AD INSERTION

STORAGE REQUIREMENTS AND CACHING WHITE PAPER

ARRIS™

TABLE OF CONTENTS

©ARRIS 2014. All rights reserved.

INTRODUCTION

Linear ad-insertion servers have as their primary function the scheduling and streaming of ad spots. To facilitate efficient and convenient accessibility to ad content, these servers must also offer a significant amount of local storage.

1. How large should ad storage be? How large is large enough?
2. How much additional preload bandwidth will occur for smaller than ideal storage?
3. How do storage requirements scale? How do requirements change from smaller systems to larger systems?

AD STORAGE – STORAGE CAPACITY LIMITS, PRELOAD BANDWIDTH, AND CACHING

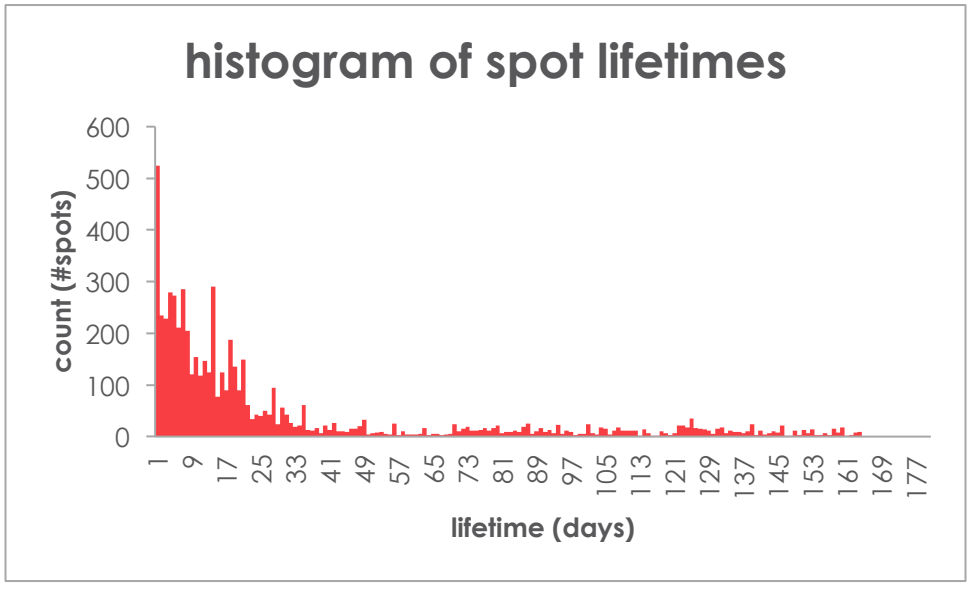
An ad-insertion system is loaded with additional new ad spots on a fairly regular and consistent basis. Ideally, these spots would be stored locally on the ad-insertion server for as long as they might be needed. Consequently, the only network bandwidth required for transmitting ads to the server would be that required to deliver brand new ads. This is the baseline, can-never-be-lower, bandwidth required to support an ad-insertion system. However, the storage capacity of any server is of course finite, so if ads were never deleted and new ads were constantly being added, the storage system would eventually fill up and no more ads could be added. To avoid this storage saturation scenario, already-stored ads must eventually be deleted to make room for new ads. Which ads should be deleted? Ideally, only the ads that will never be aired again would be deleted, but realistically this cannot be ensured. Some ads will be removed that will need to be used again and so these will eventually have to be re-loaded onto the system, thus adding an additional bandwidth burden beyond the new-ads-only baseline. In order to minimize this additional load, the spot removal algorithm should attempt to remove only the stalest/least-needed ads, i.e. the server's software should be programmed to make optimal, automated spot-removal choices. This is in effect an ad-spot caching algorithm.

©ARRIS 2014. All rights reserved.

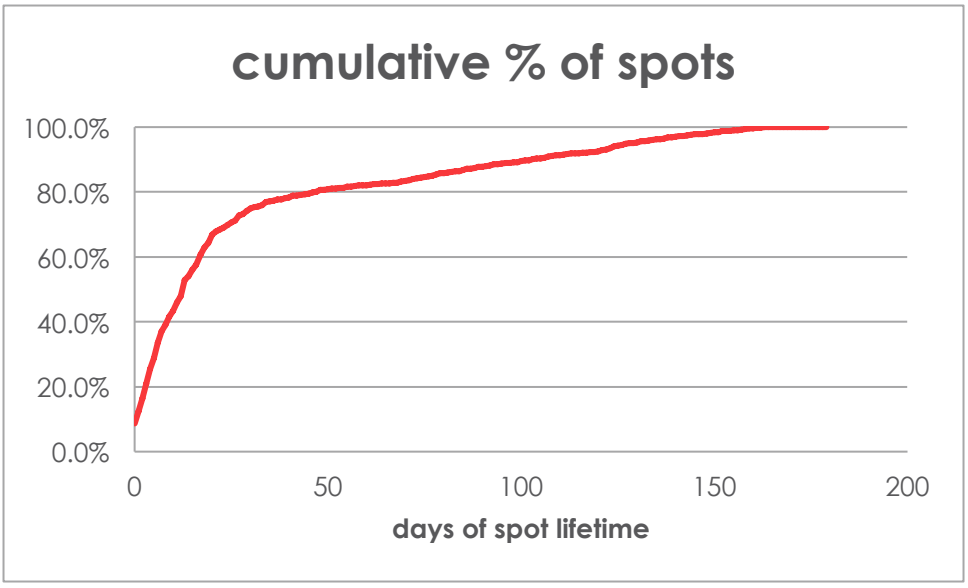
In addition to the design of the spot removal (caching) algorithm, the provisioning of storage capacity is also crucial to reaching an optimal storage vs. bandwidth tradeoff - the larger the storage capacity, the longer the average storage period for spots and the higher the probability that a deleted spot won't be needed again (or soon) and thus the lower the needed write bandwidth above and beyond the baseline. So, the four factors driving the performance of an ad storage cache are spot lifetimes, network bandwidth, spot removal algorithm, and storage capacity.

AD-SPOT LIFETIME

A TV advertisement can be very short-lived – a few days to a few weeks – or it can be regularly played for a much longer time – say a few months – or it can even be brought back for short seasonal runs every year. The histogram below illustrates an example lifetime profile of ad-spots. This histogram results from an examination of ads streamed from an ad server supporting a large market – over 2500 channels – during a window of approximately 180 days. Following the histogram is a second chart showing the cumulative profile of ad lifetimes, showing that 80% of all the unique ads played in that market had a lifetime of 45 days or less. This is fairly typical – other examples from other markets have shown similar results.



©ARRIS 2014. All rights reserved.



CONVENIENCE OF LOCAL STORAGE; AND OTHER TRADEOFFS

A relatively large local storage of ad content offers benefits not only in reduced network bandwidth load because of the reduced need to re-load previously played ads, but also in the opportunity to re-verify the content of an ad long after it has played. Advertisers occasionally wish to verify with the operator, long after the fact, that their ad content was actually and correctly loaded and stored, and never corrupted afterwards. As long as this content still resides on the server that streamed the ad, it is much easier to verify the content and convince the advertiser it was correct when streamed.

So, it would be desirable to hang onto all ads for an extended period, perhaps as long as a year. However, this desire must be weighed against the cost, the impracticality, and the reliability risk incurred by large storage arrays. The desire for long-term verification is only one of the factors that must be traded-off in the design and provisioning of an ad server's storage system.

The tradeoffs:

1. Storage costs and challenges (e.g. reliability, footprint, power, …)

©ARRIS 2014. All rights reserved.

2. Network costs and challenges (e.g. congestion, upgrade difficulty, complexity, …)
3. Spot management costs and challenges (e.g. complexity, support, …)

WHAT CONSTITUTES A 'SPOT'?

For the purposes of this paper, a 'spot' is the video and audio encoding of a unique advertising message in a specific encoding format, prepared and stored for eventual streaming delivery. On the other hand, a 'spot-set' encompasses all the encodings and formats that are stored and streamed for a unique advertisement. For example, a single 30-second SD 'spot' may require 14MB of storage, while an HD spot that is a differently-encoded version of the same ad and thus a separate and distinct 'spot' may require 42MB of storage. Together, these two spots that represent the same unique advertisement constitute a 'spot-set' that takes up 56MB of storage.

The typical encoding population of a 'spot-set' will differ from one operator or market to another, and even potentially from one advertisement to another. One operator may tend to have two encodings for an ad (e.g. one SD and one HD), while another may have three or more. The use of 'spot-set' as an invariant metric allows us to more easily compare and combine statistics from different operators and installations.

In the discussion that takes place in the remainder of this paper, references to 'spot-sets' are referring to spot-sets as defined above. Otherwise, 'spots' refers to unique encodings; and 'spots (SD-eq)' is the measure of storage space or bandwidth expressed in standard 30-second SD-equivalent spot sizes.

CACHE IMPLEMENTATION

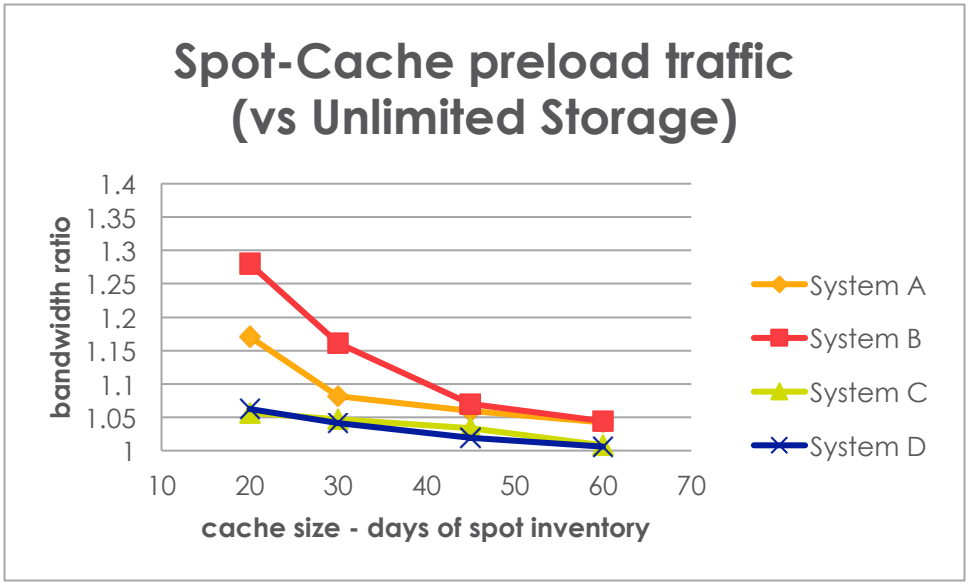
The local ad storage on an ad server has limits that must be managed. Not only is the available storage space finite, but also limits such as Table-of-Contents entries or total number of spots. The scope of this paper is limited to the management of actual storage space, through caching.

©ARRIS 2014. All rights reserved.

A storage array in an ad server will have a determinant usable spot storage space, after accounting for redundancy measures, indexing, and other overheads. A pre-determined "full" threshold is set beyond which the total used storage is not allowed to go. This threshold is generally a percentage number that will be set at something less than 100% of usable storage. When the "fullness" of the storage meets or exceeds this value, the caching algorithm will kick in and remove one or more spots to make room for new spots as they come in. The caching algorithm implements a classic LRU (least-recently-used) selection for removal. This in effect will remove short-lived spots that have stopped playing and won't be needed again and long-life spots that haven't been used recently, making room for new spots that are scheduled to be used in the very near future.

The effect of caching on the preload bandwidth of the system is illustrated in the graph below, which was derived from statistics of four actual ad-insertion sites in the field. The four sites are from three different operators in four different markets. These systems cover a wide range of sizes, in terms of channel counts – approximately 2500 channels, 500 channels, 700 channels, and 750 channels. In the chart, the cache size (x-axis) is given in "days of spot inventory" which just reflects the number of unique spot-sets that are accessed in that number of days (like 'spot-sets', 'days of inventory' is an invariant metric used to make inter-operator comparisons easier and more useful). The bandwidth ratio (y-axis) is the ratio of total preload traffic divided by the baseline preload traffic (recall that baseline traffic is from new ads only, as if storage size were unlimited), thus measuring the additional bandwidth required to re-load ads that were removed from the cache but then needed again. This bandwidth ratio thus reflects the effectiveness of the cache at a given size – a smaller ratio means a more effective cache because the bandwidth premium is less. As shown in the graph, the bigger the cache, the more effective it is at reducing the preload bandwidth requirements.

©ARRIS 2014. All rights reserved.

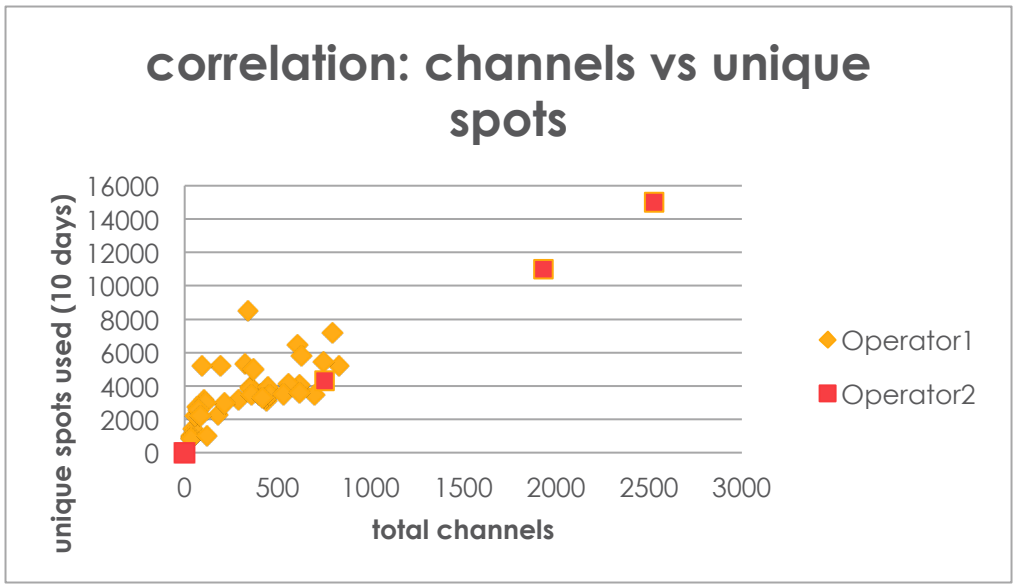


Note that, although for smaller cache sizes the cache effectiveness varies significantly across the four systems (1.06 to 1.28 at a cache size of 20 days), their effectiveness improves substantially and converges quickly at a cache size of around 45 days. The bandwidth ratio at 45 days ranges from just 1.02 to 1.07, meaning that the additional bandwidth above and beyond the idealized baseline is well below 10%.

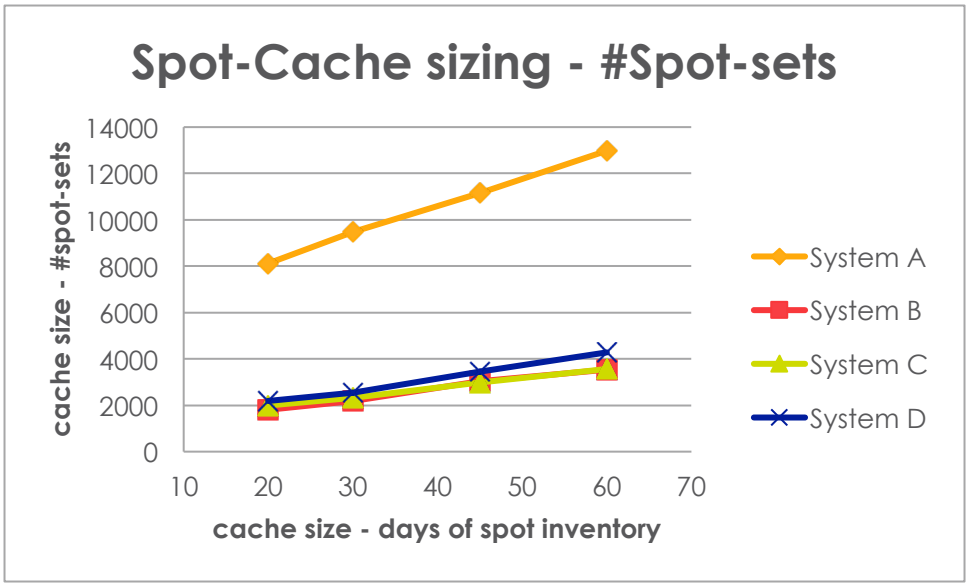
SCALING SPOTS AND STORAGE WITH SYSTEM SIZE

So, how big is a '45-day spot inventory' anyway, assuming this is how large we want our cache to be? Well, for starters, that depends on the system size and the operator. The number of unique spot-sets in a 45-day inventory tends to be greater with larger systems (more channels) - see the 10 day chart below:

©ARRIS 2014. All rights reserved.



The actual 45-day spot-set inventory for the four field systems we're referencing in our analysis ranged from 2973 to 11159 spot-sets, as shown here:

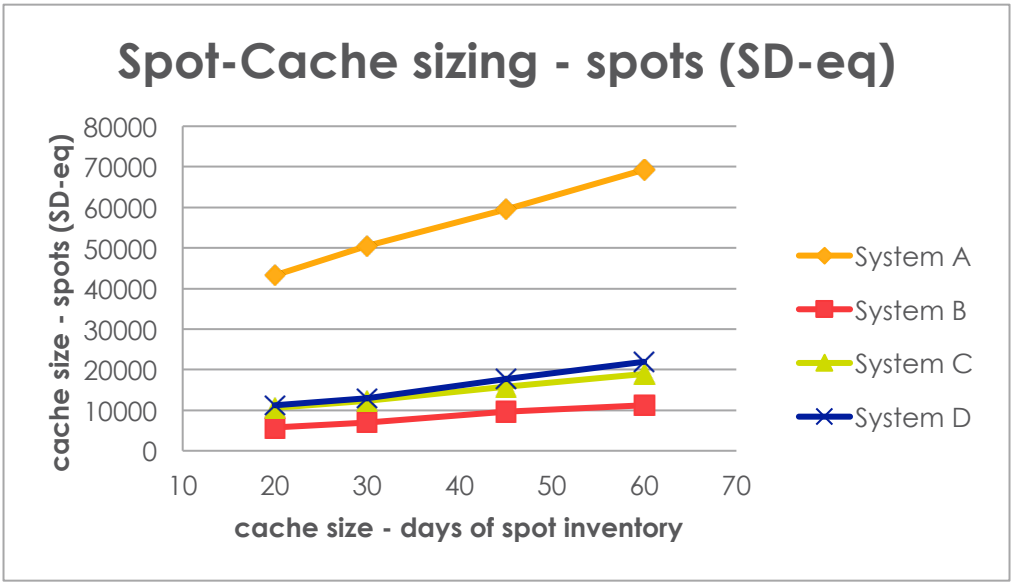


However, the spot-set in one system can be a different size than the spot-set of another. The spot-set factor, i.e. the average number of individual spots (unique encodings) that make up a spot-set, can differ from one operator or deployment to the next. For the field systems we're analyzing here, the spot-set factor ranges from about 1.8x to 2.6x. Sizing the cache

©ARRIS 2014. All rights reserved.

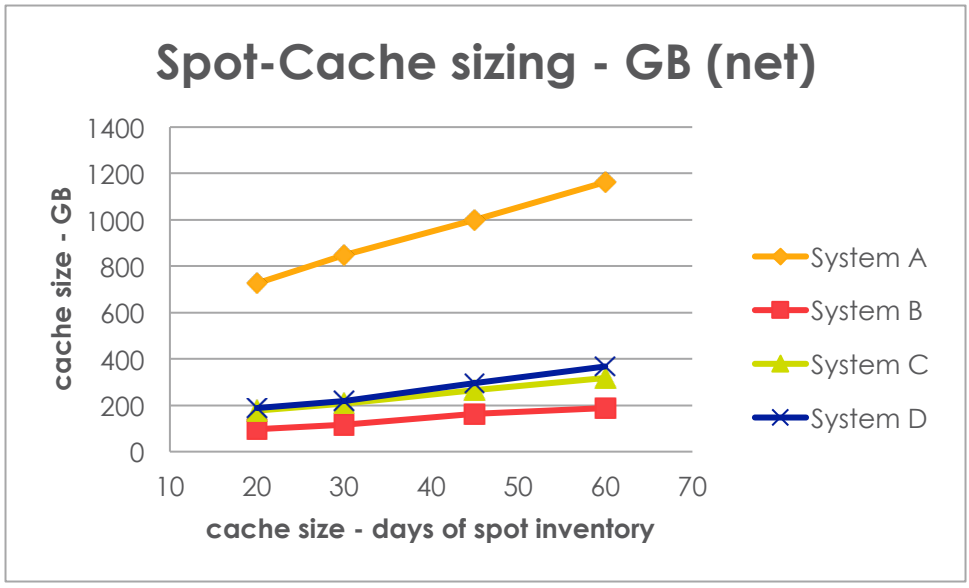
in number of spots can be done by simply scaling the spot-set count by the spot-set factor.

Bear in mind that spots alone are insufficient to fairly measure and compare cache sizes because spots tend to be encoded at different bit-rates – e.g. SD vs HD, and HD bit-rates can differ across operators and/or content providers . So, the invariant metric we use for normalizing spot counts is SD-equivalents, where a standard SD-equivalent spot is a 3.75 Mbps encoding. So, finally, we have the following graph showing cache sizes in SD-equivalent spots. Note that a 45-day cache corresponds to about 60,000 SD-eq spots for the largest system and about 10,000 SD-eq spots for the smaller system:



Now, we can convert to GBytes if desired. Note from the graph below that the larger system 45-day cache requires about 1.0 TB of net storage, while the smallest system 45-day cache requires about 165 GB net storage. Using a fully redundant storage system, for example a mirroring RAID-10 method, one might implement the larger system with 2.0 TB of raw storage (10 x 200GB SSDs) and the smaller system with 400 GB of raw storage (2 x 200GB SSDs). Keep in mind that these numbers assume an average spot length of 30 seconds. Additional storage should be reserved for long-form content.

©ARRIS 2014. All rights reserved.



Conclusions – answers to the original questions

1. A cache sized for the equivalent of a 45-day spot inventory is sufficient capacity for any ad-insertion system.

2. A 45-day-equivalent cache will incur a bandwidth premium of less than 10% over the ideal (unlimited storage).

3. The actual net physical storage required for a 45-day-equivalent cache depends on the operator (average spot-set factor and HD bit-rates); the average spot length; and the unique spot profile (how many unique spots over a 45-day period, which tends to scale with system size i.e. number of channels).

RECOMMENDATIONS

Linear ad-insertion servers have as their primary function the scheduling and streaming of ad spots. To facilitate efficient and convenient accessibility to ad content, these servers must also offer a significant amount of local storage. As demonstrated by the analysis above, and given the impracticality of an unlimited storage space, an effective local storage size for caching ads is the equivalent of 45 days of unique ad inventory.

Given a caching architecture as described, some ads that are re-used after being idle for long periods of time will have been deleted from the local cache storage and so will need to be loaded again onto the server. Therefore, every operator and every deployment should make use of an archival back-store to keep ad content long-term and from which the ad

©ARRIS 2014. All rights reserved.

servers at the edge can re-load ads as necessary. A centralized storage array for archiving ad-content is an attractive option for operators. Such an array with a capacity of several terabytes could store hundreds of thousands of spots, covering a year or more of spot inventory.

For smaller caching ad-server deployments, a 45-day-equivalent storage cache will require up to 200 to 400 GB net (not including RAID and other overheads). Medium-sized systems will be well-served with approximately 500 GB to 1.0 TB of net storage. The largest systems could require 1 to 2 TB of net storage and will likely be clustered systems. For example, the ARRIS XMS-1U server with internal SSD storage matches ideally the needs of small systems, supporting up to 250 channels of streaming and 1.6 TB raw storage (RAID-10 redundancy yields a net cache storage size of 800 GB). Likewise, the largest systems supporting channel counts in the thousands can be implemented using the ARRIS XMS-Flex server with internal SSD storage that yields up-to 4 TB raw cache storage (2 TB net). Comparing the storage capacities of these two servers with the requirements reflected in the analysis above will show that these capacities include significant headroom for growth and variability.

Caching mechanisms are offered in both of the ARRIS products mentioned above, by way of the "auto-delete" function, which implements the LRU algorithm used in the analysis above, along with accommodations for preserving space for yet-unused preloaded ads.

©ARRIS Enterprises, Inc. 2013 All rights reserved. No part of this publication may be reproduced in any form or by any means or used to make any derivative work (such as translation, transformation, or adaptation) without written permission from ARRIS Enterprises, Inc. ("ARRIS"). ARRIS reserves the right to revise this publication and to make changes in content from time to time without obligation on the part of ARRIS to provide notification of such revision or change. ARRIS and the ARRIS logo are all trademarks of ARRIS Enterprises, Inc. Other trademarks and trade names may be used in this document to refer to either the entities claiming the marks and the names of their products. ARRIS disclaims proprietary interest in the marks and names of others. The capabilities, system requirements and/or compatibility with third-party products described herein are subject to change without notice.