Rundum erneuert und fit für die nächsten 10 Jahre
Egg, 12. Juli 2016: *Mit der Version 2016/VII steht ArchivistaVM mit komplett neuer Technologie zur Verfügung. Dabei gibt es zwei gute Neuigkeiten: Erstens, die neue Version ist sowohl in der Webansicht als auch auf der Konsole zu 100% kompatibel zu den bisherigen Versionen und zweitens der technologische Unterbau enthält keine Abhänigkeiten mehr zu Drittprodukten. Im nachfolgenden Blog wird aufgezeigt, warum gerade der zweite Punkt von essentieller Bedeutung ist.*

Aus Proxmox 1.x wird ArchivistaVM
ArchivistaVM ist im Jahre 2009 als Fork zu Proxmox 1.x entstanden. Schritt für Schritt wurden dabei bisher Abhängigkeiten entfernt, sodass ArchivistaVM deutlich schlanker daherkommt als dies bei den Produkten von Proxmox der Fall ist. Zum Vergleich: Die aktuelle Version von Proxmox umfasst 700 MByte. Bei ArchivistaVM sind es 60 MByte in der freien Version und gute 100 MByte in der kommerziellen Version (mit Web-Browser und X-Server, bei Proxmox nicht enthalten).
Natürlich ist die Grösse der ausgelieferten ISO-Datei alleine das Mass aller Dinge nicht. Die Unterschiede zwischen Proxmox und ArchivistaVM sind in der Zwischenzeit beträchtlich. Die Ursache dazu liegt in den Versionen 1.x von Proxmox, wurden diese doch mit **EmbPerl** realisiert.
EmbPerl war der Versuch, Perl in der Art und Weise, wie PHP arbeitet, zu implementieren. Dies bedeutet, die Applikation wird in HTML-Seiten eingebettet ausgeliefert. Im Unterschied zu PHP, das eine hohe Verbreitung erhielt, war dies bei EmbPerl jedoch zu keiner Zeit gegeben. Um es salopp zu sagen, **EmbPerl** wird seit 2012 leider nicht einmal mehr weiterentwickelt.

Darum passt ExtJS für ArchivistaVM nicht

Proxmox löste das Problem dahingehend, dass die gesamte Web-Applikation neu programmiert wurde, und zwar mit **ExtJS.** Dabei handelt es sich um ein JavaScript-Framework. Der visuelle Teil der Applikation wird dabei (vereinfacht gesagt) mit Komponenten realisiert, die zwar sehr mächtig, aber auch sehr umfangreich sind. Hinter **ExtJS** steht die Firma Sencha, die ExtJS sowohl unter einer GPL-Lizenz als auch unter einer kommerziellen Lizenz vertreibt. Allerdings werden die Bedingungen der GPL-Version immer mal wieder geändert, wer ohne die kommerzielle Version von ExtJS entwickeln möchte, wird mehr Hürden als Hilfen bei der Arbeit erfahren.
Aus diesen Gründen war bereits vor Jahren klar, dass ExtJS nicht in ArchivistaVM Einzug halten wird. Und zwar nicht nur, weil nicht klar war/ist, inwiefern die GPL-Version von **ExtJS** brauchbar ist, sondern auch, weil über 10 oder 20 Jahre nicht vorhergesagt werden kann, ob das Framework an sich Bestand haben wird. Des weitern lässt aber auch die Grösse des Frameworks aufhorchen, ExtJS 6.0.1 kommit mit 531 MByte an Sourcen, weit über 44000 Dateien sowie fast 2.7 Millionen Codezeilen daher. Kurz und gut, ein solches 'Monster' passt für ArchivistaVM einfach nicht.

EmbPerl kleiner, aber nicht mehr zeitgemäss

Im Vergleich dazu ist EmbPerl ein wahrer Winzling, 'gerade' einmal 123911 Codezeilen umfasst das Paket. Aber, und dies ist die schlechte Nachricht, EmbPerl arbeitet nur in Verbindung mit Apache 2.2 stabil. Alle Versuche, EmbPerl auf die aktuelle Version 2.4 von Apache zu migrieren, scheiterten bislang. Zwar ist EmbPerl zu 100% OpenSource, doch die Pflege von weit über

100'000 Codezeilen nicht ganz trivial. Aus diesem Grunde war klar, EmbPerl musste ersetzt werden.

Evaluation von JavaScript-Bibliotheken

Über einen Zeitraum von ca. 1 bis 2 Jahren wurden verschiedene Alternativen getestet und erörtert. Naheliegend wäre gewesen, anstelle von ExtJS ein anderes modernes JavaScript-Framework zu verwenden, doch liessen sich von kommerziellen 'Monstern' abgesehen kaum brauchbare Alternativen finden. Am ehesten hätte **W2ui** gepasst, siehe dazu **http://w2ui.com.** Mit ca. 16'000 Codezeilen für ein ganzes Set von Komponenten ist diese Bibliothek sicher nicht überdimensioniert.

Problematisch bei der Migration von ArchivistaVM war aber, dass im bisherigen Code viele Daten an einen internen Daemon übermittelt werden. Neben einem neuen Frontend (JavaScript) hätte somit auch ein grosser Teil der Applikation serverseitig (in Perl oder einer anderen Sprache) realisiert werden müssen. Vorallem aber wäre es unmöglich gewesen, das bisherige Web-Interface auch nur annährend beizubehalten. Mit anderen Worten, ArchivistaVM wäre nicht mehr ArchivistaVM gewesen.

Nativer Perl-Code als Lösung für ArchivistaVM

Manchmal benötigt ein Entscheid etwas länger. Nach vielen Tests und Überlegungen reifte der Entschluss, den 'alten' fragmentierten' Code in nativen Perl-Code (ohne EmbPerl) zu übertragen. Dies bedeutete zwar, dass sämtliche der ca. 80 Dateien angepasst werden mussten, doch bot sich dabei auch gleich die Gelegenheit, den Code zu überarbeiten. Damit wurde es mögich, die Codebasis von über 40'000 Zeilen auf gute 10'000 Zeilen zu reduzieren, analog dazu sank auch die Anzahl der Dateien auf einen Viertel.

Der Aufwand für diese Migration belief sich auf einige hundert Stunden (inkl. der gesamten Evaluation) an Arbeit. Jede andere Lösung hätte einen weit höheren Aufwand erfordert, von denn neu eingegangenen Abhängigkeiten ganz zu schweigen. Es mag sein, dass ein solcher Blogeintrag auf den ersten Blick nun nicht wahnsinnig spannend erscheinen mag. Selbstverständlich wäre es marketingtechnisch einfacher hinzuschreiben: Komplett neue Applikation, und dann tausend neue Features anzuführen. Doch, ist es nicht so, wer die gleiche Applikation zweimal schreibt, hat der seine 'Hausaufgaben' wirklich gut gemacht?

Zum Abschluss sei nochmals erwähnt, dass mit der **neuen Version 2016/VII von ArchivistaVM 100% Kompatiblität zum bisherigen sehr stabilen**

Unterbau von ArchivistaVM erzielt werden konnte. Daher kann die Version 2016/VII problemlos produktiv eingesetzt werden. Die neue Version von ArchivistaVM steht ab sofort sowohl in der **freien Version Mini** wie auch für die Kunden mit Wartungsvertrag zur Verfügung. In diesem Sinne, auf die nächsten 10 Jahre mit ArchivistaVM.
P.S: ArchivistaVM steht bislang auf jeder ArchivistaBox mit Intel/AMD-Prozessor zur Verfügung, selbst bei ArchivistaDMS (Home-Button wählen). Im Unterschied zu den bisherigen Versionen von ArchivistaVM, die nicht unter ARM-Prozessoren liefen, stellt dies bei der aktuellen Version von ArchivistaVM keine Hürde mehr dar. Die Zukunft wird zeigen, ob ArchivistaVM auch Einzug auf den ARM-basierten ArchivistaBoxen halten wird.


Facebook

Twitter