Virtual Machine live Migration Optimization

תוכן עניינים:

1. הקדמה: הצגת הנושא VM live migration.
2. הצגת דרכים לייעול VM live migration:
   a. Bandwidth Optimization.
   b. Power Consumption Optimization.

מקורות:

1. A survey on virtual machine migration and server consolidation frameworks for cloud data centers
2. Evaluation of Delta Compression Techniques for Efficient Live Migration of Large Virtual Machines (Svärd et al.)
3. Analysis of virtual machine live-migration as a method for power-capping | SpringerLink

הקדמה

שרת - תוכנה או מחשב פיזי שנותן שירותים ללקוחותיו. בהרצאה זאת נעסוק בשרתים פיזיים.
דאטה סנטר (מרכז נתונים) - אתר פיזי בו מנוהלים באופן מסודר שרתים רבים. קיומם של מרכזים אלה הם בלבו של מחשוב הענן המודרני.
מחשוב ענן (cloud computing) - מודל בו ספקים מציעים שירותי תוכנה וחומרה בהתבסס על חיבור רשתי, לרוב באמצעות האינטרנט, אל מרכזי הדאטה של מציע השירות. לרוב, שירות ענן יכלול יכולות של כוח עיבוד, אחסון מידע ותעבורה רשתית.

מדוע מחשוב הענן חשוב? ניעזר בדוגמה: דוגמה מודרנית לשימוש בשירותי ענן ניתן לראות באופן ההקמה של רבים מהסטארטאפים בשנים האחרונות: באופן מסורתי, חברות שעשו שימוש בשרתים הקימו ותחזקו אותם באופן פנימי בחברה. אילוץ זה הקשה באופן ניכר על חברות בצמיחה, שכן כמות המשתמשים גדלה באופן מהיר, וכדי לתמוך בהם נדרש להתקין עוד מעבדים, התקני אחסון ותשתיות רשת באופן תכוף. במקום זאת, סטארטאפים כיום נוטים לשכור את אותן יכולות מחברות אחרות, כדוגמת אמזון, גוגל או מיקרוסופט. כך הסטארטאפים פנויים להתעסק בפיתוח המוצר עצמו, ואילו החברות המציעות את שירותי הענן מתמקצעות בפיתוח התשתיות.

אתגרים במודל הענן:
- ניהול מידע והרצת תוכניות שנכתבו על ידי לקוחות שונים.
- שמירה על סביבה סטרילית לכל לקוחה.
- הגנה על המידע הפרטי אותה היא מחזיקה בשרת.

פתרון **לא** טוב לאתגרים: לכל לקוח יוקצה שרת פיזי בו התהליכים שלו ינוהלו.
יתרון: סטריליות ופרטיות מיטבית.
חיסרון: יקר מאוד ואין ניצול יעיל של המשאבים, מכיוון שמשתמשים שעושים שימוש בשירותי ענן משנים באופן דינאמי את הצרכים שלהם, ולעתים הצרכים משתנים אפילו לאורך שעות היום. במצב כזה, שרת פיזי אשר נעשה בשימוש על ידי לקוח יחיד לא יהיה בניצול מרבי של משאביו לאורך רוב הזמן.

פתרון נבחר: כל לקוח ירוץ בתוך מכונה וירטואלית, ומכונות וירטואליות רבות ירוצו על גבי כל שרת פיזי.
יתרון: פותר את החסרונות שהוזכרו לעיל.
יתרון בונוס: הבחירה במכונות וירטואליות מאפשרת גם לנייד לקוחות בין שרתים פיזיים. אם לדוגמה, שרת א' מריץ בזמן נתון מכונות שדורשות מעט משאבים, ניתן להעביר את המכונות הוירטואליות לשרת ב' אשר גם בו רצות מכונות, ולכבות את שרת א'. באופן זה ננצל את השרתים ברשותנו באופן מיטבי ונחסוך צריכת אנרגיה מיותרת.

מיגרציה - תהליך העברת המכונות הוירטואליות בין שרתים.
מיגרציה קרה - השיטה הפשוטה ביותר לביצוע מיגרציה.
שלבים:
- עצירה של המכונה הוירטואלית
- העתקת הזיכרון ויתר המשאבים שבשימוש אל שרת היעד
- ולאחר מכן התחלת הרצה של המכונה במיקומה החדש.

יתרון: המימוש פשוט יותר והתהליך כמעט תמיד מצליח.

מיגרציה חמה - תהליך העברת המכונה מתבצע מבלי לעצור את ריצת המכונה כלל, או לפרק זמן קצר ביותר.
יתרון: יש פגיעה מינימלית בחווית המשתמש עבור הלקוחות המשתמשים במכונה הוירטואלית, שכן היא ממשיכה לפעול לאורך התהליך.

שיטות של מיגרציה חמה:

1. שיטת pre-copy: מעתיקים את כל הזיכרון וה-CPU state של ה-VM. מכבים את המכונה בשרת המקור ומדליקים אותה בשרת היעד.
2. שיטת post-copy: מעתיקים state וזיכרון מינימלי של ה-VM. מכבים את המכונה בשרת המקור ומדליקים בשרת היעד. ממשיכים להעתיק את יתר הזיכרון משרת המקור אל היעד לאחר מכן.

שימוש ב-post-copy עלול לגרום לירידה בביצועים גם לאחר העברת המקל משרת המקור אל היעד. הסיבה העיקרית היא שיש צורך לטפל בהרבה page faults שנגרמים בשרת יעד, וגוררים תקשורת מול שרת המקור באופן תדיר. שיטה זאת גם מורכבת יותר למימוש, ולכן שיטות האופטימיזציה שנציג יתמקדו ב-pre-copy.

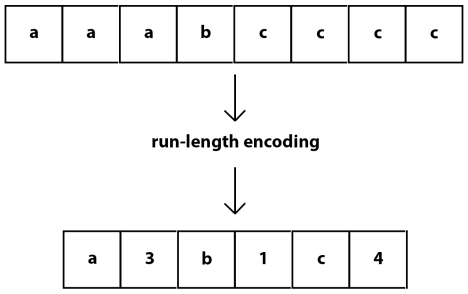
הרחבה על שיטת pre-copy:

- במהלך המיגרציה, תוכן דפי זיכרון שכבר הועברו עלול להשתנות כי ה-VM עדיין פועל.
- התהליך איטרטיבי - באיטרציה הראשונה מועברים כל דפי הזיכרון של ה-VM. בכל איטרציה עוקבת מועברים הדפים שהתלכלכו (שהשתנו) במהלך האיטרציה הקודמת.
- כשכמות הדפים שהתלכלכו קטנה מספיק, מפסיקים את ה-VM בשרת המקור, מעבירים את המידע האחרון שנותר להעביר (דפים שהתלכלכו וה-CPU State) ומדליקים את ה-VM בשרת היעד.

המאמרים אותם אנו מציגים עוסקים באופטימיזציה של תהליך המיגרציה בשני אספקטים שונים: ראשית נסקור אופטימיזציה אפשרית של רוחב הפס בעת ביצוע תהליך מיגרציה חמה, ולאחר מכן נעסוק באופטימיזציה של ניהול אספקת החשמל בהקשרי מיגרציה.

Bandwidth Optimization

רוחב פס (bandwidth) - כמות המידע הרשתי שניתן להעביר ביחידת זמן נתונה. באופן לא מקצועי, נקרא "קצב" או "מהירות" הגלישה. לדוגמה: 100Mpbs.
תהליך ה-pre-copy עשוי להיכשל אם קצב העברת המידע בין שרת המקור אל היעד נמוך מקצב לכלוך הדפים בין איטרציות. לכן, נרצה לצמצם את כמות המידע הנשלח (ובהתאמה, כמות המידע שנשלח ביחידת זמן תהיה קטנה מרוחב הפס של הרשת).
לשם כך ניתן להיעזר בהנחה שכמות המידע שהשתנה קטנה משמעותית משטח הדפים הכולל בהם שוכן המידע. לכן, אם נעביר עבור כל דף רק את הדלתאות (ההבדלים) בין האיטרציה הקודמת לאיטרציה הנוכחית, נוכל לצמצם משמעותית את נפח התקשורת הרשתית. לשם כך ניתן להיעזר ב-xor: אם נבצע xor בין הדף שהעברנו בפעם הקודמת למצבו הנוכחי כעת, נקבל 0 בכל ביט אשר לא השתנה ו-1 אם הביט השתנה. בדפים בהם התבצעו שינויים נקודתיים, תוצאת ה-xor תהיה דף שברובו המוחלט מכיל אפסים, ולכן נוח לביצוע דחיסה. כותבי המאמר השתמשו בדחיסת RLE (run-length encoding) אשר מקבצת רצפים של בתים זהים לכדי בית ומספר חזרות, כפי שמופיע באיור:



אם כן, בכל העברה חוזרת של דף, יתבצע בשרת המקור xor בין המידע שהיה באותו דף בהעברה הקודמת למידע שקיים כעת, עליו תבוצע דחיסת RLE וכך המידע יישלח לשרת היעד. שרת היעד מצדו יקבל את המידע, יבצע פתיחה של הדחיסה ויבצע xor בין תוכן הדף שקיים אצלו בזיכרון לזה שהתקבל כעת.

כדי לבצע את ה-xor, נדרש משרת המקור לשמור את התוכן הישן של דפים. נדרוש ממבנה הנתונים שישמור מידע זה שיהיה מהיר (כדי לא לעכב את ההעברה) ושיהיה לו שטח חסום, למקרה בו קבוצת הדפים החמים גדולה במיוחד. לשם כך בחרו כותבי המאמר להשתמש במנגנון caching של two way set.

באשר לעבודה עתידית, הציעו כותבי המאמר 2 שיפורים אפשריים: 1. ביצועי האלגוריתם מושפע באופן ניכר על ידי גודל ה-cache, שכעת נקבע באופן סטטי על ידי החוקרים. הם מציעים לחקור דרכים להגדיר ולשנות באופן דינמי גודל זה על מנת להגיע לביצועים מיטביים. 2. בנוסף, יש מוטיבציה רבה להעביר את הדפים שנמצאים בקבוצה החמה כמה שיותר מאוחר, אך האלגוריתם כעת מתעלם מכך. לכן, בעתיד הכותבים יחקרו מנגנונים לסידור מחדש של סדר העברת הדפים ע"פ תדירות הכתיבה אליהם.

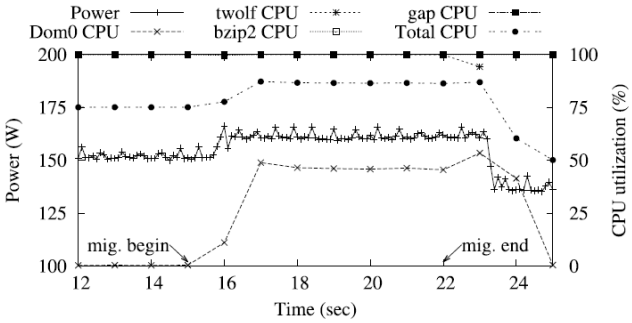
מבוסס על :
[Evaluation of Delta Compression Techniques for Efficient Live Migration of Large Virtual Machines (Svärd et al.)]

Power Consumption Optimization

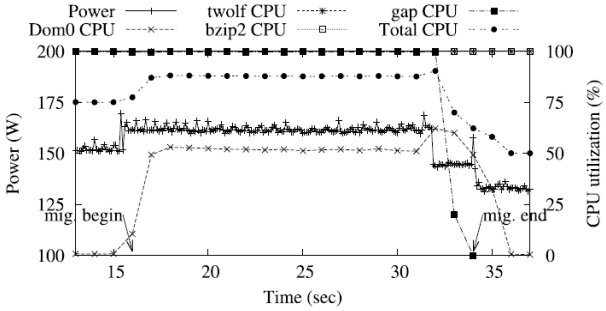
צריכת האנרגיה של הdata centers אינה זניחה. בממוצע 30% מהDCs מנצלים לכל יותר 10-15% מהמשאבים שלהם. ניצול חלקי של משאבים אלו מביא לעלות תפעולית וצריכת אנרגיה גבוהים להפליא.
החוקרים ערכו את הניסויים הבאים וגילו כי בזמן live migration יש עליית דרמטית בצריכת האנרגיה של השרת השולח.

הגדירו את מערכת ניסוי שמשתמשת בארבע linux VMs שרצות על מחשב עם 4 ליבות, קצב שעון 2.67 GHz על Xen hypervisor עם ארכיטקטורת x86.
בנוסף הגדירו שרת מקור ושרת יעד. כאשר על שרת המקור מריצים את הVMs, ובשרת היעד לא רצות VMs, אלא הוא מחכה לבקשת נדידה.
לכל VM הקצו processor וירטואלי ו600 MB זיכרון. בנוסף בכל VM הריצו אחת מהbenchmark workloads.
כאשר twolf הוא הכי פחות אינטנסיבי בזיכרון, וgap הוא הכי אינטנסיבי בזיכרון בעיקר בעקבות כתיבות.
בpre copy live migration, ביצעו השוואה בין נדידה של twolf ונדידה של gap.

Processor configuration (Intel Core i7-920)			
Num. of cores	4	L2 cache	256 KB per core
Clock freq.	2.67 GHz	L3 cache	8 MB per processor
ISA	x86_64	TDP	130 W
9 DVFS states		Hyperthreading disabled	
System configuration			
Main memory	4 GB	GPU	Nvidia 6600GT
Hypervisor	Xen 3.4.0	Guest kernel	Linux 2.6.18.8



(a) Migrating twolf (CPU intensive)



(b) Migrating gap (memory intensive)

בתרשים a ניתן לראות את צריכת האנרגיה של השרת בנדידת twolf, ובתרשים b את צריכת האנרגיה של השרת בנדידת gap. נדגיש כי Twolf השתמש ב3MB זיכרון בעוד שgap השתמש ב200MB זיכרון.
בהתאם נדידת twolf הסתיימה אחרי בערך 8 שניות בעוד שנדידת gap הסתיימה לאחר כ20 שניות. ההבדל הגדול נובע בעיקר כתוצאה מכתיבות מרובות של הgap לזיכרון, מה שגורם להרבה דפים מלוכלכים בזיכרון והעברתם בהתאם.
כמו כן ניתן לראות כי צריכת האנרגיה של השרת עלתה ברגע שהתחיל להעביר את המידע, ויורדת קצת אחרי שמסתיימת הנדידה לצריכה נמוכה יותר ממה שהייתה לפני הנדידה כצפוי.

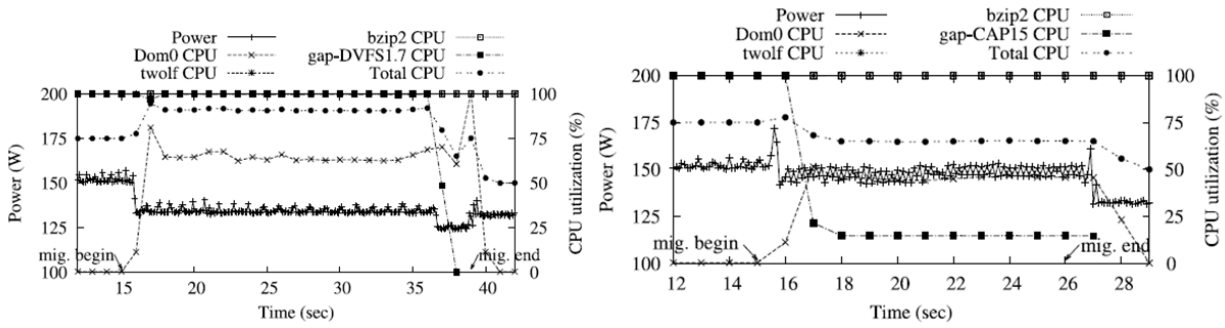
המסקנה היא שבזמן הנדידה יש עלייה בצריכת האנרגיה. בDC שמבצע המון נדידות חמות צריכת האנרגיה הופכת למשמעותית וגוררת נטל כלכלי ועולמי.

DVFS וPower Capping הן שתי שיטות שמטרתן להוריד מיידית את צריכת האנרגיה של השרת בזמן live migration, תוך כדי עמידה בתנאי הSLA ופגיעה מינימלית בQoS.

<u>שיטה ראשונה:</u> שימוש בטכנולוגיה חומרתית **DVFS = Dynamic voltage frequency scaling**.
טכנולוגיית DVFS משתמשת בתלויות בין המתח, התדירות ומהירות המעבד כדי לשנות את מחזור השעון (clock rate) של הCPU על מנת להגביל אותו לגבול מסוים שנבחר ובכך להגביר את היעילות בשימוש האנרגיה בDC.
חסרון של שיטה זו: שיטה זו מוגבלת לCPUs שהארכיטקטורה שלהם תומכת בשינויים בקצב השעון שלהם.
עבור מעבדים שכן תומכים בשינוי קצב השעון שלהם, ייתכן ולא יהיה אפשר להגדיר קצב שעון שונה בין הליבות. לכן הורדת קצת השעון בליבה מסוימת תדרוש הורדת קצב השעון אצל שאר הליבות מה שיגרום לירידה בביצועים של שאר הVMs בשרת. לכן שיטה זו אינה יכולה להיות פתרון ארוך טווח של הגבלת ההספק.

<u>שיטה שנייה:</u> **POWER CAPPING**
במתזמן של הVM בשרת (credit scheduler) הגדירו ערך CAP לכל VM. ערך זה קובע זמן CPU מקסימלי שנותנים לVM באינטרוול זמן. ערכו של הערך נע בין 0 ל100 כאשר 0 הוא הערך הדיפולטי שאומר שאין מגבלה של זמן CPU על הVM. כאשר מאתחלים נדידה עבור VM כלשהו יש לעדכן את הCAP שלו למגבלה שרוצים.

להלן ניסוי שערכו על קבוצת מעבדים לא רוויה בעבודה כדי להשוות את ההבדלים בין שתי הגישות ואיך כל אחת משפיעה על צריכת האנרגיה של המעבד:



(a) DVFS to 1.7GHz (b) CAP value to 15

בגרפים לעיל בדקו את צריכת האנרגיה והמשאבים של CPUs שונים בזמן live-migration.

בגרף a השתמשו בגישה הראשונה שמשתמשת בטכנולוגיית DVFS שהורידו את קצב השעון ל1.7GHz (בקצב שעון הכי נמוך שאפשר להגדיר).
בגרף b השתמשו בגישה השנייה והגדירו את CAP value ל15 עבור הVM שעושים לו migration.
ניתן לראות שבשתי הגישות צריכת האנרגיה וניצול הCPU צונחים ברגע שמתחיל הlive-migration. כמו כן ניתן לראות כי הגישה הראשונה עם הDVFS הוריד את צריכת האנרגיה יותר מאשר הגישה השנייה שמשתמשת בCAP. כמו כן קביעת הCAP value לVM שמצבעים לה את הנדידה, משפיעה דרמטית על הCPU utilization שלה. לעומת זאת בשיטת הDVFS יש ירידה בCPU utilization של כל הVM שבשרת. הDVFS מוריד את קצב השעון לכל הVMs שעובדים על המעבד לכן צריכת האנרגיה הכוללת במעבד יורדת משמעותית כי כל שאר הVMs גם מוגבלים. ניתן לראות כי צריכת האנרגיה הכוללת ירדה לכ135.6 Watts שזה 10% פחות. לעומת זאת בשיטת הCAP מגבילים רק את הVM שעושים לה migration, כלומר שאר הVM עובדים כרגיל ולכן יש פחות השפעה על הצריכה הכוללת של האנרגיה במעבד. אכן צריכת האנרגיה הכוללת של המעבד ירדה ל148.3 Watts שזה 3% פחות.

בנוסף, החוקרים גילו כי ככל שהמעבד בשימוש יותר גבוהה כך גישת הDVFS חוסכת יותר אנרגיה. מנגד היעילות של גישת CAP יורדת ככל שמס' הVMs שיכולות לחסום CPU גדול יותר ממס' הליבות.
לסיכום, שיטת DVFS מגיעה לביצועים טובים יותר מאשר שיטת הCAP, אך היא משפיעה גם על VMs אחרים שעובדים על אותו המעבד ונתמכת רק במעבדים שיש להם את הטכנולוגיה הנ"ל.
לפיכך נעדיף להשתמש בשיטת הCAP כל עוד הירידה בצריכת האנרגיה שמספק הערך CAP שמגדירים מקובל עלינו.

לסיכום, שיטת הpower capping מושכת תשומת לב רבה מהאקדמיה והתעשייה. שתי השיטות שציינו לעיל מנסות להגביל את צריכת האנרגיה של השרת מקור שמבצע את הנדידה לגבול מסוים שמוגדר ע"י בעלי הDC.
אולם, הpower capping schemes עדיין לא שלמות, צריך להמשיך לעצב אותן כך שנצליח לחזות מדויק יותר את צריכת האנרגיה ולשלוט טוב יותר בביצועים.
לדוגמא, נרצה שהערך של הCAP לא ישתנה עד שהpre-copy וצריכת המשאבים של הנדידה יסתיימו. אך זמן זה אינו דטרמיניסטי. לכן מגדירים את הCAP לשלוט גם בקצב צריכת המשאבים של הנדידה וזמן הסיום שלה. הגדרה זו אינה מיטבית ויש למצוא את הדרך הטובה ביותר להגדרת הCAP value של הVM.

מבוסס על :
[Analysis of virtual machine live-migration as a method for power-capping | SpringerLink]

Questions about the topics

1. What is the main benefit of using live (hot) migration over cold migration?
    a. Cold migration is harder to implement and prone to more bugs.
    b. Cold migration typically requires more bandwidth and more stable network connection between source host (server) and guest host.
    c. Live migration causes the virtual machine to pause for a shorter period of time.
    d. Live migration achieves better compression ratios.
2. Alice decided to implement live migration, but chose to use just RLE instead of XBRLE in both source and guest hosts (she didn't XORed the content of the last page sent with the current state). In her implementation:
    a. Destination host decodes pages successfully, since XORing the deltas is an optimization, and not necessary for a successful migration of a page.
    b. Destination host fails to decode the page, since binary data cannot be sent over the network without XOR.
    c. Destination host decodes pages successfully and works even better, because XOR is not invertible.
    d. Source host fails to encode the page. It is impossible to encode data using RLE if most of our data is not 0.
3. A downside of using XBRLE is:
    a. XBRLE does not compromise any aspect of the migration process.
    b. The compression algorithm requires a significant amount of processing time, compared to other compression methods.
    c. Usually, all or most of the memory pages belong to a program's hot page set.
    d. The algorithm requires additional space, since the source host has to store old content of already sent pages.
4. What are the problems DVFS technology and power capping are addressing?
    a. The jump in power consumption during live migration.
    b. On average the resource utilization in DCs is low.
    c. DC maintenance is very expensive.
    d. All answers are correct.
5. After going over the expenses of the last quarter, the CEO of a server company decided that more savings should be made in electricity expenses. What solution would we offer to the CEO that would not harm QoS?
    a. Controlled shutdown of the servers frequently.
    b. Using DVFS technology to lower the clock rate of the servers.
    c. Using a CAP value for each VM that will define a CPU time limit for each VM.
    d. Combination of DVFS and CAP for maximum saving of energy consumption.