# Review 19: Deep VULMAN: A Deep Reinforcement Learning-Enabled Cyber Vulnerability Management Framework

**Paper: https://arxiv.org/abs/2208.02369v2**

פינת הסוקר:

המלצת קריאה מעדן ומייק: מומלץ לעוסקים בתחום ה-Reinforcement learning ו-Cybersecurity

בהירות כתיבה:

ידע מוקדם: Reinforcement Learning

יישומים פרקטיים אפשריים: Cyber Vulnerability Management

פרטי מאמר:

פורסם בתאריך: 3.10.22, בארקיב.

הוצג בכנס: Artificial Intelligence (cs.AI); 2022

תחומי מאמר:

למידת חיזוקים עמוקה (Deep Reinforcement Learning)

ניצול משאבים אופטימלי

תכנון מספרים שלמים (integer programming)

יישומים:

טיפול יעיל יותר בחולשות במערכות תוכנה תחת אילוץ משאבים מוגבלים

מבוא:

ארגונים מותקפים מדי יום על ידי תוקפים שמנסים לנצל חולשות ברשת הארגונית במטרה לגרום נזק לארגון. כדי להילחם בתופעה, על אותם ארגונים לנסות לזהות את החולשות ולתקן אותן לפני שהן מנוצלות על ידי התוקפים. מציאת חולשות הינה משימה קשה, שכן מספר החולשות הזמינות הולך וגדל כל יום שכן חולשות חדשות מתגלות לעיתים קרובות. קושי נוסף הינו מספר המשאבים המוגבל המוקצה לתיקון חולשות (למשל מחסור באנשי IT שמבצעים שדרוגים למערכות ישנות בעלות חולשות). כתוצאה מכך, חולשות מסוימות לא מטופלות בזמן ומנוצלות על ידי תוקפים שמצליחים לגרום נזק רב לארגון.

תרחיש זה מעלה את הצורך בתהליך ניהול אוטומטי של מציאה ותיקון חולשות. תהליך זה מתחיל על ידי תוכנות שסורקות את הרשת אחר חולשות מוכרות (למשל כאלו הנמצאות במסד הנתונים NVD). סריקה כזו תחזיר כפלט דוח המכיל את החולשות שהתגלו ומאפיינים נוספים כגון: המכונה עליה נמצאה החולשה, תיאור החולשה וכדומה. צוות אבטחת הסייבר של הארגון (CSOC) משתמש בדוח זה כדי להקצות משאבים לצורך תיקון החולשות. תהליך זה של הקצאת משאבים מבוצע היום בצורה ידנית או על ידי מערכת חוקים. שיטות כאלו אינם מספיקות שכן הן לא מתחשבות במגוון גורמים כגון מאפיינים של המערכת עליה נמצאו החולשות. כתוצאה מכך, יתכן מצב שבו משאבים הוקצו לטפל בחולשות פחות קריטיות.

במאמר שנסקור היום, החוקרים מציעים שיטה אשר מתבססת על שילוב של למידת באמצעות חיזוקים (RL) ותכנות מספרים שלמים ( IntPr -integer programming) כדי לתת מענה לסוגיה זו. סוכן ה-RL אחראי על הקצאת המשאבים לטיפול בחולשות בזמן אמת. מודל IntPr יקבל את מספר המשאבים מסוכן ה-RL כאילוץ ובהינתן האילוץ וסט החולשות הקיימות, ינסה לבחור את החולשות הקריטיות ביותר שיטופלו על ידי המשאבים שנבחרו.

השיטה המוצעת נקראת Deep VULMAN ומטרתה לזהות את החולשות החשובות ביותר שצריכות טיפול בזמן אמת תחת אילוצי משאבים וחוסר ודאות לגבי חולשות חדשות שיכולות להתגלות בעתיד.

הסבר על השיטה:

הסביבה:

כדי ללמוד את המדיניות האופטימלית להקצאת המשאבים יש צורך בסביבת סימולציה אשר תהיה דומה ככל האפשר לסביבה האמיתית אותה יראה הסוכן במערכת ארגונית לאחר סיום האימון. יצירת סביבה כזו הינה משימה קשה שכן דאטה המתאר תהליכי טיפול בחולשות במערכות תוכנה (כגון סוג החולשה, הציון שלה, מספר המשאבים והזמן שהוקצו לטיפול בה) לא נגיש לציבור הרחב. כדי להתמודד עם הבעיה, החוקרים שיתפו פעולה עם צוות CSOC בארגון גדול (לא צוין שם הארגון מתאמי פרטיות המידע) ומהדאטה שנאסף מארגון זה הכותבים יצרו סימולציה של הסביבה. במצב אמת לא ידוע לנו מתי חולשה מסוימת תתגלה על ידי סורקי החולשות ולכן החוקרים סימלצו 3 קצבים שונים של הגעת חולשות למערכת: איטי, בינוני וגבוה. הסביבה מחליפה ביניהם באופן רנדומלי ובנוסף בכל נקודת זמן חולשות שונות נדגמות מהדאטה ההיסטורי.

סוכן RL:

סוכן RL הינו אחראי על הקצאת המשאבים לטיפול בחולשות. בעיית הקצאת משאבים במערכות תוכנה טומנת בה מרכיב של חוסר ודאות. למשל אם נקצה 8 מתוך 8 המשאבים שלי לטיפול ב-8 חולשות ביום מסוים ויום למחרת יתגלו 10 חולשות חדשות שחייבות טיפול מיידי נהיה בבעיה. הכותבים מציעים לפתור בעיה זו באמצעות טטניקת של RL.

בעיית הקצאת משאבים ממודלת באמצעות תהליך קבלת החלטות מרקובי (MDP). בתהליך זה יש צורך בהגדרה של משתנים הבאים: מצב, תגמול, פעולה ומטריצת המעברים בין מצבים, שהוגדרו במחקר זה באופן הבא:

מצב - המצב מייצג את המידע שזמין לסוכן בזמן t. כאן מידע זה מכיל מאפייני החולשה כגון: קריטיות החולשה, רמת החשיבות של המכונה עליה התגלה החולשה, האם החולשה התגלתה על ידי מערכות ההגנה של הארגון וציון ה-CVSS שלה.

פעולות - הפעולות שהסוכן יכול לבצע בזמן t:להקצות משאבים נוספים או לא להקצות משאבים כלל. כלומר, הפלט הינו מספר בין 0 למספר המשאבים המקסימלי.

מטריצת המעברים - מידול ההסתברויות של מעבר ממצב למצב בבעיה זו הינה משימה קשה ולכן החוקרים החליטו לא לעשות זאת בצורה מפורשת.

תגמול: הדרך של הסוכן לדעת האם הפעולה שביצע בזמן t הינה טובה או רע. בבעיה זו התגמול הוגדר להיות לפי הנוסחה הבאה:

כאשר קריטיות החולשה שתוקנה ו- הוא מספר המשאבים שנוצלו בזמן t. שתי המשקולות ו- הינן פרמטרים השולטים בחשיבות היחסית של כל תגמול.

הסוכן בו השתמשו הכותבים משתמש באלגוריתם PPO לאיתור המדיניות האופטימלית להקצאת המשאבים. אחת הסיבות לשימוש בגישה זו היא גודל עצום של מרחב המצבים והפעולות האפשריות שלא מאפשר שמירתם בזיכרון. כתוצאה מכך אי אפשר להשתמש בשיטות סטנדרטיות כגון Q-Learning בשביל למידת Q(S,a). כדי להתגבר על סוגיה זו החוקרים משתמשים ברשת נוירונים לשערוך של Q(S,a) בצורה מקוונת.

מודל תכנון מספרים שלמים (integer programming):

בעייה של בחירת החולשות שיטופלו, בעזרת המשאבים שהוקצו על ידי סוכן ה-RL, מנוסחת במאמר כבעיית אופטימיזציה קומבינטורית. משתני אופטימיזציה הם בינאריים כאשר ערך 1 מסמן שהמשאב i נבחר לטיפול בחולשות ו-0 מסמן שמשאב זה לא נבחר. פונקציית המטרה כאן היא האימפקט הכולל על המערכת בעקבות טיפול בחולשות שייבחרו; ואת האימפקט הזה מנסים למקסם. במילים פשוטות אנו מנסים לתקן כמה שיותר חולשות בעלות מאפיינים קריטיים עם המשאבים שהוקצו על ידי סוכן ה-RL.

כעת נעבור לניסוח המתמטי של הבעיה. פונקציית המטרה מוגדרת באופן הבא:

כאשר הינו ערך (חשיבות) של מאפיין של החולשה. כאן I מסמן את מספר המאפיינים של החולשה כאשר J הוא המספר הכולל של החולשות. למעשה פונקציית המטרה של ה״חשיבות הממוצעת״ של כל החולשות שטופלו פר משאב שהוקצה לטיפול.

האילוץ נראה כך:

כאשר הוא הזמן שנדרש לתיקון של חולשה j ו- הוא משך זמן המשאבים, המוקצים באמצעות סוכן RL, שזמינים לנו. המשמעות של אילוץ זה היא לא לאפשר להקצות משאבים ליותר זמן ממה שהם מוקצים לנו. נזכיר כי המטרה כאן היא למצוא קומבינציה של ערכי הממקסמים את פונקצית המטרה ומקיימים את האילוץ.

המחברים פותרים בעיית אופטימיזציה זו באמצעות גישת הנקראת תכנות מספרים שלמים. שיטה זו היא משפחה של שיטות מתמטיות שמטרתן היא לפתור בעיות אופטימיזציה כאשר חלק מהמשתנים הם דיסקרטיים, למשל יכולים לקבל ערכים טבעיים בלבד. בעיות כאלו הן בדרך כלל קשות הרבה יותר מבעיות אופטימיזציה עם אילוצים בהן כל המשתנים הם רציפים.

תרשים כללי של המערכת:

כאשר מגיעה חולשה חדשה, המאפיינים שלה נשלפים ממסד הנתונים הנחוצים להגדרת המצב (state) שלה. מצב זה נכנס כקלט לסוכן ה-RL שמחשב את מספר המשאבים שצריך להקצות בנקודת זמן זו, כדי להתמודד עם כלל החולשות שהתגלו. פלט זה נכנס כקלט למודל התכנון המספרים השלמים שמוציא כפלט רשימה של חולשות הדורשות מענה מיידי. החולשות מתוקנות ותגמול מחושב עבורן מוזרם חזרה לסוכן ה-RL והתהליך חוזר על עצמו עד התכנסות.

סביבת האימון:

החוקרים שיתפו פעולה עם צוות ה-CSOC של ארגון גדול, (מטעמי אבטחה שמו לא נחשף), כדי לקבל גישה לדאטה שלו. הדאטה הכיל מידע על חולשות שהתגלו בארגון במשך שנתיים כגון: תיאור החולשה, CVSS ציון, קריטיות החולשה וכדומה. בנוסף, הדאטה כלל מידע על מכונות ברשת. לבסוף, נאספו גם התראות שעלו מכלי הגנה שונים של הארגון (IDS). כל אלו שימשו את החוקרים ליצירת הדאטהסט של החולשות שעליו ביססו הכותבים את מחקרם. החוקרים הפעילו מנגנון preprocess שכלל מתן ציון נומרי על:

חשיבות מכונה (למשל מכונה שעליה יושב SQL DB תקבל ציון גבוה יותר ממחשב אישי רגיל).

דרגת ההגנה של מכונה (למשל האם קיים אנטי וירוס על המכונה, האם היא מאחורי חומת אש וכדומה).

חשיבות המכונה כלפי הארגון (לא צוין מה זה כולל או איך חושב).

מדאטהסט זה החוקרים יצרו סביבת סימולציה המדמה סביבת CSOC אמיתית. קצב הגעת החולשות למערכת מודל על ידי התפלגות פואסון, התפלגות שנועדה למדל תהליך התרחשות של אירועים בלתי תלויים בקצב הגעה ממוצע קבוע. קצב הגעת החולשות השתנה כל שבוע והיה לאחד מבין המודים הבאים: נמוך, בינוני, גבוה.

נזכיר כי כאשר משאב מסוים נבחר לטיפול בחולשה מסוימת על ידי Deep VULMAN, הוא (המשאב) יהיה תפוס עד שלא יסיים לטפל בה ועד אז אי אפשר להשתמש בו לטיפול בחושבות אחרות.

הישגי מאמר:

הגישה המוצעת הושוותה מול שיטות אחרות של בחירת חולשות שיטופלו. השיטות שנבחרו להשוואה זו הינן: VPSS, VULCON שלפי הספרות מהוות את השיטות הנפוצות ביותר (משך האיטרציה בתיאור של הגרף = שבוע). ההשוואה בין השיטות כאן מתבצעת על בסיס כמות האנומליות המטופלות.

מצב a מתאר את מספר החולשות שנאספו ממכונות בעלי חשיבות גבוה.

מצב b מתאר את מספר החולשות שנאספו ממכונות בעלי חשיבות נמוכה.

מצב c מתאר את מספר החולשות שנאספו ממכונות בעלי חשיבות לארגון (כלומר סובייקטיבית לארגון ולא אובייקטיבית כגון מצב a).

מצב d מתאר את מספר החולשות שנאספו ממכונות שעליהן עלו התראות ממערכות ההגנה השונות.

התוצאות מראות בבירור שבכל שלב ובכל סביבה VULMAN הצליחה לתעדף יותר חולשות קריטיות מאשר השיטות האחרות. כותבי המאמר מצביעים על כך ש-VULMAN השיגה את התוצאות הנ"ל על דאטה אשר לא התאמנה עליו.

בשרטוט הנ"ל ניתן לראות סימולציה של חודש ימים כאשר בכל שבוע קצב הגעת החולשות משתנה. למשל קצב הגעת החולשות בשבוע הראשון נמוך מהשבוע השני. העמודה הכתומה מתארת את הזמן הצפוי הנדרש לטיפול בכל החולשות שהתגלו באותו שבוע והעמודה הכחולה מתארת את מספר המשאבים שסוכן ה-RL הקצה כדי להתמודד עם אותן חולשות. העמודה האדומה מתארת את מספר החולשות שסומנו כקריטיות מתוך סך החולשות באותו שבוע. לבסוף הקו הכחול מתאר מצב תיאורטי בו התפלגות הקצאת המשאבים היא אחידה בין השבועות. אסטרטגיה זו של התפלגות אחידה היא מה שנהוג בשיטות האחרון (VPSS & VULCON).

מהתוצאות ניתן לראות כי בשבוע הראשון VULMAN הקצה פחות משאבים כדי להתמודד עם קצב הגעת החולשות משום שהוא זיהה שקצב זה הוא נמוך ואין צורך בהקצאת כל במשאבים. אסטרטגיה זו השתלמה שכן בהמשך קצב הגעת החולשות התגבר ול-VULMAN היה יותר משאבים פנויים להשתמש ואכן הוא ניצל את רובם כפי שניתן לראות משבוע 3 ו-4.

סיכום:

במאמר הוצגה מערכת VULMAN, המיועדת לזיהוי ודירוג חולשות במערכות תוכנה בארגון כדי שיוכלו לקבל טיפול תחת אילוץ של מספר משאבים (לטיפול החולשות) מוגבל. הגישה המוצעת כוללת שימוש בסוכן RL שמאומן להקצות את המשאבים באופן אופטימלי. בנוסף משתמשים ב-Integer Programming לבחירה את ההתראות(חולשות) שדורשות טיפול מיידי. הניסויים והמידע שנאסף היו בשיתוף עם צוות CSOC בארגון גדול וסביבת הסימולציה שנבנתה מדמה בצורה אמינה את המתרחש במערכת תוכנה של ארגון. הניסויים מראים ש-VULMAN לומד כיצד לקבל החלטות "מושכלות" בסביבה בעלת חוסר ודאות גבוה כמו סביבת CSOC שנובע מיכולתו להקצות משאבים בצורה חכמה יותר מהשיטות האחרות.

שיתוף פעולה: הסקירה נכתבה יחד עם עדן יבין.