Neuinstallation der auf Linux basierenden VM am Desktop-Client, aufgrund von Konfigurationsproblemen beim GUI am Vortag -> GUI LXQt (Leightweighted) ○ Netzwerkeinstellungen im GUI -> Statische IPs ○ Installation und Testen der Tools "PowerTop" und "PowerState" ○ Da wir keine Verbindung zum "RAPL" herstellen konnten, (kann in VMs nicht genutzt werden, nur am Parent), können mit dieser Schnittstelle konnten keine Daten zur Energiemessung ausgelesen werden ○ Nach Recherchen und Tests konnte herausgefunden werden, dass es mit Hilfe des Tools „Speed“ etwaige Änderungen des Energieverbrauchs beim Intel Power Gadget am Parent gab. • "Hacken" des von der FH zur Verfügung gestellten Servers mit Hilfe von Recherchen aus dem Web, da das Root-Passwort gesetzt war und wir keinen Zugriff auf den Sever hatten
Tom	
Marcel	

	<ul style="list-style-type: none"> • Erstellung einer VM am HP Server • Mit Hilfe der "XCP-ng Center" App, konnte zum Server eine SSH Verbindung aufgebaut werden, sowie diverse VMs erstellt werden • Nach einigen Schwierigkeiten beim Einlesen des ISO-Files (DEBIAN), welches wir aus dem Internet herunterladen konnten, wurde erfolgreich die Installation einer virtuellen Linux-Maschine am Server sowie etwaige Netzwerkkonfigurationen durchführen werden
	3. Treffen Aufbau Infrastruktur am 14.10.2019 (10:10 Uhr - 12:20 Uhr)
Colin	<ul style="list-style-type: none"> • Es wurde eine native Linux-Partition als zweite Partition am Desktop-Client G2 aufgesetzt. • Weiters wurde ein NGINX-Webserver auf VM Debian am Windows Server aufgesetzt • Die Installation und Konfiguration der BSD Streaming Software wurde auf der VM durchgeführt • 1. Streamingversuch über Server via RTMP-Protokoll, welcher jedoch nicht funktionierte <ul style="list-style-type: none"> ○
Marcel	
Tom	<ul style="list-style-type: none"> • Inbetriebnahme und Konfiguration eines Access Points mit OpenWrt <ul style="list-style-type: none"> ○ Die Konfiguration konnte aufgrund von Komplikationen nicht vollständig umgesetzt werden
	4. Treffen Aufbau Infrastruktur am 17.10.2019 (- 17 Uhr)
Colin	<ul style="list-style-type: none"> • Beseitigung von Installationsproblemen mit NGINX • Installation des Mediaplayers Kodi (Media Server) auf VM Debian <ul style="list-style-type: none"> ○ Konfiguration der Kodi-Software im Web-Interface, um auf den Media Player via http zugreifen zu können ○ Download diverser Video Clips in unterschiedlichen Qualitäten (4k, UHD, 1080p) und Zugriff auf diese Clips ○ Es stellte sich heraus, dass das Streamen von Kodi nur über den Internet Explorer möglich ist • Durchführung erster Leistungsmessungen mit privaten Notebooks durch Streaming der Videos über Kodi <ul style="list-style-type: none"> ○ Das Ergebnis zeigte, dass eine kurzfristige CPU-Auslastung von rund 100% erreicht wurde
Marcel	
Tom	<ul style="list-style-type: none"> • Erneute Konfiguration des Access Points, um via Wi-Fi auf den Media-Server zugreifen zu können <ul style="list-style-type: none"> ○ Dabei wurden wiederum erste Leistungs- und Energieverbrauchstests durchgeführt

	<ul style="list-style-type: none"> • ILO-Schnittstelle wurde konfiguriert und getestet, ob man darauf zugreifen kann <ul style="list-style-type: none"> ◦ Der Zugriff ist nur mit Hilfe älterer Explorer wie dem Internet Explorer möglich. Außerdem ist sehr interessant, dass Energiemessungen nur im Intervall von 24 Stunden aufgezeichnet werden können
	5. Treffen Aufbau Infrastruktur am 23.10.2019 (11 Uhr – 18:30 Uhr)
Colin	<ul style="list-style-type: none"> • Recherchen zu geeignete APIs, um Daten vom Intel Power Gadget auslesen zu können
Marcel	<ul style="list-style-type: none"> • Erste Ergebnisse konnten via CSV-File in Excel eingelesen und aufgelistet werden • Recherche bezüglich HSL via RMTP und NGINX • Nach der Recherche nach geeigneten Anleitungen, versuchten wir durch Änderungen des nginx.conf-Files "HLS" freizuschalten. Da sich herausstellte, dass nach der Manipulation des config-Files sich NGINX nicht mehr starten lässt, mussten wir den Fehler debuggen • Das Debugging zeigte, dass der Fehler beim HLS liegt • Letztendlich setzten wir eine neue VM basierend auf CentOS (vom ftp-Server der TU-Graz) auf und befolgten eine „tolle“ Schritt für Schritt Anleitung, von der kompletten Neuinstallation von NGINX bis zum Aufsetzen des Streaming-Servers. • Da wir nicht mit alten Versionen arbeiten wollten, suchten wir natürlich die neuesten Releases und installierten diese. Auch neue Directorys legten wir an, da wir das OS so lightweighted wie möglich aufgesetzt haben. • Dieses lightweighted OS sowie die neuen Versionen wurden uns zum Verhängnis, da diese nicht mit der Anleitung (Konfiguration) übereinstimmten. Hinzu kam, dass die Anleitung teilweise fehlerhaft war und Fehler wie (zu wenige geschwungene Klammern usw.) beinhaltete. • Nach langer Fehlersuche konnten wir die Installation von NGINX mit HLS abschließen und zum Streamen beginnen. • Beim Versuch einen Clip mit Hilfe von ffmpeg zu streamen, erhielten wir wiederum einen Fehler nämlich "Server error: NetStream.Record.Failed" und beließen es für den Tag.
Tom	<ul style="list-style-type: none"> • Tom Kleinhapl versuchte einen RSTP-Server aufzusetzen
	6. Treffen Aufbau Infrastruktur am 24.10.2019 (12:30 Uhr – 19:30 Uhr)
Colin	

Marcel	<ul style="list-style-type: none"> • Nach dreistündiger Recherche und Ausprobieren von etwaigen ffmpeg-Befehlen, konnten wir zufällig beim Betrachten der Fehlermeldung, beim Ausführen des ffmpeg-Befehls herausfinden, dass man im Config-File von Nginx den Parameter "Record" auf off stellen muss. • Danach konnte der ffmpeg-Befehl problemlos ausgeführt werden. Es wurden einige Probemessungen mit unterschiedlichen Parametern von ffmpeg durchgeführt, wie z.B. (unterschiedliche Kodierungen (libx264 usw.). • Da standardmäßig "copy" verwendet wird, war die CPU des PCs bei minimaler Leistung. • Erst nach einfügen von "libx264" erzielte die CPU max. Auslastung (100% bei allen Cores). • Danach konnten wir durch diverse Einstellungen mithilfe von Parametern die CPU-Auslastung drosseln. • Als nächsten Schritt versuchten wir den Stream über HLS (http) zu erreichen. Nach einigen Problemen und Recherchen konnten wir die richtige Anleitung finden. Dabei mussten wir den ffmpeg-Befehl direkt auf unserer CentOS-VM ausführen, da das MP4-File in HLS Segmente gespeichert wird. • Aus diesen Segmenten wird eine Playlist generiert, auf welche per http://172.17.32.53/live/playlist.m3u8 zugegriffen werden kann. • In der Config von NGINX wurde nämlich festgelegt, dass man auf den Ordner Live für HLS zugreifen kann. • Da beim Streamen der Playlist, die Playlist automatisch von NGINX gelöscht wird, mussten wir in der NGINX Config "hsl_cleanup = off" einstellen, damit man unendlich streamen kann
Tom	<ul style="list-style-type: none"> • Aufsetzen des YouTube Clones auf dem HP Server – VM Debian unter Einfluss kleinerer Probleme
	7. Treffen Aufbau Infrastruktur am 29.10.2019 (13:45 Uhr – 17 Uhr)
Colin	<ul style="list-style-type: none"> • Projektcheck für PMTGL vorbereitet • Versuch ffmpeg Befehl für HLS in die NGINX Konfigurationsdatei (nginx.conf) zu implementieren
Tom	
Marcel	
	8. Treffen Aufbau Infrastruktur am 30.10.2019 (10:10 Uhr – 17 Uhr)

	<ul style="list-style-type: none">• 30 Minuten Meeting um 14 Uhr mit Sandra Schadenbauer und Georg Mittenecker → Vorführung des bereits umgesetzten und was geplant ist<ul style="list-style-type: none">○ Wichtige Punkte des Meetings: Echtzeitauslesung und Übertragung der Daten, sowie verschlüsselter Stream
Colin	<ul style="list-style-type: none">• Versuch mit der API Daten vom Intel Power Gadget in ein eigens File zu speichern, welches das Partnerteam zum Visualisieren benötigt
Tom	<ul style="list-style-type: none">• Versuch NGINX auch auf dem HP Server zu installieren, da die Installation erfolgreich am CentOS umgesetzt wurde<ul style="list-style-type: none">○ Verursachte Konfigurationsprobleme, welche erst nach stundenlanger Recherche gelöst werden konnte → Firewall Portfreischaltung○ Die Konfigurationsdatei von NGINX wurde angepasst und verbessert• Projektdokumentation erstellt sowie die wichtigsten Überschriften ergänzt• Erstellung einer PowerPoint für den 2. Projektcheck in PMTGL
Marcel	
	9. Treffen am 05.11.2019 (15 Uhr – 17:15 Uhr)
Colin	<ul style="list-style-type: none">• Erster Versuch unsere Wi-Fi Plugs zu konfigurieren<ul style="list-style-type: none">○ Konnte leider nicht umgesetzt werden, da für die Konfiguration ein DHCP benötigt wird, welchen wir jedoch nicht am Access Point eingestellt hatten → Zur Genehmigung musste mit Herrn Geister Kontakt aufgenommen werden.• Durchführung von Recherchen für den HP Server Xenon, bezugnehmend auf etwaige Energie Messungstools<ul style="list-style-type: none">○ Bis auf die ILO-Schnittstelle wurde kein vergleichbares Tool gefunden• Recherche etwaiger ffmpeg-Befehle zur Codierung• Recherche diverser Stresstesttools um http Request Traffic zu generieren
Tom	
Marcel	
	10. Treffen am 06.11.2019 (10 Uhr – 14 Uhr)
	<ul style="list-style-type: none">• Meeting mit anderem Projektteam<ul style="list-style-type: none">○ Absprache auf zukünftige Vorhaben z.B. Echtzeitmessungen○ Projektteam Visualisierung stellt uns ihre statische und dynamische Website zur Verfügung, um Tests damit durchzuführen

Colin	<ul style="list-style-type: none"> • Aufsetzen eines Apache-Servers für die Website des Partnerteams • Zugriff mit Hilfe von Python und über eine API auf den Wi-Fi Plug
Tom	<ul style="list-style-type: none"> • Testen und arbeiten mit dem Stresstesttool JMeter und herausfinden, wie wir am besten Loadtests durchführen können • Recherche für etwaige andere Stresstesttools • Inbetriebnahme und Konfiguration der Wi-Fi Plugs mit der App Kasa sowie eine Neuvergabe von IP Adressen, da zur Konfiguration der Plugs ein eigener DHCP-Server benötigt wurde. Dabei musste uns ein internes Netz von Herrn Geister zur Verfügung gestellt werden, damit unser eigener DHCP nicht ins FH Netz routet.
Marcel	
	11. Treffen am 08.11.2019 (11:15 Uhr – 15:45 Uhr)
Colin	<ul style="list-style-type: none"> • TP-Link → Python3 Script inkl. Plotfunktion • Livescatterplot • Datenspeicherung in Listen und Datenspeicherung in CSV
Tom	<ul style="list-style-type: none"> • YouTube Clone gefixt, da durch die Neuvergabe der IP's die Konfiguration beschädigt wurde • Einige Probetests mit dem Jmeter Tool durchgeführt
Marcel	<ul style="list-style-type: none"> • Beginn der Dokumentation wie Zielsetzung, Einleitung, Struktur und Aufbau • Beginn der Definition von Use Cases für Messungen
	12. Treffen am 12.11.2019 (10 Uhr – 17:15 Uhr)
Colin Marcel	<ul style="list-style-type: none"> • Versuch auf den CentOS-VMs, zuerst auf der G2 und dann auf dem HP Server die neueste Version von ffmpeg zu installieren, da mit der neueren Version mehr Optionen möglich sind. • Da wir zuvor nur herausfanden, dass ffmpeg nur bis zur Version 2 für CentOS supported wird, konnten wir nur Version 2 installieren. Doch nach einigen Recherchen kamen wir drauf, dass doch die Version 4 mit Hilfe eines Plug-Ins namens "snapd" installiert werden kann. • Dabei mussten wir zuerst die alte ffmpeg Version von CentOS löschen, um die neue mit Hilfe von Snapd installieren zu können. Im nächsten Schritt versuchten wir, ob mit der neuen Version NGINX noch ausgeführt werden konnte. Das Ergebnis zeigte, dass es keine Probleme beim Ausführen und Starten von NGINX gab.

	<ul style="list-style-type: none"> • Einziger Fehler war, dass der Stream, welcher mit ffmpeg im Browser als Mobile Version zurückgegeben werden hätte sollen, dass index.m3u8 File im Mobile Ordner nicht fand. • Nach über zweistündiger Fehlersuche, Recherche und Tests wie z.B. das Einfügen eines NGINX Users (user root) und das verwenden eines push-Befehls für die Subapplications (Moblie) in der NGINX Config, funktionierte aus irgendeinem unerklärlichen Grund das Aufrufen des Streams plötzlich. • Somit konnten wir die vorgenommenen Einstellungen und Konfiguration auch für die HP Greta CentOS verwenden, was auch in kürzester Zeit umgesetzt wurde. • Weiters versuchten wir mit dem JMeter-Tool Traffic zu generieren, was sich schwieriger als gedacht herausstellte. Mit Hilfe einer Anleitung von Blazemeter konnten einige Testversuche durchgeführt werden, jedoch passten unsere Ergebnisse nicht mit den Ergebnissen nicht überein. Somit ließen wir diesen Punkt offen. • Zu guter Schluss fertigten wir einige Usecases an, was wir eigentlich Messen möchten wie z.B. unterschiedliche ffmpeg Einstellungen (mit Codierungen) und z.B. ohne Codierungen usw.
Tom	<ul style="list-style-type: none"> • Versuch die ILO-Schnittstelle neu zu konfigurieren, um mit Hilfe von Python, Daten über JSON auszulesen.
	13. Treffen am 13.11.2019 (12:15 Uhr – 17:15 Uhr)
Colin	<ul style="list-style-type: none"> • Aufsetzen des Apache Servers für die WordPress Website vom Partnerteam • Dabei wurde die VM Ubuntu gelöscht und eine VM CentOS aufgesetzt, da es zu Problemen kam
Tom	<ul style="list-style-type: none"> • Installation des HTML5 Video Player und entfernen des Flash Players • Dabei mussten einige neue Konfigurationen durchgeführt werden
Marcel	<ul style="list-style-type: none"> • Recherche und Tests mit JMeter durchgeführt • Ein Record Script wurde in JMeter erstellt, um den Zugriff auf dynamische Webseiten zu simulieren • Weiters wurde ein weiteres Script mit dem HLS-Plugin erstellt, damit man auf Streams Traffic generieren kann

21 Zugangsdaten BÜPA

Server - Host	
Host	
hpgreta	Pa55w.rd
Server – VM	
hpgretadebian	Password1
ILO2 172.17.32.10	
Desktop Client G2	
Partition – Linux	
Greta	Password1
Partition – Windows Server 2019	
Administrator	Password1
VM1 – GretaDebian - Server	Password1
VM2 – GretaWin10 - Server	Password1
VM3 – GretaWin10 - Client	Password1
VM4 – GretaCentOS – Server	Password1
Desktop Client G1	
None	None
Access Point	
OpenWrt	apple123
Youtube Clon	
Maia DB	Password1

22 IP Konzept

Host	IPs
HPGreta	192.168.47.14
HPGreta-VM	192.168.47.24
HPGreta-VM2	192.168.47.34
G2-greta	192.168.47.13
G2-grteaNative	192.168.47.15
G2-VMDebian	192.168.47.23
G2-VMWin10-Client1	192.168.47.33

G2-VMCentOS	192.168.47.53
G2-VMUbuntu	192.168.47.63
G2-VMUbuntu2	192.168.47.43
G2-VMUbuntu3	192.168.47.73
G1-Client	192.168.47.12
Access Point	
DNS-Server	10.25.1.6
Gateway	192.168.47.254

23 Literaturverzeichnis

24 Abbildungsverzeichnis