DOCUMENT REFERENCE:
SQ301-005-EN

SAMKNOWS TEST METHODOLOGY
Methodology and technical information relating to the SamKnows testing platform.

February 2015

SAMKNOWS QUALITY CONTROLLED DOCUMENT.					
SQ	REV	LANG	STATUS	OWNER	DATED
301	003	EN	FINAL	SC	20150227
REVISION HISTORY					
DATED	REV	AUTHOR	COMMENTS		
20150224	005	SC	Updated Video section		
20141027	004	RR	Updated YouTube section		
20140901	003	SC	General updates		
20120407	002	SC	General updates		
20111101	001	SC	Original.		

Contents

1	**IMPORTANT NOTICE**	**3**
2	**INTRODUCTION**	**4**
3	**PERFORMANCE TESTS**	**6**
3.1	Speed Tests	6
3.2	Web Browsing	7
3.3	Video Streaming	7
3.4	Voice Over IP (Jitter)	8
3.5	UDP Latency and Packet Loss	8
3.6	UDP Latency and Loss Under Load	8
3.7	UDP Contiguous Loss (Disconnections)	9
3.8	DNS Resolution	9
3.9	FTP Transfer	9
3.10	Peer-To-Peer (BitTorrent)	9
3.11	Email Relaying	10
3.12	You Tube Measure	10
3.13	Netflix measure	12
3.14	BBC iPlayer measure	13
3.15	Multicast IPTV	13
3.16	Cross-traffic detection (Inline / pass-through)	14
3.17	Cross-traffic detection (Out-of-band)	15

1 Important Notice

IMPORTANT NOTICE

This document is confidential and is the property of SamKnows Ltd ("SamKnows"), and this document contains confidential and/or proprietary information belonging to SamKnows.

The distribution of this document is strictly controlled by SamKnows. This document may only be issued by an authorised officer of SamKnows to named persons with whom SamKnows has a current valid Non Disclosure Agreement ("Authorised Recipients"). Authorised Recipients are forbidden from distributing any part of this document in any form to any other persons without the express written permission of SamKnows.

Any person who is not an Authorised Recipient and who receives this document therefore in error is requested to (1) notify SamKnows of the fact by email to info@samknows.com, and (2) take reasonable steps to confidentially dispose of or delete (as may be applicable) this document.

Nothing in this document is intended to nor will create any binding obligation on SamKnows.

2 Introduction

SamKnows Whiteboxes – modified routers acting as an Ethernet bridge with additional testing software integrated are to be deployed onto the home network of each panellist to test and report on a range of key metrics as outlined in the table below:

Metric	Primary Measure(s)
Download speed	Throughput in Megabits per second utilising one or more concurrent TCP connections
Upload speed	Throughput in Megabits per second utilising one or more concurrent TCP connections
Web browsing	The total time taken to fetch a page and all of its resources from a popular website
Voice over IP	Upstream packet loss, downstream packet loss, upstream jitter, downstream jitter, round trip latency
UDP latency	Average round trip time of a series of randomly transmitted UDP packets
UDP packet loss	Percentage of UDP packets lost from latency test
UDP latency/loss under load	Average round trip time and packet loss of UDP packets whilst the line is heavily loaded with downstream or upstream traffic
UDP contiguous loss (Disconnections)	Events of two or more consecutively lost UDP packets to the same destination – essentially internet connection is lost
DNS resolution	The time taken for the ISP's recursive DNS resolver to return an A record for a popular website domain name
FTP throughput	Throughput in Megabits per second at which a file can be downloaded from or uploaded to an FTP server
Peer-to-peer	Throughput in Megabits per second at which a file can be downloaded from BitTorrent
Email Relaying	The time taken to relay an email via the ISP's SMTP servers and reach a target mail server
Video streaming (Generic)	The initial time to buffer, the number of buffer underruns and the total time for buffer delays of an emulated fixed-rate TCP video stream

Metric	Primary Measure(s)
Video Quality of Experience – YouTube, Netflix, BBC iPlayer	The highest bitrate that can be streamed from the content servers / caches of YouTube, Netflix and iPlayer without rebuffering (Bitrate Reliably Streamed) and the time taken for the video to start playing (Startup Delay)
Multicast IPTV	Time to switch IPTV channels, and the jitter and packet loss observed in the multicast stream

3 Performance Tests

SamKnows has designed and developed its performance tests in house, adhering to IETF RFCs where appropriate. All measurements are written in C, for performance and portability across a range of hardware platforms.

SamKnows performance tests do not incorporate any third party commercial or free or open source (F/OSS) code. Some tests do however dynamically link to F/OSS libraries.

All tests support IPv4 and IPv6.

3.1 Speed Tests

Measures the download and upload speed of the broadband connection in bits per second. The transfer is conducted over one or more concurrent HTTP connections (using the GET verb of download and the POST verb for uploads).

In the download speed test the client will fetch a portion of a 1GB binary (non-zero, randomly generated) payload hosted on an HTTP server on the target test node. The content is discarded as soon as it is received.

In the upload test the client will generate the payload itself (using /dev/urandom as a non-blocking source of random content) to send to the server. The measure of throughput may be optionally carried out on the server side (the receiver) in the upload test.

The speed tests (both download and upload) operate for either a fixed-duration (specified in seconds) or a fixed-volume (specified in MB). Where possible, a fixed-duration test is preferred as it will cater well for all broadband access speeds. However, a fixed-volume test may be necessary where predictability of bandwidth usage is desired.

Four separate variations of the test are supported:

- Single TCP connection download speed test
- Multiple TCP connection download speed test
- Single TCP connection upload speed test
- Multiple TCP connection upload speed test

For multiple TCP connection tests we typically recommend that three concurrent connections are used. In some cases (e.g. where the round-trip time between client and server is very high) it may be necessary to increase this.

Factors such as TCP slow start are accounted for through the use of a "warm-up" period. This period begins as soon as the test starts and seeks to establish that the throughput has reached stable rate before starting the real test (which will continue over the same TCP connection(s)). It is important to note that the data transferred in the warm-up period is excluded from the main test results, but it is still recorded separately as a supplementary metric.

The speed test client will record the throughput, bytes transferred and time taken at the end of the test. It may also record these values at multiple intervals during the test. This is commonly used to help characterise the difference between 'burst' and 'sustained' throughput (where transfer speeds may be inflated at the start of a TCP connection).

Clients Supported: Fixed, Mobile, Web based speed test, Smartphone

3.2 Web Browsing

Measures the time taken to fetch the HTML and referenced resources from a page of a popular website. This test does not test against centralised testing nodes; instead it tests against real websites, allowing for content distribution networks and other performance enhancing factors to be taken into account.

Each Whitebox will test ten common websites on every test run. The time taken to download the resources, the number of bytes transferred and the calculated rate per second will be recorded. The primary measure for this test is the total time taken to download the HTML page and all associated images, Javascript and stylesheet resources.

The results include the time taken for DNS resolution. The test uses up to eight concurrent TCP connections to fetch resources from targets. The test pools TCP connections and utilises persistent connections where the remote HTTP server supports them.

The test may optionally run with or without HTTP headers advertising cache support (through the inclusion or exclusion of the "Cache-Control: no-cache" request header). The client advertises the user agent of Microsoft Internet Explorer 10.

Clients Supported: Fixed, Mobile

3.3 Video Streaming

This generic streaming test can be configured to model the characteristics of a variety of voice and video protocols. For the purpose of the video streaming test, the intention is to simulate an end user streaming a constant bitrate video from the web. The test seeks to characterise the user's quality of experience by looking for stability in throughput and looking for buffer underrun events.

The test is conducted over TCP. The client and server components negotiate the test parameters at the start of each test.

A three second playout buffer is configured and the client will attempt to download data from the server at the maximum rate necessary to ensure that this buffer is never empty. A separate thread is reading data from this buffer at a fixed rate, looking for buffer underruns (which would manifest themselves to users as a pause in video). The client will record the time to initial buffer, the total number of buffer underruns and the total delay in milliseconds due to these underruns.

The configured bitrate used in this test will often vary according to the quality of video that the customer is seeking to test.

Clients Supported: Fixed, Mobile

3.4 Voice Over IP (Jitter)

This test uses a fixed-rate stream of UDP traffic, running between client and test node. A bi-directional 64kbps stream is used with the same characteristics and properties (i.e. packet sizes, delays, bitrate) as the G.711 codec.

The client initiates the connection, thus overcoming NAT issues, and informs the server of the rate and characteristics that it would like to receive the return stream with.

The standard configuration uses 500 packets upstream and 500 packets downstream.

The client records the number of packets it sent and received (thus providing a loss rate), and the jitter observed for packets it received from the server. The server does the same, but with the reverse traffic flow, thus providing bi-directional loss and jitter.

Jitter is calculated using the PDV approach described in section 4.2 of RFC5481. The 99th percentile will be recorded and used in all calculations when deriving the PDV.

Clients Supported: Fixed, Mobile

3.5 UDP Latency and Packet Loss

Measures the round trip time of small UDP packets between the Whitebox and a target test node. Each packet consists of an 8-byte sequence number and an 8-byte timestamp. If a packet is not received back within two seconds of sending, it is treated as lost. The test records the number of packets sent each hour, the average round trip time of these and the total number of packets lost. The test will use the 99th percentile when calculating the summarised minimum, maximum and average results on the Whitebox.

As with the availability test, the test operates continuously in the background. It is configured to randomly distribute the sending of the echo requests over a fixed interval, reporting the summarised results once the interval has elapsed.

Typically 600 samples are taken per hour, distributed throughout the hour. If the line is busy then fewer samples will be taken. A higher sampling rate may be used if desired.

Clients Supported: Fixed, Mobile, Web based speed test, Smartphone

3.6 UDP Latency and Loss Under Load

This test seeks to characterise the change in latency and packet loss whilst the line is heavily loaded.

To do this, the latency under load test relies on the existing UDP Latency/Loss test and the download and upload speed tests. It first launches the speed test, and then immediately after it has launched it begins sending UDP datagrams to the

target server for 30 seconds, with packets spaced 500ms apart (and a 3 second timeout). Once the speed test has completed the UDP datagrams cease being sent and the test records the mean, minimum and maximum round trip times in microseconds. The number of lost UDP packets was also recorded.

The test is conducted separately when download and upload speeds are being tested, allowing us to characterise how upstream load affects latency/loss differently to downstream load.

Clients Supported: Fixed, Mobile

3.7 UDP Contiguous Loss (Disconnections)

This test is an optional extension to the UDP Latency/Loss test. It records instances when two or more consecutive packets are lost to the same measurement server. Alongside each event we record the timestamp, the number of packets lost and the duration of the event.

By executing the test against multiple diverse servers, a user can begin to observe server outages (when multiple probes see disconnection events to the same server simultaneously) and disconnections of the user's home connection (when a single probe loses connectivity to all servers simultaneously).

Typically this test is accompanied by a significant increase in the sampling frequency of the UDP Latency/Loss client to ~2000 packets per hour (providing a resolution of 2-4 seconds for disconnection events).

Clients Supported: Fixed, Mobile

3.8 DNS Resolution

This test measures the DNS resolution time of a selection of common website domain names. These tests will be targeted directly at the ISPs recursive resolvers. A list of appropriate servers will be sought from each ISP in advance of the tests.

Clients Supported: Fixed, Mobile

3.9 FTP Transfer

This test measures the throughput of performing an FTP transfer to/from a designated FTP server. The test can transfer a file of an arbitrary size (although a 10MB minimum is recommended). In both the download and upload modes the test uses a single TCP connection to perform the transfer.

Clients Supported: Fixed, Mobile

3.10 Peer-To-Peer (BitTorrent)

The peer-to-peer test conducts a BitTorrent transfer of a fixed size or duration. The various test servers around the world act as dedicated seeders and are permanently registered with the BitTorrent tracker, seeding this sample file. Upon startup, the client communicates with the tracker and obtains a list of peers (the seeds), handshakes with each, and then begins transferring the payload. The client

is expressly prohibited from sharing peer-to-peer traffic with other test clients and will only communicate with the super seeders. BitTorrent encryption is not used in this approach.

Metrics that are capture include: (1) the average and peak throughput during the transfer; (2) the number of connections established to peers; (3) the total number of pieces transferred; and (4) the number of TCP connections that were reset during transfer.

The total throughput would broadly match the results of the downstream speed test in a network without traffic shaping. The presence of TCP connections being reset may also indicate the presence of other forms of traffic management.

Clients Supported: Fixed, Mobile

3.11 Email Relaying

This test measures how long it takes for an email to be sent via the ISP's public SMTP relay and to reach a third-party (SamKnows-hosted) email server. Each email sent by the Whitebox has a timestamp embedded within it of when the email was sent. Once the Whitebox has sent an email, it records the message ID in a temporary file to be checked later. Every 10 minutes after, the Whitebox will contain the SamKnows mail server and attempt to find each of the message IDs listed from its temporary file. Upon finding a matching email, the timestamp found in the email is subtracted from the current timestamp in order to find the total time for delivery.

Should an email not be found on the mail server after 3 days, the test is marked as having failed.

Clients Supported: Fixed, Mobile

3.12 You Tube Measure

The YouTube test is an application-specific test, supporting the streaming of video and audio content from YouTube using their protocols and codecs.

The test begins by seeking out the most popular video in the user's country. This is achieved by fetching a list of the most popular YouTube videos from a central SamKnows server. The central list of videos is refreshed once every 12 hours using the YouTube API. We filter for videos that are at least 60 seconds in length and have an HD quality variant. Note that by interacting with the YouTube API from a central location we can ensure that every probe is delivered the same list of videos.

The test running on the probe will now fetch the YouTube web page for the most popular video, and parse the Javascript contained within the page. Within this Javascript is held a list of all of the encodings of the video in question and the content server hostname. By making this request from the probe we ensure that the test is receiving the same content server as the user would if they were using a desktop computer on the same connection.

The test will then connect to the content server (using whatever server YouTube would normally direct a real client on the same connection to) and begins streaming the video and audio. MPEG4, WebM, Dash (adaptive) and Flash video codecs are supported. Although the adaptive codec is supported, the test does not actually adapt its rate; we stream at full rate all of the time, which provides for reproducibility.

The test parses video frames as it goes, capturing the timestamp contained within each video frame. After each frame we sample how much realtime has elapsed versus video time. If video time > realtime at a sample period, then an underrun has not occurred. Otherwise, one has occurred.

The test downloads 10 seconds of audio and video at a time, with a buffer of 40 seconds. So on startup, the test will immediately download (at full speed) 40 seconds of video and audio, and will then download more as required, keeping the 40 second playback buffer full. By default the test will run for a fixed duration of 20 seconds of realtime.

In its default mode of operation the test will capture the 'bitrate that can be reliably streamed' on the user's connection. This is achieved through the following process:

1. Find the fastest recent speedtest result that the probe has completed.

2. As described above, fetch the list of YouTube videos, find the most popular one, and then select the highest bitrate encoding which is less than the fastest speedtest result found in step 1.

3. Attempt to stream this video, for a fixed duration of 20 seconds of realtime. If successful, then the "bitrate reliably streamed" for this instance is the bitrate that we just fetched.

4. However, if a stall event occurs, then we immediately abort the test and retry at the next lower bitrate.

5. If we find a bitrate that we can stream without a stall event occurring then that bitrate is our "bitrate reliably streamed" for this instance.

6. However, if we encounter stalls for every bitrate, then the "bitrate reliably streamed" is zero.

The key outputs from this metric are:

a) The bitrate reliably streamed

b) The startup delay (the time taken to download two seconds of video)

c) The TCP connection time

d) The number of stalls and their duration (this is only applicable if the test is not running in the 'bitrate reliably streamed' mode)

3.13 Netflix measure

The Netflix test is an application-specific test, supporting the streaming of binary data from Netflix's servers using the same CDN selection logic as their real client uses. The test has been developed with direct cooperation with Netflix.

The test begins by calling a Netflix hosted web-based API. This API examines the client's source IP address and uses the existing proprietary internal Netflix logic to determine which Netflix server this user's IP address would normally be served content from. This logic will take into account the ISP and geographic location of the requesting IP address. Where the ISP participates in Netflix's Open Connect programme, it is likely that one of these servers will be used. The API will return to the client a HTTP 302 redirect to a 25MB binary file hosted on the applicable content server.

The test will then establish an HTTP connection to the returned server and attempt to fetch the 25MB binary file. This runs for a fixed 20 seconds of realtime. HTTP pipelining is used to request multiple copies of the 25MB binary, ensuring that if the payload is exhausted before the 20 seconds are complete, we can continue receiving more data. The client downloads data at full rate throughout; there is no client-side throttling taking place.

It's important to note that this 25MB binary content does not contain video or audio; it is just random binary data. However, with knowledge of the bitrates that Netflix streams content at, we can treat the binary as if it *were* video/audio content operating at a fixed rate. This allows us to determine the amount of data consumed for each frame of video (at a set bitrate) and the duration that it represents. Using this, we then have the ability to infer when a stall occurred (by examining when our simulated video stream has fallen behind realtime). The test currently simulates videos at bitrates of 235Kbps, 375Kbps, 560Kbps, 750Kbps, 1050Kbps, 1750Kbps, 2350Kbps, 3000Kbps, 4500Kbps, 6000Kbps and 15600Kbps.

This approach also allows us to derive the 'bitrate reliably streamed', using the same methodology as the YouTube test described above. A small difference here is that we do not need to restart the download at a lower bitrate if a stall is encountered; because the incoming stream of binary data is decoded at a simulated bitrate, we can simply recompute the playback characteristics of the same network stream at a different bitrate entirely on the client side. This simply means that the test uses a predictable amount of bandwidth, even in cases where stalls occur.

The key outputs from this metric are:

a) The bitrate reliably streamed

b) The startup delay (the time taken to download two seconds of video)

c) The TCP connection time

d) The number of stalls and their duration

e) The downstream throughput achieved

3.14 BBC iPlayer measure

The BBC iPlayer test is an application-specific test, supporting the streaming of video and audio content from iPlayer using their protocols and codecs.

The test begins by fetching a list of the most popular videos from an iPlayer XML API. The most popular video is chosen for playback, on the basis that this is most representative of what users will be watching at the time. Moreover, if there are iPlayer caches present in the ISP's network then this content is more likely to be cached there.

The XML manifest for this video is then fetched from the BBC's webservers. This contains paths to all of the different bitrates that this video is encoded at, and the different CDNs that serve them. At the time of writing BBC iPlayer uses multiple CDNs to serve content. By having the probe directly fetch the XML manifest we can ensure that any decisions the BBC make (e.g. "ISP X should always be served by CDN Y") are followed by our test. The test parses the XML manifest file, honouring the priority assigned to each CDN and building an ordered list of available bitrates.

At this point the test can begin to fetch video content from the content server. This is currently achieved over RTMP, mimicking the iPlayer web browser client. The test parses video frames as it goes, capturing the timestamp contained within each frame. After each frame we sample how much realtime has elapsed versus video time. If video time > realtime at a sample period, then an underrun has not occurred. Otherwise, one has occurred.

As with the YouTube and Netflix tests, the client is configured to start testing at the highest supportable bitrate and then step down if and when stalls occur. This allows us to identify the 'bitrate reliably streamed'.

The key outputs from this metric are:

a) The bitrate reliably streamed

b) The startup delay (the time taken to download two seconds of video)

c) The TCP connection time

d) The number of stalls and their duration (this is only applicable if the test is not running in the 'bitrate reliably streamed' mode)

e) The downstream throughput achieved

3.15 Multicast IPTV

This test measures the performance of multicast IPTV carried over RTP, which is commonly used by ISPs worldwide. Four aspects of the multicast delivery mechanism are measured:

- The success and performance of IGMP join/leave instructions (analagous to switching channels)
- The success and performance of retransmission requests
- The jitter observed in the multicast stream

- The level of packet loss on the multicast channel
- Consecutive packet loss events on the channel, including the number of packets and their duration

The test client connects to a multicast test channel as if it were a normal client. Once it has successfully joined the channel it receives the streams for a fixed duration and uses the received stream to calculate the metrics above.

The test is capable of working with SD, HD and UHD streams.

3.16 Cross-traffic detection (Inline / pass-through)

A key benefit to the Whitebox approach is the fact that 'cross-traffic' (other traffic in the participant's home) can be accounted for. This means we can avoid running tests when the user is using their connection, resulting in (a) cleaner results for us and (b) a happy participant (because their use of the Internet is not being interrupted).

Participants are instructed to connect their wired devices via the Whitebox and leave their wireless devices unchanged.

Prior to and between tests, a threshold manager service monitors the inbound and outbound traffic across the WAN interface of the Whitebox to calculate if a panellist is actively using the Internet connection. The threshold for traffic is set to 64kbps downstream and 32kbps upstream. If these thresholds are breached prior to the test starting or between tests, the test will be delayed for a minute and the process repeated. If the connection is being actively used throughout, this pause and retry process will occur up to 5 times before the entire test cycle is abandoned.

A similar process is performed for wireless clients. Wireless users are not asked to make any changes. As with the wired approach, measurements are not conducted when there is wireless activity detected. Wireless activity is determined by passively monitoring the traffic from the user's wireless SSID(s). There are two techniques used to determine the user's wireless SSID:

1) Perform a scan for wireless networks in the vicinity of the Whitebox. Search for an access point that has a MAC address adjacent to the MAC of the LAN interface on the volunteer's CPE. In a user's home environment, this is typically a combined modem/router/WAP. This takes advantage of the fact that most CPE use similar MACs on their Ethernet and WiFi interfaces. This provides significantly improved confidence in high density wireless environments (like apartment blocks).

2) Where no adjacent wireless MAC is found, the Whitebox falls back to the old approach of choosing the device with the strongest signal, whereby the SamKnows Whitebox passively monitors the strongest nearby wireless network for traffic.

Once an SSID has been identified, the Whitebox passively monitors all traffic that the SSID exchanges and records volume information. Note that it does not matter if the wireless network is encrypted; the Whitebox does not need to join

the wireless network, it simply cares about volumes of data (it makes a conservative assumption that all wireless traffic is destined for the Internet).

If a wireless AP is broadcasting multiple SSIDs on the same channel, then the Whitebox will catch traffic from all, because they will use the same or adjacent MAC addresses. It does not matter if the user has hidden their SSID or encrypted their wireless network; the Whiteboxes are simply passively monitoring packet volume and do not need access to the data contained within the packets.

The wireless monitoring process described above is repeated in both the 2.4GHz and 5GHz channels, for applicable Whitebox models.

3.17 Cross-traffic detection (Out-of-band)

Some ISPs may use services that require the user to connect devices directly to the CPE, meaning that the inline approach described above is not suitable. In those cases we can still support cross-traffic detection by interrogating the CPE out-of-band. We can do this using UPnP, SNMP, HTTP or any other protocol (custom development may be required).

[DOCUMENT ENDS]