Apprentissage d'un modèle comportemental d'une application distribuée pour la détection d'intrusion

Mouna HKIMI

April 28, 2016

Introduction

De nos jours, l'informatique distribuée est devenue omniprésente. Au cours des dernières années, de nombreux systèmes informatiques ont été ciblés par des attaques. Ceci a conduit les recherches à se focaliser sur la sécurisation des systèmes informatiques et la protection des données. La sécurité des applications distribuées n'a pas, malheureusement, connu une évolution comparable. Dans ce résumé nous présentons une solution qui permet d'identifier la présence d'une attaque en se basant uniquement sur l'impact et les modifications qu'elle peut engendrer sur le comportement attendu d'une application distribuée.

1 Systèmes de détection d'intrusion

Un système de détection d'intrusion est un outil utilisé pour surveiller des systèmes hôtes ou des réseaux et détecter la présence d'activités malveillantes.

Pour détecter la présence d'intrusions, un IDS cherche soit à détecter des attaques connues, c'est la méthode dite par scénario ou *misuse detection*. Cette méthode exige une base de données contenant un nombre important d'attaques. La deuxième méthode, dite approche comportementale ou *anomaly detection*, consiste à analyser le comportement de l'application en cours d'exécution et vérifier si ce comportement respecte le comportement de référence de l'application. Dans le cas contraire, il s'agit d'une attaque probable.

La détection d'intrusion dans les systèmes distribués consiste essentiellement en des détections locales de comportements anormaux, puis à une corrélation centralisée des alertes levées par ces IDSs [7]. Toutefois, une telle corrélation repose sur une hypothèse très forte: l'existence d'une horloge globale permettant d'ordonner les alertes produites par les différents

IDSs. Dans notre approche, nous avons relaxé cette hypothèse et supposé que nous ne disposions pas des moyens de créer cette horloge globale, ou qu'elle était particulièrement difficile à mettre en oeuvre dans la réalité. Nous avons développé une solution pour la détection d'intrusions dans des applications distribuées en s'inspirant de quelques solutions généralement utilisées pour modéliser des comportements distribués dans le domaine du test logiciel [2] [1].

2 Inférence d'un modèle de comportement distribué

Dans la littérature, de nombreux travaux [2] [5] [1], ont pour objectif de modéliser le comportement normal d'une application à partir de traces d'exécution mais dans ces travaux le but est en général de vérifier si les comportements observés sont conformes ou pas aux spécifications.
Dans notre étude, le but est d'obtenir une modélisation des comportements sains (en l'absence d'attaquant) et d'utiliser ensuite ce modèle pour déterminer si un nouveau comportement est conforme ou pas. Par hypothèse, un comportement non prévu est considéré comme étant un comportement malveillant.
Le modèle est construit durant une phase d'apprentissage supervisé au cours de laquelle l'application est exécutée à plusieurs reprises. A chaque exécution est collectée une trace qui est composée d'un fichier de log par processus. Le fichier identifie les événements de communication (envoi de message, réception de message) et les événements internes pertinents qui ont été produits séquentiellement sur ce site.
Dans le cas d'une application répartie, l'ensemble des événements d'une trace constitue un ensemble partiellement ordonné par la relation "s'est produit avant" définie par Lamport [4]. Pour chaque exécution, nous exploitons cette information pour construire un automate qui identifie toutes les séquences d'événements possibles qui sont compatibles avec l'ordre partiel observé. Ces automates sont ensuite fusionnés en un seul automate. La procédure de fusion consiste à merger les préfixes communs aux deux automates. A l'issue de la fusion, nous obtenons un automate qui reconnait exactement toutes les traces utilisées pour l'apprentissage.
Parallèlement, nous nous appuyons sur une deuxième technique pour modéliser le comportement observé par un ensemble de prédicats temporels (invariants temporels) [2] sur l'ordre d'occurrence des événements produits par l'application distribuée.
L'originalité de l'approche repose sur le fait que l'automate produit par apprentissage est ensuite compressé et généralisé (grâce à l'algorithme k-tail [3]), introduisant ainsi des comportements non appris. Ces comportements peuvent être des comportements corrects ou incorrects. Les invariants préalablement calculés permettent, durant la phase de détection, de

distinguer ces deux cas de figure. L'approche a été implémentée, et appliquée, d'une part à un exemple simple, permettant de mettre en évidence la corrélation entre une forte généralisation et un faible taux de faux positifs. D'autre part, le processus a été soumis à des tests de passage à l'échelle en réalisant la phase d'apprentissage sur des traces comportant un grand nombre d'événements. L'application choisie pour cela est un système de fichiers distribué nommé XtreemFS [6]. Cette application réelle nous a permis d'analyser les temps de construction du modèle en fonction du nombre d'événements observés et de montrer la corrélation entre la phase d'apprentissage et le taux de faux positifs.

3 Conclusion

Dans ce résumé, nous avons décrit brièvement notre approche conçue pour inférer le comportement d'une application distribuée en se basant sur des traces collectées séparément sans être contraint par la présence d'une horloge globale. Nous avons essayé d'attirer l'attention sur le fait que les outils que nous avons développés sont capables de passer à l'échelle contrairement aux outils existants. Nous avons également montré que cette approche produit un modèle permettant de détecter des attaques. Les travaux que nous menons actuellement se focalisent sur l'apprentissage exhaustif du comportement distribué de l'application XtreemFS afin de mener des tests de détection d'attaques réelles contre cette application.

4 Biographie: Mouna HKIMI

Je suis doctorante à l'université de Rennes 1 et membre de l'équipe CIDRE, INRIA Rennes Bretagne Atlantique. En 2007, j'ai commencé mes études préparatoires a l'Institut Préparatoire aux Etudes Scientifiques et Technique de la Marsa (IPEST) en Tunisie. En 2012, j'ai obtenu un diplôme d'Ingénieur en Informatique de l'Ecole Nationale des Sciences de l'Informatique (ENSI). En 2012, j'ai effectué un stage de fin d'études ingénieur au laboratoire informatique Paris 6, LIP6. En janvier 2013, j'ai obtenu mon diplôme de master spécialité Réseaux et Systèmes Multimedia de l'ENSI. En octobre 2013, j'ai commencé une thèse en sécurité informatique sous la thématique "Détection des intrusions dans les systèmes distribués" au sein de l'équipe CIDRE à Rennes. Mon projet de thèse porte sur le développement d'une solution capable d'inférer le comportement normal d'une application distribuée, qui est ensuite utilisé pour la détection d'intrusion.

References

[1] Ivan Beschastnikh, Yuriy Brun, Michael D. Ernst, and Arvind Krishnamurthy. Inferring models of concurrent systems from logs of their

behavior with csight. In *Proceedings of the 36th International Conference on Software Engineering*, ICSE 2014, pages 468–479. ACM, 2014.

[2] Ivan Beschastnikh, Yuriy Brun, Michael D. Ernst, Arvind Krishnamurthy, and Thomas E. Anderson. Mining temporal invariants from partially ordered logs. In *Managing Large-scale Systems via the Analysis of System Logs and the Application of Machine Learning Techniques*, SLAML '11, pages 3:1–3:10. ACM, 2011.

[3] A. W. Biermann and J. A. Feldman. On the synthesis of finite-state machines from samples of their behavior. *IEEE Trans. Comput.*, pages 592–597, June 1972.

[4] Leslie Lamport. Time, clocks, and the ordering of events in a distributed system. *Commun. ACM*, 21(7):558–565, July 1978.

[5] Davide Lorenzoli, Leonardo Mariani, and Mauro Pezzè. Automatic generation of software behavioral models. In *Proceedings of the 30th International Conference on Software Engineering*, ICSE '08, pages 501–510. ACM, 2008.

[6] XTreemFS Team. XTreemFS. `http://www.xtreemfs.org/`, 2016.

[7] E. Totel, B. Vivinis, and L. Mé. A language driven intrusion detection system for event and alert correlation. *" in Proceedings ot the 19th IFIP International Information Security Conference. Toulouse: Kluwer Academic*, pages 209–224, August 2004.