Proyecto Fin de Carrera
Ingeniería de las Tecnologias Industriales

Seguridad en aeropuertos basada en la Teoría de Juegos

Autor: Alfredo Maza Moreno
Tutor: Andrés Jiménez Losada, Manuel Ordóñez Sánchez

Dpto. de Matemática Aplicada II
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2019

Proyecto Fin de Carrera
Ingeniería de Telecomunicación

Seguridad en aeropuertos basada en la Teoría de Juegos

Autor:
Alfredo Maza Moreno

Tutores:
Andrés Jiménez Losada
Manuel Ordóñez Sánchez

Dpto. de Matemática Aplicada II
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2019

Proyecto Fin de Carrera: Seguridad en aeropuertos basada en la Teoría de Juegos

Autor: Alfredo Maza Moreno

Tutores: Andrés Jiménez Losada, Manuel Ordóñez Sánchez

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2019

El Secretario del Tribunal

A mi familia y especialmente a Begoña

Agradecimientos

Quería agradecer la ayuda prestada en la realización del trabajo a mis tutores y a Begoña por el tiempo y el ánimo que me han dado siempre.

Alfredo Maza Moreno

Sevilla, 2019

Resumen

La seguridad en los aeropuertos es un tema muy actual debido a la gran cantidad de personas que se mueven por ellos y la gran cantidad de comercio que permiten.

Es bien sabido que son centros neurálgicos de comunicaciones entre países y que su inutilización conlleva un serio revés al tránsito de pasajeros y al comercio de mercancías.

Ha habido ataques terroristas de gran alcance sobre todo en personas en los últimos años. Son casos tristemente conocidos por todos.

En este trabajo intentamos dar un método de atenuación o prevención de estos ataques. Para ello planteamos una situación donde hay posibles ataques a servidores web por un lado y ataques físicos a anillos de protección policial en un aeropuerto estándar.

El aeropuerto dispone de sistemas de defensa informáticos y de policías con perros. El problema radica en que el aeropuerto no sabe cómo le van a atacar mientras que los atacantes conocen las defensas que tiene el aeropuerto. Esto se llama juego de información incompleta.

Para nuestro trabajo usamos en particular juegos bayesianos-stackberger para porder hallar una estrategia óptima sea quien sea el ataque en ambos frentes, informático y físico.

Este juego, con tanto volumen de información, pasa por optimizar un programa no lineal. Para resolverlo usamos un algoritmo llamado DBOSS que permite convertir el programa no lineal en varios lineales que pueden resolverse con ayudas de paquetes informáticos estándar. Otra característica es que este algoritmo no presente suboptimalidades con lo cual la solución optima es única.

Abstract

Security at airports is a very current issue due to the large number of people who move by them and the large amount of trade they allow.
It is well known that they are neuralgic centers of communications between countries and that their disablement entails a serious setback to passenger traffic and merchandise trade.
There have been powerful terrorist attacks especially on people in recent years. They are cases sadly known by all.
In this work we try to give a method of attenuation or prevention of these attacks. In order to do so, we raised a situation where there are possible attacks on web servers on the one hand and physical attacks on police protection rings at a standard airport.
The airport has computer defense systems and police officers with dogs. The problem is that the airport does not know how they are going to attack while the attackers know the defenses that the airport has. This is called an incomplete information game.
For our work we use in particular Bayesian games-stackberger to find an optimal strategy for whoever is the attack on both fronts, computer and physical.
This game, with so much information, goes through optimizing a non-linear program. To solve it, we use an algorithm called DBOSS that allows converting the nonlinear program into several linear ones that can be solved with standard software packages. Another feature is that this algorithm does not present suboptimalities with which the optimal solution is unique.

Índice

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

1 INTRODUCCIÓN Y OBJETO

El trabajo fin de grado "Seguridad de los Aeropuertos basada en la "Teoría de juegos Bayesianos de Stackelberg" pretende mostrar la efectividad que tiene el uso de esta teoría para analizar la seguridad

en los aeropuertos y para ello se aplica a un caso real.

Proteger la infraestructura nacional, como aeropuertos, puntos de referencia históricos o una ubicación de importancia política o económica, es una tarea difícil para la policía y las agencias de seguridad de todo el mundo, un desafío que se ve agravado por la amenaza del terrorismo. Dicha protección de ubicaciones importantes incluye tareas como la supervisión de todas las entradas o caminos de entrada y la verificación del tráfico de entrada. Sin embargo, los recursos limitados implican que normalmente sea imposible proporcionar una cobertura de seguridad completa en todo momento. Además, los adversarios pueden observar los arreglos de seguridad a lo largo del tiempo y explotar cualquier patrón predecible en su beneficio. La asignación aleatoria de horarios para patrullar, controlar o monitorear es, por lo tanto, una herramienta importante en el arsenal de la policía para paliar las vulnerabilidades debidas a la previsibilidad en la defensa.

Incluso más allá de proteger la infraestructura, las patrullas aleatorias son importantes en tareas que van desde la seguridad en los campus universitarios y zonas financieras hasta la seguridad marítima o fronteriza.

Este trabajo se enfoca en un agente auxiliar de software desplegado para ayudar a la policía u otras agencias de seguridad a asignar aleatoriamente sus programas de seguridad.

Nos enfrentamos al menos a tres desafíos clave en la construcción de un asistente de software de este tipo.

Primero, el asistente debe proporcionar garantías de calidad en la aleatorización sopesando adecuadamente los costos y beneficios de las diferentes opciones disponibles. Por ejemplo, si un ataque en una parte de una infraestructura causa daños económicos mientras que un ataque en otra podría costar vidas humanas, debemos sopesar las dos opciones de manera diferente, dando mayor peso a la protección de esta última.

Segundo, el asistente debe abordar la incertidumbre en la información que las fuerzas de seguridad tienen sobre el adversario.

En tercer lugar, el asistente debe permitir una interacción de iniciativa mixta con usuarios potenciales en lugar de dictar un horario; el asistente puede desconocer las restricciones del mundo real de los usuarios y, por lo tanto, los usuarios deben ser capaces de configurar el desarrollo de la programación.

2 TEORÍA DE JUEGOS

La Teoría de Juegos es un principio fundacional bien establecido dentro de los sistemas multiagente, cada uno persiguiendo sus propios intereses. Es una metodología formal para analizar interacciones entre jugadores inteligentes: gente, empresas, sistemas informáticos o incluso robots.

Los ejemplos básicos del modelado teórico del juego incluyen simulaciones de los procesos competitivos en economía, política, psicología o biología. Proporciona herramientas para determinar el comportamiento óptimo en ambientes competitivos y los jugadores son grupos interesados, políticos, etc.

En informática, la teoría de juegos se usa durante el modelado de sistemas multi-agentes, algoritmos online o procesos en redes de trabajo.

Formalmente, un juego se refiere a todas las situaciones que involucran a dos o más individuos inteligentes que toman decisiones racionales. Nos basamos en estos fundamentos de la teoría de juegos para razonar acerca de dos agentes, la fuerza policial y su adversario, para proporcionar un método de aleatorización. En los casos en los que la seguridad es muy importante, la Teoría de juegos puede ser muy útil, como por ejemplo el comportamiento de la policía en los aeropuertos. El terrorismo o cualquier acto vandálico de tipo informático en los aeropuertos también puede ser modelado mediante esta teoría.

2.1. JUEGOS BAYESIANOS

Muchas formas de juego suponen que todos los jugadores saben qué juego se está jugando. Específicamente, se asume que la cantidad de jugadores, las acciones disponibles para cada jugador y la recompensa asociada con cada vector de acción son conocimientos comunes entre los jugadores.

Esto es cierto incluso para los juegos de información imperfecta.

Los movimientos reales de los agentes no son de conocimiento común, pero el juego en sí lo es.

En contraste, los juegos bayesianos, o juegos de información incompleta, nos permiten representar las incertidumbres de los jugadores sobre el juego que se está jugando. Esta incertidumbre se representa como una distribución de probabilidad sobre un conjunto de juegos posibles.

Hacemos dos suposiciones.

1. Todos los juegos posibles tienen la misma cantidad de agentes y el mismo espacio de estrategia para cada agente; sólo difieren en sus pagos.

2. Las creencias de los diferentes agentes son posteriores, obtenidas mediante el condicionamiento de un previo común sobre señales privadas individuales.

El segundo supuesto es sustantivo, y volvemos a él en breve. El primero no es particularmente restrictivo, aunque al principio podría parecerlo. Uno puede imaginar muchos otros tipos posibles de incertidumbre que los jugadores pueden tener sobre el juego: cuántos jugadores están involucrados, qué acciones están disponibles para cada jugador y quizás otros aspectos de la situación. Podría parecer que hemos limitado seriamente la discusión al descartar esto.

Sin embargo, resulta que estos otros tipos de incertidumbre pueden reducirse a incertidumbre solo sobre los beneficios mediante la reformulación del problema.

Por ejemplo, en una situación en la que un jugador no está seguro de la cantidad de acciones disponibles para los otros jugadores, podemos reducir esta incertidumbre a acerca de los pagos

acolchando el juego con acciones irrelevantes.

Consideremos el siguiente juego de dos jugadores, en el que el jugador de la fila no sabe si su oponente tiene solo las dos estrategias L y R o también la tercera C:

	L	R
U	1, 1	1, 3
D	0, 5	1, 13

	L	C	R
U	1, 1	0, 2	1, 3
D	0, 5	2, 8	1, 13

	L	C	R
U	1, 1	0, −100	1, 3
D	0, 5	2, −100	1, 13

Claramente, la columna recién agregada está dominada por las otras y no participará en ningún equilibrio de Nash (ni en ningún otro concepto de solución razonable). De hecho, existe un isomorfismo entre los equilibrios de Nash del juego original y el relleno. Por lo tanto, la incertidumbre sobre el espacio de la estrategia se reduce a la incertidumbre sobre los pagos. Usando tácticas similares, se puede demostrar que también es posible reducir la incertidumbre acerca de otros aspectos del juego a la incertidumbre acerca de los pagos solamente. Esto no es una afirmación matemática, ya que no hemos dado una caracterización matemática de todas las formas posibles de incertidumbre, pero es el caso que tales reducciones se han demostrado para todas las formas comunes de incertidumbre.

El segundo supuesto sobre los juegos bayesianos es el supuesto común anterior. Por lo tanto, un juego bayesiano define no solo las incertidumbres de los agentes sobre el juego que se está jugando, sino también sus creencias sobre las creencias de otros agentes sobre el juego y, de hecho, toda una jerarquía infinita de creencias anidadas (el llamado espacio de tipo epistémico). El supuesto común anterior es un supuesto sustantivo que limita el alcance de la aplicabilidad. Sin embargo, hacemos esta suposición ya que nos permite formular las ideas principales en los juegos bayesianos, y sin la suposición, el tema se vuelve mucho más complicado de lo que es apropiado para este texto. De

hecho, la mayoría (pero no todo) el trabajo en la teoría de juegos hace esta suposición.

Hay varias formas diferentes de presentar los juegos bayesianos; Ofreceremos tres definiciones de juegos bayesianos. Incluimos los tres ya que cada formulación es útil en diferentes entornos y ofrece diferentes intuiciones sobre la estructura subyacente de esta familia de juegos.

Primero, presentamos una definición que se basa en conjuntos de información. Bajo esta definición, un juego bayesiano consiste en un conjunto de juegos que se diferencian solo en sus pagos, una definición común previa sobre ellos y una estructura de partición sobre los juegos para cada agente. Un juego bayesiano es una tupla (N, G, P, I) donde:
N es un conjunto de agentes;
G es un conjunto de juegos de tal forma que para todo g, g'en G e i en N, el espacio de estrategia de i en g es idéntica a la de i en g'. Es decir, el espacio de estrategia en g es idéntico al espacio de estrategia en g′;
P es el conjunto de todas las distribuciones de probabilidad
sobre G
$I= (I_1, ..., I_N)$ es una tupla de particiones de G, una para cada agente.

Una segunda forma de capturar la idea anterior es suponer la hipótesis de un agente especial llamado Naturaleza que toma decisiones probabilísticas. Si bien podríamos intercalar arbitrariamente la elección de la Naturaleza con los movimientos de los agentes, sin perder la generalidad, suponemos que la Naturaleza toma todas sus decisiones desde el principio. La naturaleza no tiene una función de utilidad (o, alternativamente, se puede considerar que tiene una función constante) y tiene la estrategia única de aleatorizar de una manera comúnmente conocida. Los agentes reciben señales individuales sobre la elección de la Naturaleza, y estos son capturados por sus conjuntos de información de manera estándar. Los agentes no tienen información adicional; en particular, los conjuntos de información captan el hecho de que los agentes toman sus decisiones sin conocer las elecciones de los demás. Por lo tanto, hemos reducido los juegos de información incompleta a los juegos de información imperfecta, aunque con movimientos aleatorios. Estos movimientos aleatorios de la Naturaleza requieren ajustes menores de las definiciones existentes, reemplazando los beneficios por sus expectativas, dados los movimientos de la Naturaleza.

En el tercer tipo el supuesto es que todo lo anterior es de conocimiento común entre los jugadores, y que cada agente conoce su propio tipo. Esta definición puede parecer misteriosa, porque la noción de tipo puede ser bastante opaca. En general, el tipo de agente encapsula toda la información que posee el agente que no es de conocimiento común. A menudo, esto es bastante simple (por ejemplo, el

conocimiento del agente de su función de pago privado), pero también puede incluir sus creencias sobre los pagos de otros agentes, sobre sus creencias sobre su propia recompensa y cualquier otra creencia de orden superior.

2.2. JUEGOS BAYESIANOS-STACKBERGER

Los juegos de Stackelberg son juegos no simétricos donde un jugador o un grupo específico de jugadores tienen una posición privilegiada para la toma de decisiones antes que otros jugadores. Ellos juegan primero y el resto de jugadores siguen al líder/líderes y toman sus decisiones basadas en las acciones de éstos. Estos juegos pueden ser una buena propuesta para dar respuesta a los sistemas informáticos, donde la seguridad sigue siendo un desafío importante. Los juegos de Stackelberg están relacionados con aspectos de seguridad en los sistemas de telecomunicaciones y transporte los cuales se examinan y se analizan las propiedades desde la perspectiva de la implementación.

En particular, la principal contribución de nuestro trabajo es trazar el problema de la programación de seguridad como un juego Bayesian Stackelberg y resolverlo a través del algoritmo óptimo más rápido para tales juegos, abordando los dos primeros retos. El algoritmo utilizado se basa en varios años de investigación en sistemas multiagente y seguridad.

Los juegos de seguridad de Stackelberg se han implementado con éxito en sistemas de TI realistas a gran escala para respaldar la administración del sistema y las decisiones de los usuarios y administradores. La implementación más espectacular del modelo de juego de seguridad Stackelberg es el sistema de seguridad del Aeropuerto Internacional de Los Ángeles.

La asignación aleatoria de horarios en tales sistemas para monitorear el rendimiento del sistema es un problema crítico. La razón principal de esto es la importancia del conocimiento sobre las posibles patrullas que pueden causar ataques terroristas. Este caso de uso se realizó como un sistema multiagente de asistente de software llamado ARMOR (Asistente para monitoreo aleatorio sobre rutas). Este modelo es compatible con las decisiones de los administradores y usuarios sobre la ubicación de los puntos de control en el entorno físico o en las rutas de patrulla canina. El modelo de decisión se basa en los juegos bayesianos de Stackelberg, en los que se genera una estrategia mixta óptima para el líder (patrulla) y el seguidor (terrorista) el cual puede conocer esta estrategia mixta al elegir su propia estrategia en el juego.

El siguiente ejemplo del juego práctico Stackelberg es el sistema de asignación de seguridad estratégica en redes de transporte (IRIS) utilizado por Federal Air Marshal Service (FAMS). En las redes de transporte con cientos de miles de vehículos, la policía tiene que crear horarios de patrullaje para garantizar la seguridad. Los agresores pueden observar los patrones de aplicación de la ley y

tratar de explotar el cronograma generado. Los sistemas IRIS utilizan el solucionador más rápido conocido para esta clase de juegos de seguridad, a saber ERASER-C.

Otro caso de uso de Stackelberg es el sistema de Administración de Seguridad del Transporte de los Estados Unidos (TSA). Los sistemas de transporte son muy grandes y su protección requiere mucho personal y actividades de seguridad. El sistema apoyó las decisiones sobre cómo dividir adecuadamente los recursos entre capas de actividades de seguridad. En este tipo de juego, la TSA actúa como un defensor que tiene un conjunto de objetivos que proteger, una serie de actividades de seguridad y un número limitado de recursos. El nombre de sistema de software dedicado es Gametheoretic (GUARDS).

Hay muchas aplicaciones de la teoría de juegos a las comunicaciones y redes. Usando una variedad de herramientas de la teoría de juegos, fue posible encontrar nuevas soluciones en áreas relacionadas con redes celulares y de banda ancha como el control de potencia de enlace ascendente en redes CDMA, asignación de recursos en redes OFDMA, despliegue de puntos de acceso de femtocélulas, acceso inalámbrico de banda ancha IEEE 802.16 y traspaso vertical en redes inalámbricas heterogéneas.

Como hemos visto, los Juegos de Stackelberg son herramientas muy prometedoras para modelar los datos y la administración de usuarios, así como para soportar procesos de decisión complejos en entornos computacionales competitivos con posibles conflictos de intereses de los usuarios y administradores de sistemas y proveedores de servicios y recursos.

Todos los modelos encuestados se basaron en las características realistas de los sistemas, a saber, las limitaciones existentes en el acceso a los recursos, la incertidumbre sobre los tipos de seguidores, el comportamiento no óptimo de los jugadores o el conocimiento limitado de las acciones y estrategias de los oponentes. El aumento de la eficiencia del modelo de juego está estrictamente relacionado con el aumento del número calculado de parámetros en el juego y las ecuaciones para resolver en los modelos de optimización del juego, lo que hace que la implementación de todos estos modelos sea más compleja. Si bien todos los problemas de optimización relacionados con la resolución de los juegos de seguridad de Stackelberg presentados son NP-duros, los casos de uso práctico que se reportan en este documento muestran los altos beneficios prácticos potenciales de usar los juegos presentados en los sistemas de transporte en los Estados Unidos.

Todo esto convierte a estos modelos en una herramienta potencialmente eficiente para apoyar las decisiones complejas en entornos de nube a gran escala, que será el próximo paso de la investigación de los autores sobre aspectos de seguridad en la computación en la nube.

A continuación, mostraremos el modelo matemático a través del cual podemos resolver juegos Bayesianos de Stackelberg.

Juego de Stackelberg: Un agente (líder) debe acometer una estrategia que debe ser observada por otro agente (el seguidor)

Juego Bayesiano Stackelberg: es un juego de Stackelberg donde el líder tiene incertidumbre sobre los tipos de adversarios que puede encontrar.

Ejemplo: Juego de Stackelberg: problema de seguridad.

	c	d
a	2-1	4-0
b	1-0	3-2

1. Movimiento (a-c): el líder gana 2
2. Movimiento puro (b-d): el líder gana 3
3. Movimiento mixto: (0.5 a+ 0,5 b-d): el líder gana 4*0.5+3*0.5=3,5

El objetivo es encontrar la estrategia óptima que debe tomar el líder en este tipo de juegos.

El problema es que la solución al problema es muy costosa en general.

Soluciones existentes:

-Harsanyi: Referencia: J.C.Harsanyi y R.Selten. Una solución de Nash generalizada para juegos de negociación para dos personas con información incompleta. Management Science, 18 (5): 80-106

-MIP-Nash: Referencia: T. Sandholm, A. Gilpin, y V. Conizer. Métodos de programación de enteros mixtos para encontrar equilibrios nash. En AAAI, 2005.

-ASAP:Referencia: P. Paruchuri, J.P.Pearce, M.Tambe, G.Ordóñez, y S.Kraus. Un enfoque heurístico eficiente para la seguridad contra múltiples adversarios. En AAMAS, 2007.

En este trabajo usaremos la técnica DOBSS debido a que tiene ventajas sobre las otras, estas son:

1. Tenemos un juego bayesiano en forma compacta.
2. Sólo se requiere un programa lineal entero-mixto para resolverlo.
3. Búsqueda directa de la estrategia óptima del líder en vez de un equilibro de Nash.

Definición del problema:

1. Dos agentes, el líder y el seguidor.
2. Conjunto de posibles tipos para líderes, L
3. Conjunto de posibles tipos para seguidores, S
4. Conjunto de estrategias del líder, EL
5. Conjunto de estrategias del seguidor, ES
6. Función de utilidad del líder, $U: LxSxELxES \rightarrow R$
7. Objetivo: Encontrar la estrategia mixta óptima para que el líder se comprometa, dado que el seguidor puede conocer esta estrategia mixta al elegir su propia estrategia.

DBOSS: Originalmente el problema es un programa cuadrático entero-mixto.

La primera acción es descomponerlo en programas lineales y luego descomponer el lineal en otros más sencillos.

El seguidor busca la estrategia pura que le de una recompensa mayor.

El líder busca la estrategia mixta que le de el mayor rendimiento, dada la estrategia del seguidor. En el ejemplo vemos la razón de escoger una estrategia mixta.

PROGRAMA CUADRÁTICO ENTERO-MIXTO.

Nociones:

-

x_i: proporción de tiempo en el que el líder usa la estrategia pura i.

- X: conjunto de índices de las estrategias puras del líder.
- Q: conjunto de índices de las estrategias puras del seguidor.
- R_{ij}: Recompensa del líder cuando éste elige una estrategia i y el seguidor la estrategia j.

- C_{ij}: Recompensa del seguidor cuando éste elige una estrategia j y el líder la estrategia i.
-

PROGRAMA ÓPTIMO PARA EL SEGUIDOR

Prog

rama primario.

$$max_q \sum_{j\in Q} \sum_{i\in X} C_{ij}\, x_i q_j$$

s.t.

$$\sum\nolimits_{j\in Q} q_j = 1$$

$$q_j \geq 0$$

Progr

ama dual

$$min_a a$$

s.t.

$$a \geq \sum_{i\in X} C_{ij}\, x_i, \qquad j \in Q$$

Holgura complementaria

$$q_j \left(a - \sum_{i\in X} C_{ij}\, x_i \right) = 0, j \in Q$$

INCISO SOBRE PROGRAMACIÓN LINEAL

Cada problema de programación lineal, denominado problema primario, se puede convertir en un problema dual, que proporciona un límite superior al valor óptimo del problema primario.

Podemos expresar el problema primario (P) como:

$$max\ c^T x$$

s.t.

$$Ax \leq b, x \geq 0$$

El problema dual sería

$$min\ c^T y$$

s.t.

$$Ay \geq b, y \geq 0$$

Holgura: Si x e y son soluciones factibles del programa primario y dual respectivamente entonces:

$$\forall \text{ i}, (b_i - \sum_{j \in Q} a_{ij} x_j) y_i = 0$$

$$\forall \text{ j}, (b_i - \sum_{j \in Q} a_{ij} x_j) y_i = 0$$

PROGRAMA ÓPTIMO PARA EL LIDER

$$max_{x,q,a} \sum_{i \in X} \sum_{j \in Q} R_{ij}\, x_i q_j,$$

$$\sum\nolimits_{j \in Q} q_j = 1\,, \sum\nolimits_{i \in X} x_i = 1\,, q_j \geq 0, x_i \geq 0$$

$$0 \leq \left(a - \sum_{i \in X} C_{ij}\, x_i \right) \leq \left(1 - q_j\right) M$$

$$x_j \in [0,1], q_j \in \{0,1\}$$

(1) (3): Aplicar una política mixta viable para el líder

(2) (4): Hacer cumplir una estrategia pura viable para el seguidor

(3): Desigualdad de la izquierda: Aplica la doble viabilidad del problema del seguidor

Desigualdad en el extremo derecho: restricción de holgura complementaria para una estrategia pura óptima q para el seguidor.

DESCOMPOSICIÓN DEL PROGRAMA CUADRÁTICO DEL LIDER EN PROGRAMAS DE PROGRAMACIÓN ENTERA MIXTA LINEALES.

NOTACIONES:

-p^l: probabilidad a priori de que un seguidor tipo l pueda aparecer.

-L: conjunto de los tipos de seguidor

-X: conjunto de índices de las estrategias puras del líder.

-Q: conjunto de índices de las estrategias puras del seguidor.

-R^l: matriz de pago del líder. $R_{ij} = \sum_{l \in L} R_{ij}^l$

-C^l: matriz de pago del líder. $C_{ij} = \sum_{l \in L} C_{ij}^l$

PROBLEMA A RESOLVER:

$$max_{x,q,a} \sum_{i \in X} \sum_{l \in L} \sum_{j \in Q} p^l {R_{ij}}^l \, x_i {q_j}^l,$$

$$\sum\nolimits_{j \in Q} {q_j}^l = 1 \, , \sum\nolimits_{i \in X} x_i = 1 \, , {q_j}^l \geq 0, x_i \geq 0$$

$$0 \leq \left(a^l - \sum_{i \in X} {C_{ij}}^l \, x_i \right) \leq \left(1 - {q_j}^l\right) M$$

$$x_j \in [0,1], {q_j}^l \in \{0,1\}, a^l \in R.$$

LINEALIZACIÓN PARA CADA TIPO DE SEGUIDOR

La desigualdad de la restricción 2 más a la izquierda asegura para todo $j \in Q$, dado el vector x del líder, a^l es un límite superior en el seguidor.

La desigualdad más a la derecha es inactiva para cada acción donde $q^l = 0$, ya que M es grande. Para la acción con $q^l = 1$, esta desigualdad indica que la recompensa del seguidor por esta acción debe ser menor o igual que a^l, que combinada con la desigualdad anterior muestra que esta acción

debe ser óptima para el seguidor de tipo l.

Observe que el problema 1 es un MIQP descompuesto en el sentido que no utiliza una transformación completa de Harsanyi;

En su lugar, resuelve múltiples problemas más pequeños utilizando adversarios personales (indexados por l) en lugar de uno solo.

Además, esta descomposición no causa ninguna sub-optimalidad (Paruchuri et Alabama. 2008).

Ahora podemos linealizar.

$$max_{x,q,a} \sum_{i \in X} \sum_{l \in L} \sum_{j \in Q} p^l {R_{ij}}^l z_{ij}^l$$

$$\sum\nolimits_{j \in Q} \sum\nolimits_{x \in X} {z_{ij}}^l = 1 \,, \sum\nolimits_{i \in X} x_i = 1 \,, {q_j}^l \geq 0, x_i \geq 0$$

$$0 \leq \left(a^l - \sum_{i \in X} {C_{ij}}^l \left(\sum\nolimits_{h \in Q} {z_{ih}}^l \right) \right) \leq \left(1 - {q_j}^l\right) M$$

$$z_{ij} \in [0,1], {q_j}^l \in \{0,1\}, a^l \in R.$$

2.3 ARMOR (ASISTENTE PARA EL MONITEOREO DE RUTAS)

En particular, ARMOR se basa en un algoritmo óptimo llamado DOBSS (solucionador de Bayesiano Stackelberg óptimo descompuesto) mientras que un juego bayesiano nos permite abordar la incertidumbre sobre los tipos de adversarios. Al resolver de manera óptima dichos juegos bayesianos de Stackelberg (que brindan estrategias aleatorias óptimas como soluciones), ARMOR ofrece garantías de calidad en los cronogramas generados. Estas garantías de calidad obviamente no implican que ARMOR proporcione seguridad perfecta; en su lugar, garantiza la optimización en la utilización de recursos de seguridad fijos (cantidad de unidades de policía o caninas), siempre que las recompensas se modelen con precisión. En otras palabras, dado un número específico de recursos de seguridad y áreas a proteger, ARMOR crea un programa que asigna al azar el posible despliegue de esos recursos de una manera que optimiza la recompensa esperada.

El tercer desafío se aborda mediante el uso de ARMOR de una interfaz basada en iniciativas mixtas, donde los usuarios pueden ingresar gráficamente diferentes restricciones para dar forma al cronograma generado. ARMOR es, por lo tanto, un asistente de colaboración que itera sobre los horarios generados en lugar de un programador de un solo disparo rígido. También alerta a los usuarios en caso de que las anulaciones puedan deteriorar la calidad de la programación. Este asistente, por lo tanto, representa una transición muy prometedora de la investigación multiagente en una aplicación implementada.

ARMOR se ha desplegado con éxito desde agosto de 2007 en el Aeropuerto Internacional de Los Ángeles (LAX) para ayudar a la policía del Aeropuerto Mundial de Los Ángeles (LAWA) en la programación aleatoria de los puntos de control y desde noviembre de 2007 para generar horarios de patrullaje aleatorios para las unidades caninas. En particular, ayuda a la policía a determinar dónde establecer aleatoriamente los puntos de control y dónde asignar caninos al azar a los terminales. De hecho, febrero de 2008 marcó el final exitoso del período de prueba de seis meses de implementación de ARMOR en LAX.

La respuesta de la policía al final de este período de seis meses fue extremadamente positiva; ARMOR continuaría implementándose en LAX y se expandiría a otras actividades policiales en LAX.

Descripción del dominio de seguridad en LAX

Ahora describiremos los desafíos específicos en los problemas de seguridad que enfrenta la policía de LAWA. LAX 1 es el quinto aeropuerto más ocupado de los Estados Unidos y el aeropuerto de destino más grande de los Estados Unidos, con 60 a 70 millones de pasajeros por año. Desafortunadamente, también se sospecha que LAX es un objetivo terrorista principal en la costa oeste de los Estados Unidos, con múltiples arrestos de conspiradores que intentan atacar a LAX. Para proteger a LAX, la policía de LAWA ha diseñado un sistema de seguridad que utiliza múltiples anillos de protección. Como es evidente para cualquiera que viaje a través de un aeropuerto, estos anillos incluyen cosas tales como puestos de control de vehículos, unidades de la policía que patrullan las carreteras hacia las terminales y dentro de las terminales (con perros), y controles de seguridad y controles de equipaje para los pasajeros. Desafortunadamente, no hay suficientes recursos (oficiales de policía) para monitorear cada evento en el aeropuerto; dado su tamaño y la cantidad de pasajeros atendidos, tal nivel de revisión requeriría considerablemente más personal y causaría mayores retrasos a los viajeros. Por lo tanto, suponiendo que todos los puntos de control y terminales no estén siendo monitoreados en todo momento, la configuración de los puestos de control disponibles, unidades caninas u otras patrullas en horarios deterministas permite a los adversarios aprender los horarios y trazar un ataque que evite los puestos de control y patrullas de la policía, lo que hace que los horarios deterministas sean ineficaces.

La aleatorización ofrece una solución aquí. En particular, entre todas las medidas de seguridad a las que se podría aplicar la asignación al azar, la policía de LAWA ha planteado hasta ahora dos problemas cruciales para nosotros. Primero, dado que hay muchos caminos que conducen a LAX, quieren saber dónde y cuándo deben establecer puntos de control para controlar los automóviles que ingresan en LAX. Por ejemplo, la fi gura 1 muestra un punto de control vehicular configurado en una carretera que se dirige hacia LAX. Los oficiales de policía examinan los automóviles que pasan, y si algún automóvil parece sospechoso, hacen una inspección más detallada de ese automóvil. La policía de LAWA deseaba obtener un horario aleatorio para tales puntos de control durante un período de tiempo particular. Por ejemplo, si vamos a establecer dos puntos de control y el período de interés es de 8 am a 11 am, entonces un calendario de candidatos puede sugerir a la policía que el lunes, los

puntos de control se coloquen en la ruta 1 y la ruta 2, mientras que en Los martes durante el mismo intervalo de tiempo, deben estar en las rutas 1 y 3, y así sucesivamente. Segundo, la policía de LAWA deseaba obtener una asignación de caninos para patrullar rutas a través del Terminales dentro de LAX. Por ejemplo, si hay tres unidades caninas disponibles, una posible asignación puede ser colocar los caninos en las terminales 1, 3 y 6 el primer día, pero en las terminales 2, 4 y 6 en otro día, y así sucesivamente, en función de La información disponible.

Dados estos problemas, nuestro análisis reveló tres desafíos clave: primero, los posibles atacantes pueden observar los programas de las fuerzas de seguridad a lo largo del tiempo y luego elegir su estrategia de ataque; este hecho hace que los programas deterministas sean altamente susceptibles de ser atacados; segundo, hay información desconocida e incierta sobre los tipos de adversarios que podemos enfrentar; y tercero, aunque la aleatorización ayuda a eliminar los patrones deterministas, también debe tener en cuenta los diferentes costos y beneficios asociados con objetivos particulares.

Al resumir los requisitos del dominio, enfatizamos los siguientes puntos clave. Primero, es la policía de LAWA, como expertos en dominios, quien expresó un requisito para la aleatorización, lo que nos lleva a diseñar ARMOR. En segundo lugar, existen diferentes anillos de seguridad (incluidos los caninos y los puntos de control que ARMOR programa), que no son estáticos y, por lo tanto, pueden cambiar independientemente de los otros anillos en diferentes momentos. El resultado final de tales cambios aleatorios de seguridad aleatorios es que los costos del adversario y la incertidumbre aumentan, particularmente para ataques bien planificados, que a su vez pueden ayudar a disuadir y prevenir ataques.

Enfoque

Modelamos las decisiones de establecer puntos de control o rutas de patrullaje canino en el aeropuerto de LAX como juegos Bayesian Stackelberg. Estos juegos nos permiten realizar tres tareas importantes: modelan el hecho de que un adversario actúa con conocimiento de los cronogramas de las fuerzas de seguridad y, por lo tanto, aleatorizan los cronogramas de manera apropiada; nos permiten definir múltiples tipos de adversarios, enfrentando el desafío de nuestra información incierta sobre nuestros adversarios; y nos permiten sopesar la importancia de diferentes objetivos de manera diferente. Dado que los juegos Bayesian Stackelberg abordan los desafíos planteados por nuestro dominio, están en el centro de la generación de horarios aleatorios significativos. A partir de este punto, explicaremos en qué consiste un juego Bayesian Stackelberg, cómo se puede asignar un

problema de seguridad LAX a los juegos Bayesian Stackelberg, algunos de los métodos anteriores para resolver juegos Bayesian Stackelberg y cómo utilizamos el DOBSS para resolver de manera óptima el problema en cuestión.

En un juego de Stackelberg, un líder se compromete primero con una estrategia, y luego un seguidor optimiza egoístamente su recompensa, considerando la acción elegida por el líder. Por ejemplo, dado nuestro dominio de seguridad, la fuerza policial (líder) debe primero comprometerse con una estrategia mixta para colocar puntos de control en las carreteras para que sean impredecibles para los adversarios (seguidores), donde una estrategia mixta implica una distribución de probabilidad sobre las acciones de establecer puntos de control. Los adversarios, después de observar los puntos de control a lo largo del tiempo, pueden elegir su propia estrategia para atacar una carretera específica.

Técnicas para resolver Juegos de Stackelberg

Los investigadores en el pasado han identificado un enfoque, al que nos referiremos como el método de múltiples LP, para resolver los juegos de Stackelberg, y esto se puede usar para resolver los juegos Bayesianos de Stackelberg. Este enfoque, sin embargo, requiere transformar un juego bayesiano en un juego de forma normal utilizando la transformación Harsanyi. De manera similar, se pueden aplicar algoritmos eficientes para encontrar equilibrios de Nash, pero requieren la misma transformación de Harsanyi, de la cual, es importante comprender su transformación e impacto.

Transformación de Harsanyi

El primer paso para resolver los juegos bayesianos de métodos anteriores es aplicar la transformación Harsanyi que convierte el juego de información incompleta en un juego de forma normal. Dado que la transformación de Harsanyi es un concepto estándar en la teoría de juegos, lo explicamos brevemente a través de un simple ejemplo sin introducir el cálculo matemático. Considere el caso de los dos tipos de seguidor 1 y 2 como se muestra en las tablas de abajo.

	Seguidor 1	
	c	**d**
a	2,1	4,0
b	1,0	3,2

	Seguidor 2	
	c'	d'
a	1,1	2,0
b	1,0	3,2

El seguidor de tipo 1 estará activo con probabilidad α, y el seguidor de tipo 2 estará activo con probabilidad 1 - α. Realizar la transformación de Harsanyi implica introducir un nodo de azar que determine el tipo de seguidor, transformando así la información incompleta del líder con respecto al seguidor en un juego de información imperfecto. El juego de forma normal transformada se muestra en la tabla *Transformada de Harsanyi*.

	Transformada de Harsanyi			
	cc'	cd'	dc'	dd'
a	2α + (1- α), 1	2,α	4α + (1-α), (1-α)	4α +2(1-α), 0
b	α, (1- α)	α + 3(1-α), 2(1- α)	3α, 2α + (1-α)	3,2

En el juego transformado, el líder aún tiene dos estrategias, mientras que hay un solo tipo de seguidor con cuatro (2 * 2) estrategias. Por ejemplo, considere la situación en el juego transformado donde el líder toma acción ay el seguidor toma acción cc'. La recompensa del líder en el nuevo juego se calcula como una suma ponderada de sus recompensas de las dos tablas de los seguidores, es decir, α veces la recompensa del líder cuando el seguidor tipo 1 realiza la acción c más 1-α veces la recompensa del líder cuando el seguidor escribe 2 toma acción c'.

Todas las demás entradas en la nueva tabla, tanto para el líder como para el seguidor, se derivan de una manera similar. En general, para n tipos de seguidores con k estrategias por tipo de seguidor, la transformación da como resultado un juego con estrategias de k^n para el seguidor, lo que provoca una explosión exponencial que pierde la compacidad. Métodos como los descritos en Conitzer y Sandholm (2006) y Sandholm, Gilpin y Conitzer (2005) deben usar esta transformación de Harsanyi, lo que implica que el juego pierde su estructura compacta. No obstante, las soluciones que obtienen sus métodos pueden volver a formarse en el juego

original.

DOBSS

Una ventaja clave del enfoque DOBSS es que opera directamente en la representación bayesiana, sin requerir la transformación de Harsanyi. En particular, DOBSS obtiene un esquema de descomposición al explotar la propiedad de que los tipos de seguidores son independientes entre sí. La clave para la descomposición de DOBSS es la observación de que evaluar la estrategia del líder contra una matriz de juego transformada por Harsanyi es equivalente a evaluar contra cada una de las matrices de juego para los tipos de seguidores individuales.

Primero presentamos DOBSS en su forma más intuitiva como un programa cuadrático de enteros mixtos (MIQP); Luego, ilustramos cómo se puede transformar en un programa lineal de enteros mixtos equivalentes linealizados (MILP). El modelo que proponemos representa explícitamente las acciones del líder y las acciones óptimas para los tipos de seguidores en el problema resuelto por el agente.

2.3.1. IMPARCTO DE ARMOR EN LAX

Diseñar e implementar el software ARMOR a modo de prueba en LAX planteó numerosos desafíos y problemas para nuestro grupo de investigación. Algunas de las lecciones clave aprendidas durante el diseño y la implementación de ARMOR incluyen la importancia de las herramientas para la aleatorización, la importancia de las anulaciones de los horarios manuales y la importancia de brindarles a los oficiales de policía flexibilidad operativa.

Importancia Herramientas de Aleatorización

Existe una necesidad crítica de aleatorización en las operaciones de seguridad. Los funcionarios de seguridad son conscientes de que requerir que los humanos generen horarios aleatorios no es satisfactorio porque, como los estudios psicológicos han demostrado, los humanos tienen dificultades para la aleatorización. En cambio, la aleatorización matemática que pesa adecuadamente los costos y beneficios de las diferentes acciones y la aleatorización en consecuencia lleva a mejores resultados. Por lo tanto, los funcionarios de seguridad se mostraron extremadamente entusiastas al recibir nuestra investigación y ansiosos por aplicarla a su dominio. Además, estos funcionarios han indicado que la obtención de horarios reduce automáticamente la carga de tener que construir dichos horarios manualmente teniendo en cuenta todos los factores relevantes.

Importancia de anulaciones de horarios manuales

Si bien ARMOR incorpora todo el conocimiento que podríamos obtener de la policía de LAWA y proporciona el mejor resultado posible, es posible que no esté al tanto de los desarrollos dinámicos en el terreno. Por ejemplo, los oficiales de policía pueden tener inteligencia muy específica para requerir un punto de control en una carretera de entrada particular. Por lo tanto, fue crucial permitir que los oficiales de policía de LAWA (en raras ocasiones cuando sea necesario) anulen manualmente de forma selectiva el programa proporcionado.

Importancia de la flexibilidad operativa a la Policía

Al generar inicialmente horarios para patrullas caninas, el sistema creó un cronograma muy detallado, que microgestiona a las patrullas. Esto no obtuvo una recepción tan positiva por parte de los oficiales. En lugar de eso, se recibió mejor un calendario abstracto que brindaba a los oficiales cierta flexibilidad para responder a situaciones dinámicas en el terreno.

Los resultados experimentales exploran la eficiencia del tiempo de ejecución de DOBBS y evalúan la calidad de la solución y la implementación del sistema ARMOR.

Análisis en tiempo de ejecución

DOBSS supera significativamente a sus competidores, que incluyen MIP Nash y varios L en un dominio experimental que involucra a un agente de seguridad que patrulla un mundo compuesto por m casas , 1… my un ladrón tratando de robar estas casas. Estos resultados muestran que incluso para un mundo tan pequeño como tres casas, MIP-Nash y múltiples LP no pueden converger en una solución dentro del tiempo permitido de 30 minutos cuando hay 8 o más tipos de adversarios. DOBSS, sin embargo, puede lograr una solución en menos de 10 segundos para hasta 14 tipos de adversarios. Además, a medida que el número de casas aumenta hasta cinco, MIP-Nash y varios LP no pueden converger en una solución dentro de los 30 minutos, incluso para un número bajo de tipos de adversarios.

El análisis de tiempo de ejecución se suma a los resultados anteriores, centrándose específicamente en el dominio de seguridad actual. Por esta razón, se comparan los resultados en tiempo de ejecución de DOBSS frente a varios LP, descritos anteriormente, dado el dominio específico utilizado para la implementación de caninos en LAX. MIP-Nash no se ha incluido en este análisis de tiempos de ejecución, ya que solo proporciona el mejor equilibrio de Bayes Nash en comparación con las estrategias mixtas óptimas proporcionadas por el método de múltiples LP y el método DOBSS. El objetivo de este análisis es mostrar que DOBSS es, de hecho, el procedimiento más adecuado para la aplicación en dominios reales como el canino LAX y la asignación de puntos de control. Con ese fin, se utilizan los datos de una semana completa de despliegue canino para analizar el tiempo necesario para generar un programa dado el método DOBSS y el método de múltiples LP. Para completar, se muestran los resultados dados de uno a cuatro tipos de adversarios donde cuatro tipos de adversarios es la cantidad mínima que LAWA ha establecido según sea necesario.

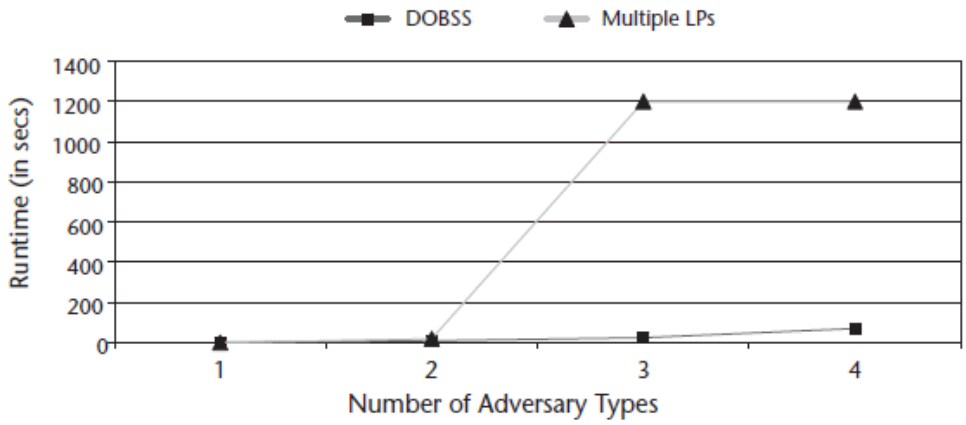


Figura 1

En la figura 1 se resumen los resultados del tiempo de ejecución de nuestros juegos bayesianos utilizando DOBSS y varios LP. Se prueban los resultados en los juegos bayesianos provistos desde el dominio canino con varios tipos de adversarios que varían entre uno y cuatro. Cada juego entre LAWA y un tipo de adversario se modela como un juego de forma normal. Por lo tanto, hay cuatro juegos de forma normal diseñados para el juego entre LAWA y los distintos tipos de adversarios para el caso base. El tamaño de cada uno de estos juegos de forma normal es (784, 8), que corresponde a 784 estrategias para LAWA y 8 para el adversario. Luego se utilizan las siete instancias generadas, tomadas de una semana arbitraria de despliegue canino, de este caso base para obtener resultados promediados.

El eje x en muestra el número de tipos de seguidores a los que se enfrenta el líder y el eje y del gráfico muestra el tiempo de ejecución en segundos. Todos los experimentos que no se concluyeron en 20 minutos (1.200 segundos) se cortaron. A partir de la gráfica, se resume que DOBSS supera el método de múltiples LPs por un margen significativo dado el dominio dominio canino real. En el gráfico, si bien varios LP podrían resolver el problema solo para hasta dos tipos de adversarios, DOBSS podría resolver los cuatro tipos de adversarios en 80 segundos.

Por lo tanto, se comprueba que el método DOBSS es más rápido que el método de múltiples LPs. En consecuencia, DOBSS es el algoritmo de elección para los juegos Bayesian Stackelberg (Paruchuri et al. 2008), especialmente teniendo en cuenta los juegos particulares creados por dominios de seguridad reales como el problema de patrullaje canino.

Evaluación de ARMOR

Ahora evaluamos la calidad de la solución obtenida cuando se aplica DOBSS al dominio de

seguridad LAX. Ofrecemos tres tipos de evaluación. Mientras nuestra primera evaluación está en el laboratorio, ARMOR es un asistente desplegado, y por lo tanto, nuestras dos evaluaciones restantes son de su implementación en el campo. Con respecto a nuestra primera evaluación, realizamos cuatro experimentos. Los tres primeros compararon la aleatorización de ARMOR con una técnica de aleatorización uniforme que no usa los pesos de ARMOR en la aleatorización.

Los resultados del primer experimento se muestran en las figuras 2a, 2b y 2c. El eje x representa las probabilidades de ocurrencia de los dos tipos de adversarios que elegimos para enfocar. Dado que la cantidad real de tipos de adversarios utilizados para LAX es información segura, usamos dos tipos de adversarios para simplificar este análisis. El eje x muestra la probabilidad p del adversario tipo 2 (la probabilidad del adversario tipo 1 se obtiene en 1 - p). El eje y representa la recompensa obtenida por LAWA. Esta recompensa representa la recompensa esperada que LAWA obtendría dada la respuesta óptima del adversario a la estrategia adoptada por LAWA. La Figura 2a muestra la comparación cuando se coloca un punto de control. Por ejemplo, cuando ocurre un adversario de tipo 1 con una probabilidad de 0.1 y un tipo 2 con una probabilidad de 0.9, la recompensa obtenida por la estrategia DOBSS es de -1,72 mientras que la recompensa obtenida por una estrategia aleatoria uniforme es de -2,112. Es importante tener en cuenta que la recompensa de la estrategia DOBSS es estrictamente mayor que la recompensa de la estrategia aleatoria uniforme para todas las probabilidades de ocurrencia de los tipos adversarios. La Figura 2b también tiene la distribución de probabilidad en el eje x y la recompensa obtenida en el eje y. Muestra la diferencia en la recompensa obtenida cuando se colocan dos puntos de control. Aquí también la recompensa en el caso de la estrategia DOBSS es mayor que la recompensa de la estrategia aleatoria uniforme. Cuando tenemos dos puntos de control, el adversario de tipo 2 elige la acción ninguno (para no atacar). Esto lleva a la observación de que la recompensa de la estrategia DOBSS y la recompensa de la estrategia uniforme son las mismas cuando solo está presente el adversario de tipo 2. La figura 2c presenta el caso de tres puntos de control. Aquí los valores de recompensa obtenidos por DOBSS siempre son positivos, esto se debe a que las posibilidades de atrapar al adversario de tipo 1 mejoran significativamente con tres puntos de control. Esto también lleva a que la recompensa de DOBSS disminuya con la disminución de la probabilidad de ocurrencia del adversario de tipo 1. Hay que tener en cuenta que el adversario de tipo 2, como en el caso de dos puntos de control, decide ninguno y, por lo tanto, la recompensa de la estrategia DOBSS y la estrategia aleatoria uniforme son las mismas cuando solo está presente el adversario de tipo 2.

.

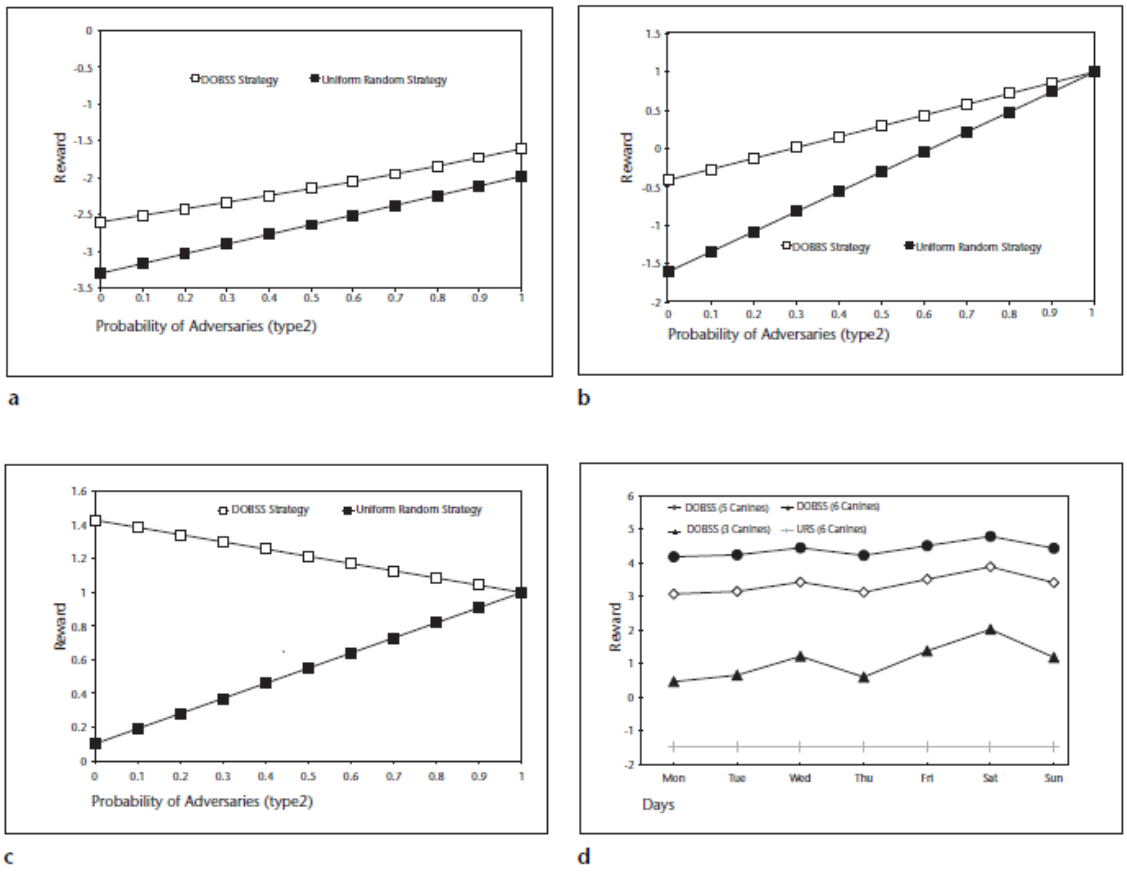


Figura 2

Los tres experimentos informados nos permiten concluir que la aleatorización ponderada DOBSS proporciona mejoras significativas sobre la aleatorización uniforme en el mismo dominio, lo que ilustra la utilidad de nuestros algoritmos. Continuamos estos resultados en el cuarto experimento siguiente, enfocándonos ahora en las unidades caninas. La Figura 2d muestra la comparación de la recompensa obtenida entre la programación de unidades caninas con DOBSS y su programación con una estrategia de administración uniforme (denominada URS). En la estrategia aleatoria uniforme, los caninos se asignan aleatoriamente a los terminales con igual probabilidad. El eje x representa el día de la semana y el eje y representa la recompensa obtenida. Podemos ver que DOBSS funciona mejor incluso con tres unidades caninas en comparación con las seis unidades caninas programadas utilizando la estrategia aleatoria uniforme. Por ejemplo, el viernes, la recompensa de una estrategia aleatoria uniforme con seis unidades caninas es −1.47, mientras que la recompensa de tres, cinco y seis unidades caninas con DOBSS es 1.37, 3.50 y 4.50, respectivamente. Estos resultados muestran que la aleatorización ponderada DOBSS con incluso tres unidades caninas proporciona mejores resultados contra la aleatorización uniforme en el mismo dominio con seis unidades caninas. Por lo tanto, nuestro algoritmo proporciona mejores recompensas y puede ayudar a reducir el costo de los

recursos necesarios.

En la siguiente evaluación, examinamos la configuración de ARMOR de los puntos de control en LAX. El primer experimento examina el cambio en la implementación del punto de control durante un turno fijo (es decir, manteniendo el tiempo fijo) durante dos semanas. Los resultados se muestran en la tabla *Variación en el Porcentaje de Uso*. Los números del 1 al 5 en la tabla indican el número del punto de control (hemos asignado números de identificación arbitrarios a todos los puntos de control para este experimento) y los valores de la tabla muestran el porcentaje de veces que este punto de control se utilizó. Por ejemplo, en la semana 1, el punto de control 2 se usó solo menos del 5 por ciento de las veces, mientras que el punto de control 2 se usó aproximadamente el 25 por ciento de las veces en la semana 2. Podemos hacer dos observaciones a partir de estas dos semanas: Primero, no aparecemos tener un uso uniforme de estos puntos de control; es decir, hay una gran variación en el porcentaje de veces que se despliegan los puntos de verificación. Segundo, el despliegue del punto de control varía de una semana a otra; por ejemplo, el punto de control 4 no se usó en la semana 1, pero se usó el 15 por ciento de las veces en la semana 2.

Checkpoint Number	1	2	3	4	5
Week 1	33.33	4.76	33.33	0	28.57
Week 2	19.04	23.80	23.80	14.28	19.05

Tabla *Variación en el Porcentaje de Uso*

El objetivo del siguiente experimento fue proporcionar resultados en el análisis de sensibilidad, específicamente, cómo cambiarán las probabilidades de diferentes acciones si cambiamos la proporción de tipos de adversarios. La Figura 3 muestra la variación en la estrategia para colocar dos puntos de control juntos cuando cambia la probabilidad de ocurrencia del adversario. El eje x muestra la variación en la probabilidad de ocurrencia de los tipos adversarios, mientras que el eje y muestra la variación en las probabilidades en la estrategia DOBSS. Por ejemplo, cuando ocurre un adversario de tipo 1 con una probabilidad de 1, la probabilidad de colocar ambos puntos de control 1 y 4 es 0.353; cuando los adversarios 1 y 2 ocurren con las probabilidades 0.4 y 0.6, respectivamente, entonces la probabilidad de colocar los puntos de control 3 y 4 es 0.127. Podemos observar que no hay variación en las probabilidades en las estrategias DOBSS cuando las probabilidades de ocurrencia de los dos tipos de adversarios varían de 0.1 a 0.9. Esto indica que nuestros resultados no son particularmente sensibles a las variaciones en las probabilidades de los oponentes, excepto en los extremos.

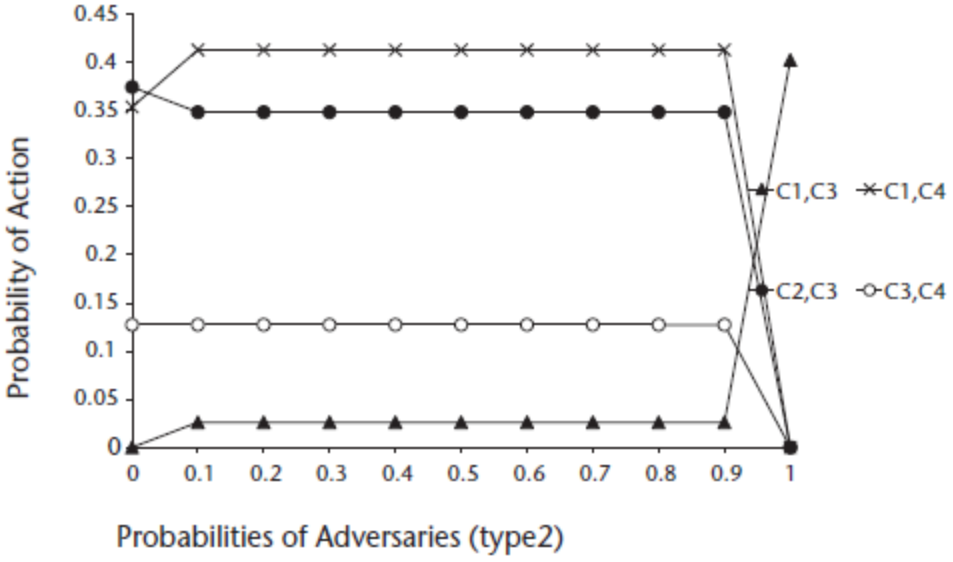


Figura 3

Nuestra evaluación final es más informal basada en los comentarios de la policía de LAWA. En primer lugar, han proporcionado comentarios muy positivos sobre el despliegue. Sugieren que la técnica que habían utilizado anteriormente no era la aleatorización, sino la de alternar los puntos de control; Tal rutina puede provocar determinismo en la programación, que hemos evitado. En segundo lugar, ARMOR ha reducido el trabajo de rutina en la programación, lo que permite a la policía de LAWA concentrarse en tareas más importantes. En tercer lugar, se han realizado varios arrestos en los puntos de control programados por ARMOR. Normalmente, estos coches implicados intentaban llevar armas a LAX. Finalmente, el director James Butts de la policía de LAX comentó sobre esto que la nueva colocación aleatoria hacía que los viajeros se sintieran más seguros e incluso les daba un mayor sentimiento de presencia policial al hacer que la policía parezca más numerosa. La mejoría evidente de la seguridad en el aeropuerto no se debió única y exclusivamente al software ARMOR pero las autoridades coincidían en que ARMOR supone la primera línea de defensa en el perímetro exterior del aeropuerto está ayudando a aliviar la amenaza de violencia en el aeropuerto.

Resumen

El problema del patrullaje en sí ha recibido una atención significativa en la literatura de múltiples agentes debido a su amplia variedad de aplicaciones que van desde la patrulla de robots hasta el patrullaje de fronteras de grandes áreas. La idea clave detrás de las políticas proporcionadas por estas técnicas es la aleatorización, que disminuye la cantidad de información que se le da a un adversario. Sin embargo, no se ha proporcionado ningún algoritmo / procedimiento específico para la generación de políticas aleatorizadas.

Si bien ARMOR es un programador de la seguridad de la teoría de juegos, hay muchas de sus

herramientas competidoras de la no teoría de la competencia en nosotros para aplicaciones relacionadas. Por ejemplo, el "Modelo de colocación en colas de Hypercube" (Larson, 1974), basado en la teoría de colas, describe la operación espacial detallada de los departamentos de policía urbanos y los servicios médicos de emergencia y ha encontrado aplicación en el diseño de batidos de la policía, la asignación del tiempo de patrullaje, etc. Sin embargo, este modelo no tiene en cuenta modelos adversos específicos; ARMOR, por otro lado, diseña políticas para combatir a varios adversarios potenciales. Previamente, se presentaron dos enfoques diferentes para encontrar soluciones para los juegos Bayesian Stackelberg de manera eficiente. Uno de los enfoques, denominado ASAP, es capaz de operar en la forma bayesiana de los juegos de Stackelberg, pero proporciona una solución aproximada. El segundo enfoque, el método de múltiples LPs, requiere que un juego bayesiano se transforme en un juego de forma normal utilizando la transformación Harsanyi. DOBSS es superior a ASAP, ya que proporciona soluciones exactas y, como se muestra, también supera el método de múltiples LP para nuestro dominio de interés. En resumen, establecer la seguridad en torno a la infraestructura de importancia económica o política es un desafío que enfrentan hoy las fuerzas policiales de todo el mundo. Si bien el monitoreo aleatorio es importante (ya que los adversarios pueden observar y explotar cualquier previsibilidad), la aleatorización debe usar diferentes funciones de pesaje para reflejar los complejos costos y beneficios de las diferentes acciones policiales. Este artículo describe un asistente de agente desplegado llamado ARMOR que resuelve el problema de monitoreo como un juego Bayesiano Stackelberg, donde la generación aleatoria de horarios puede sopesar los costos y beneficios, así como la incertidumbre sobre los tipos adversarios. ARMOR combina dos características clave: primero, usa el solucionador más rápido conocido para los juegos Bayesianos de Stackelberg llamado DOBSS, donde las estrategias mixtas dominantes proporcionan una aleatorización programada; y segundo, su interfaz basada en la iniciativa mixta permite a los usuarios ajustar o anular ocasionalmente el programa automatizado en función de sus limitaciones locales. ARMOR se ha desplegado con éxito en el Aeropuerto Internacional de Los Ángeles, asignó al azar la asignación de puntos de control desde agosto de 2007 y el despliegue de caninos desde noviembre de 2007. ARMOR representa una transición exitosa de avances algorítmicos de múltiples agentes para el pasado Dos años en el mundo real.

3. INTRODUCCIÓN AL PROYECTO DE APLICACIÓN

Como hemos comentado, la seguridad en los aeropuertos es uno de los temas que más preocupan a las autoridades hoy en día. El volumen de usuarios que transitan en los mismos, el impacto que pueda tener cualquier accidente en una aeronave o la repercusión a nivel mundial que pueda tener el hecho que un país tenga deficiencias en términos de seguridad ante cualquier incidente que se produzca, hacen de los aeropuertos un blanco perfecto para cualquier tipo de ataque.

El aeropuerto de Sevilla ha vivido en 2018 el mejor año de su historia con 6,38 millones de pasajeros. Este balance supone una media diaria de casi 17.500 viajeros diarios en la terminal, convirtiendo a San Pablo en el aeródromo de tamaño mediano que más crece de España y ha logrado superar a Tenerife Norte y Bilbao.

Esta tendencia, que comenzó en 2016, se mantendrá, al menos, en 2019 cuando las compañías aéreas tienen programadas nuevas rutas y más frecuencias de los vuelos con mayor demanda. La recuperación del tráfico nacional ha sido fundamental para conseguir estos objetivos, aunque el foráneo ha logrado superarlo. Cabe destacar la gran variedad de aerolíneas que operan en el aeródromo sevillano, que a día de hoy son casi una treintena. En relación con el tráfico extranjero, Francia, Italia y Reino Unido fueron, son los destinos con mayor demanda, seguidos muy de cerca por Alemania. En el caso de Dinamarca y Polonia, la cifra de pasajeros se multiplicó por seis y por cuatro, respectivamente, aunque debido, en gran parte, a que las conexiones con esos países no estuvieron operativas durante todo 2017 –se estrenaron avanzado ya el ejercicio-. Portugal es otro de los destinos que más crece, un 41,1 por ciento con respecto al año anterior y en este punto hay que explicar que no es sólo por el tránsito de viajeros entre ambos destinos, sino por ser uno de los puntos de escala para desplazamientos de larga distancia a Estados Unidos y Brasil.

El buen comportamiento de las estadísticas durante el año pasado se sustentó en la diversificación de rutas y mercados –sólo en la temporada de invierno se han programado 16 destinos nuevos-, además de los altos índices de ocupación de las aeronaves.

La colaboración con las instituciones locales encargadas de la promoción turística y de negocios de Sevilla ha sido fundamental en la consecución de los registros de 2018, así como las distintas líneas de trabajo que viene impulsando Aena para contribuir a la captación de conexiones. A lo largo de este ejercicio se seguirá trabajando para conseguir la primera ruta de larga distancia con Norteamérica o Asia.

El continuo crecimiento de visitantes año tras año justifica la transformación a la que el Aeropuerto se va a someter en los próximos dos años, lo que permitirá ganar capacidad para acoger a un volumen mayor de pasajeros y de operadores, y una modernización de las instalaciones.

Tanto el centro de coordinación como el filtro de seguridad se encuentran en la zona sur, que será la parte que soporte la mayor los trabajos con mayor intensidad en los primeros meses. La ampliación del edificio supondrá un incremento del 42 por ciento en la superficie útil del edificio, así como una redistribución de espacios y dotación de medios que permitirá agilizar de forma notable los diferentes procesos aeroportuarios: desde la facturación hasta el filtro de pasajeros, embarque y desembarque, control de pasaportes o recogida de equipajes.

El proyecto de remodelación, contempla el desarrollo de un área destinado específicamente a tráfico internacional, debido a la proyección que ha adquirido. Esta zona incorporará también servicio de comercio y restauración.

Aprovechando el cambio sustancial que va a tener el aeropuerto de Sevilla, desde este trabajo se pretende colaborar con la seguridad. Se analizarán dos posibles ataques que pueden producirse contra el aeropuerto y, mediante un algoritmo eficiente y exacto para resolver juegos Bayesianos Stackelberg, se proporcionará una solución óptima que sin duda contribuirá a tener una sistema de vigilancia más fiable y robusto.

4. TIPOS DE ATAQUES

Se considerarán dos tipos de ataques: físico e informático. El ataque físico contempla cualquier incidente que pueda producirse cuya responsabilidad recae sobre personas (ataque terrorista, accidentes de tráfico, etc), mientras que el informático son aquellos que impactan directamente a la red informática del aeropuerto pudiendo alterar programas informáticos de planificación, seguimiento, datos…etc.

Estos ataques serán las estrategias que lleve a cabo el atacante/seguidor para vulnerar la seguridad del aeropuerto teniendo como objetivos más amenazantes accidentes aéreos, secuestros, posibles atentados, el colapso del tráfico aéreo o la obtención de datos confidenciales entre otros.

4. 1. ATAQUES INFORMÁTICOS

Se analizarán 8 tipos de ataques comunes sobre los cuáles el defensor/líder, en nuestro caso el aeropuerto, se defenderá con 6 estrategias de defensa para contrarrestarlos. A continuación se definen los tipos de ataques y defensa del seguidor y el líder respectivamente:

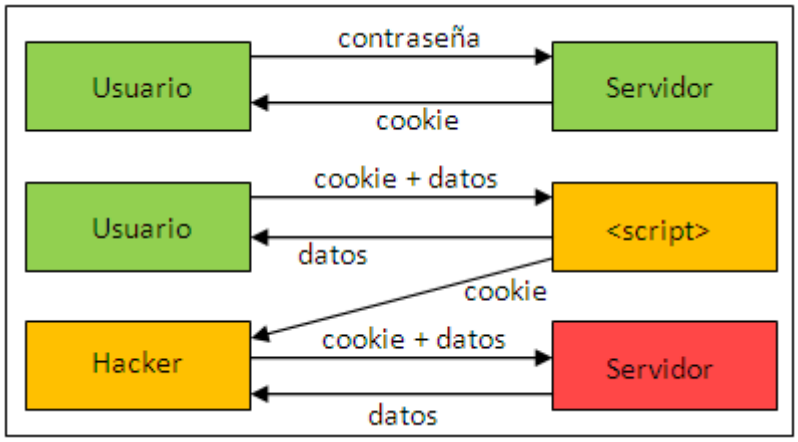
Tipos de ataque:

SQLi→ Los ataques de inyección SQL utilizan las vulnerabilidades del lenguaje de consulta estructurado (SQL) que se utiliza normalmente en las bases de datos relacionales para inhabilitar o penetrar en las bases de datos. Se ha convertido en un modo cada vez más destacado de sabotaje e infiltración en sitios web, en los que se ataca los distintos tipos de funcionalidades asociadas. Los ataques de inyección SQL tienen por objetivo atacar directamente una aplicación o reenviar lógica a una base de datos donde realmente pueden poner en peligro los datos almacenados en su interior.

Las inyecciones SQL son actualmente el segundo tipo de amenaza de más importante (27 %) y atacan a los sitios web mediante la introducción de afirmaciones SQL en un formulario web para saturar la base de datos asociada.

XSS→ Es un ataque de inyección de código malicioso para su posterior ejecución que puede realizarse a sitios web, aplicaciones locales e incluso al propio navegador.

Sucede cuando un usuario mal intencionado envía código malicioso a la aplicación web y se coloca en forma de un hipervínculo para conducir al usuario a otro sitio web, mensajería instantánea o un correo electrónico. Así mismo, puede provocar una negación de servicio (DDos).



Operación de un ataque XSS

Generalmente, si el código malicioso se encuentra en forma de hipervínculo es codificado en HEX (basado en el sistema de numeración hexadecimal, base 16) o algún otro, así cuando el usuario lo vea, no le parecerá sospechoso. De esta manera, los datos ingresados por el usuario son enviados a otro sitio, cuya pantalla es muy similar al sitio web original.

De esta manera, es posible secuestrar una sesión, robar cookies y cambiar la configuración de una cuenta de usuario

BASM (Broken Authentication and Session Management)→ Las funciones de las aplicaciones relacionadas con la autentificación y la administración de servidores a menudo no se implementan adecuadamente.

IDOR→ Insecure Direct Object Reference, también llamado IDOR. Se refiere a cuando una referencia a un objeto de implementación interna, tal como un archivo o llave de base de datos, se expone a los usuarios sin ningún otro control de acceso. Según el curso de protección de datos

personales, el atacante puede manipular esas referencias para obtener acceso a los datos no autorizados.

SM (Security Misconfiguration) La configuración incorrecta de la seguridad surge cuando la configuración de seguridad se define, implementa y mantiene como valores predeterminados. La buena seguridad requiere una configuración segura definida e implementada para la aplicación, el servidor web, el servidor de base de datos y la plataforma. Es igualmente importante tener el software actualizado.

FRUA (Failure to Restrict URL Access)→ Un problema común en las aplicaciones web, al no restringir el acceso a la URL, ocurre cuando una página no tiene implementada la política de control de acceso correcta. Los usuarios no autorizados pueden ver contenido que no deberían tener la capacidad de ver.

Tener estas vulnerabilidades en su aplicación expone una funcionalidad privilegiada a usuarios no autorizados. También puede crear un problema con sus pistas de registro. Si los usuarios pueden acceder a los registros sin ser autenticados, la cadena de custodia se rompe por completo, lo que impide que se realicen las buenas auditorías.

El hecho de no restringir el acceso a la URL también puede ocasionar problemas al omitir la administración de sesiones, otro de los 10 principales de OWASP

ITLP (Insufficient Transport Layer Protection)→ La protección insuficiente de la capa de transporte es una debilidad de seguridad causada por las aplicaciones que no toman ninguna medida para proteger el tráfico de la red. Durante la autenticación, las aplicaciones pueden usar SSL / TLS, pero a menudo no pueden utilizarlo en ningún otro lugar de la aplicación, lo que deja los datos y los identificadores de sesión expuestos. Los datos expuestos y las ID de sesión pueden ser interceptados, lo que significa que la aplicación es vulnerable a la explotación.

CSRF (Cross Site Request Forgery)→ Un ataque CSRF fuerza al navegador web validado de una víctima a enviar una petición a una aplicación web vulnerable, la cual entonces realiza la acción elegida a través de la víctima. Al contrario que en los ataques XSS, los cuales explotan la confianza que un usuario tiene en un sitio en particular, el cross site request forgery explota la confianza que un sitio tiene en un usuario en particular

Tipos de Defensa:

ER→(Escaping routines) Básicamente es un sistema escape seguro ante muchos tipos de ataque permitiendo al sistema atacado enmascararse bajo una interfaz diferente.

SS→ (Session security) El módulo de sesión no puede garantizar que la información que se almacena en una sesión sea vista sólo por el usuario que creó la sesión. Se necesita tomar medidas adicionales para proteger activamente la integridad de la sesión, dependiendo del valor asociado con ella.

Hay varias maneras de filtrar un id de sesión existente a terceros. Un id de sesión filtrada habilita al tercero a acceder a todos los recursos que están asociados con un id específico. Primero, las URL portan id de sesiones. Si enlaza con un sitio externo, la URL incluído el id de sesión podrían estar almacenados en el registro de consultas del sitio externo. Segundo, un atacante más activo podría escuchar su tráfico de red. Si no está encriptado, el id de sesión fluirá en texto plano por la red. La solución aquí es implementar SSL en su servidor y hacerlo obligatorio para los usuarios.

AC→ (Access Control) En seguridad informática, el control de acceso general incluye autenticación, autorización y auditoría. Una definición más restringida de control de acceso cubriría solo la aprobación de acceso, por lo que el sistema toma la decisión de otorgar o rechazar una solicitud de acceso de un sujeto ya autenticado, según lo que el sujeto está autorizado a acceder. La autenticación y el control de acceso a menudo se combinan en una sola operación, de modo que el acceso se aprueba en función de una autenticación exitosa o en un token de acceso anónimo. Los métodos y tokens de autenticación incluyen contraseñas, exploraciones biométricas, claves físicas, claves y dispositivos electrónicos, rutas ocultas, barreras sociales y monitoreo por humanos y sistemas automatizados.

CSRFG (OWASP CSRFGuard)→ Implementa una variante del patrón de token del sincronizador para mitigar el riesgo de ataques CSRF. Para implementar este patrón, CSRFGuard debe ofrecer la capacidad de colocar el token de prevención CSRF dentro del HTML producido por la aplicación web protegida.

ES (Enviroment security)→ El uso generalizado de computadoras en instalaciones militares y de defensa ha requerido durante mucho tiempo la aplicación de las reglas y regulaciones de seguridad. Un principio básico que subyace a la seguridad de los sistemas informáticos ha sido tradicionalmente

el de 'aislamiento: simplemente eliminar todo el sistema a un entorno físico en el que la penetrabilidad se minimiza de forma aceptable. El uso cada vez mayor de sistemas en los que algunos componentes del equipo, como las terminales de acceso de usuarios, están ampliamente distribuidos geográficamente, ha introducido nuevas complejidades y problemas. Estos problemas no son susceptibles de solución a través de la protección elemental del aislamiento físico. Así pues se requiere de una compilación de técnicas y procedimientos que deben considerarse por separado y en combinación al diseñar o adoptar sistemas de procesamiento de datos para brindar seguridad o privacidad al sistema.

SSL (Secure Sockets Layer)→ Es un protocolo diseñado para permitir que las aplicaciones para transmitir información de ida y de manera segura hacia atrás. Las aplicaciones que utilizan el protocolo Secure Sockets Layer sí sabe cómo dar y recibir claves de cifrado con otras aplicaciones, así como la manera de cifrar y descifrar los datos enviados entre los dos.

¿Cómo funciona el SSL? Algunas aplicaciones que están configurados para ejecutarse SSL incluyen navegadores web como Internet Explorer y Firefox, los programas de correo como Outlook, Mozilla Thunderbird, Mail.app de Apple, y SFTP (Secure File Transfer Protocol) programas, etc Estos programas son capaces de recibir de forma automática SSL conexiones.

Para establecer una conexión segura SSL, sin embargo, su aplicación debe tener una clave de cifrado que le asigna una autoridad de certificación en la forma de un Certificado. Una vez que haya una única clave de su cuenta, usted puede establecer una conexión segura utilizando el protocolo SSL

Para cada uno de los ataques mencionados anteriormente, existe una estrategia de defensa óptima para contrarrestarlos. La recompensa/éxito del líder será la adecuada según qué tipo de defensa utilice para cada uno de los ataques. A continuación, se muestra las estrategias de defensa óptimas para cada uno de los ataques:

Estrategias Ataque del Seguidor	Estrategias Defensa del Líder
SQL	ER
XSS	ER
BASM	SS
IDOR	AC
CSRF	CSRFG
SM	ES
FRUA	AC
ITLP	SSL

Utilizar una estrategia de defensa o ataque conlleva un impacto sobre el adversario el cual será beneficioso o perjudicial según sea la estrategia adecuada o no. Del mismo modo, cada acción lleva asociada un coste o fallo. Se adjunta también las tablas de Impacto-Coste/Fallo-Éxito para del Líder y del Seguidor:

LÍDER			
Estrategia Defensa	Impacto	Coste/Fallo	Éxito
ER	8	-1	7
SS	4	-2	2
AC	2	-2	0
CSRFG	4	-2	2
ES	3	-1	2
SSL	4	-2	2

SEGUIDOR			
Estrategia Ataque	Impacto	Coste/Fallo	Éxito
SQL	4	-1	3
XSS	4	-1	3
BASM	4	-2	2
IDOR	1	-0,5	0,5
CSRF	4	-2	2
SM	3	-1	2
FRUA	1	-0,5	0,5
ITLP	4	-3	1

Evaluemos pues la gravedad de estos ataques y las soluciones relativas y estimemos las recompensas para el líder y el seguidor, así como, el costo de estas acciones. Para calcular la matriz de Coste/Fallo tenemos en cuenta lo siguiente:

Si la estrategia de defensa es efectiva a la del ataque:

$$\textit{Recompensa del Líder} = \textit{Impacto de la acción} - \textit{Coste de fijar}$$

$$\textit{Recompensa del atacante} = \textit{Coste del ataque}$$

Si la estrategia de ataque es efectiva a la defensa la recompensa del líder es negativa debido al coste el fijado y al impacto del seguidor.

ATAQUE INFORMÁTICO.

	SQLi	XSS	BASM	IDOR	CSCF	SM	FRUA	ITCP
ER	7/-1	7/-1	-5/2	-2/0.5	-5/2	-4/1	-2/0,5	-5/1
SS	-6/3	-6/3	2/-2	-3/0,5	-6/2	-5/2	-3/0,5	-6/1
AC	-6/3	-6/3	-6/2	0/-0,5	-6/2	-5/2	0/-0,5	-6/1
CSRFG	-6/3	-6/3	-6/2	-3/0,5	2/-2	-5/2	-3/0.3	-6/1
ES	-5/3	-5/3	-5/2	-2/0,5	-5/2	2/-1	-2/0,5	-5/1
SSL	-6/3	-6/3	-6/2	-3/0,5	-6/2	-5/2	-3/0,5	2/2

En todo supondremos que el ataque informático tiene probabilidad 0,7 de producirse ante el físico que tiene probabilidad 0.3. También tendremos en cuenta que los ataque óptimos de los followers son estrategias puras, $q_i \in \{0,1\}$.

Cálculo con Mathematica

Max 0.7[7 Z11+7 Z12-5 Z13-2 Z14 -5 Z14-5 Z15-4 Z16-2 Z17-2 Z18-6 Z21-6 Z22+2 Z23-3 Z24 -6 Z25-5 Z26-3 Z27-6 Z28-6 Z31-6 Z32-6 Z33+0 Z34 -6 Z35-5 Z36+0 Z37-6 Z38+*

-6 Z41-6 Z42-6 Z43-3 Z44 +2 Z45-5 Z46-3 Z47-6 Z48-5 Z51-5 Z52-5 Z53-2 Z54 -5 Z55+2 Z56-2 Z57-5 Z58-6 Z61-6 Z62-6 Z63-3 Z64 -6 Z65-5 Z66-3 Z67+2 Z68]

SUJETO A

$$\sum_{IJ} ZIJ = 1, I: 1, \dots, 6; J: 1, \dots, 8. \qquad \sum_{I} ZIJ \leq 1, \forall J: 1, \dots, 8$$

para calcular a^l hemos de resolver el programa dual asociado al atacante para cada tipo de ataque.

$$min_a a$$

s.t.

$$a \geq \sum_{i \in X} C_{ij}\, x_i, \qquad j \in Q$$

En particular tenemos 8 problemas, uno por cada tipo de ataque.

Max $\sum_{i \in X} C_{ij}\, x_i,\ j: 1, \ldots, 8,$

s.t. $\sum_{i \in X} x_i = 1$

Así para j=1,

$$\text{Max} \sum_{i \in X} C_{i1}\, x_i$$

$$\text{s.t.} \sum_{i \in X} x_i = 1$$

Para j=1 resulta x_2=1 y a_1=3

Para j=2 resulta lo mismo y a_2=3

Para j=3 resulta $x_1 = 1$ y a_3=2

Para j=4 resulta $x_1 = 1$ y a_4=0.5

Para j=5 resulta $x_1 = 1$ y a_5=2

Para j=6 resulta $x_2 = 1$ y a_6=2

Para j=7 resulta $x_1 = 1$ y a_7=0.5

Para j=8 resulta $x_1 = 1$ y a_8=2

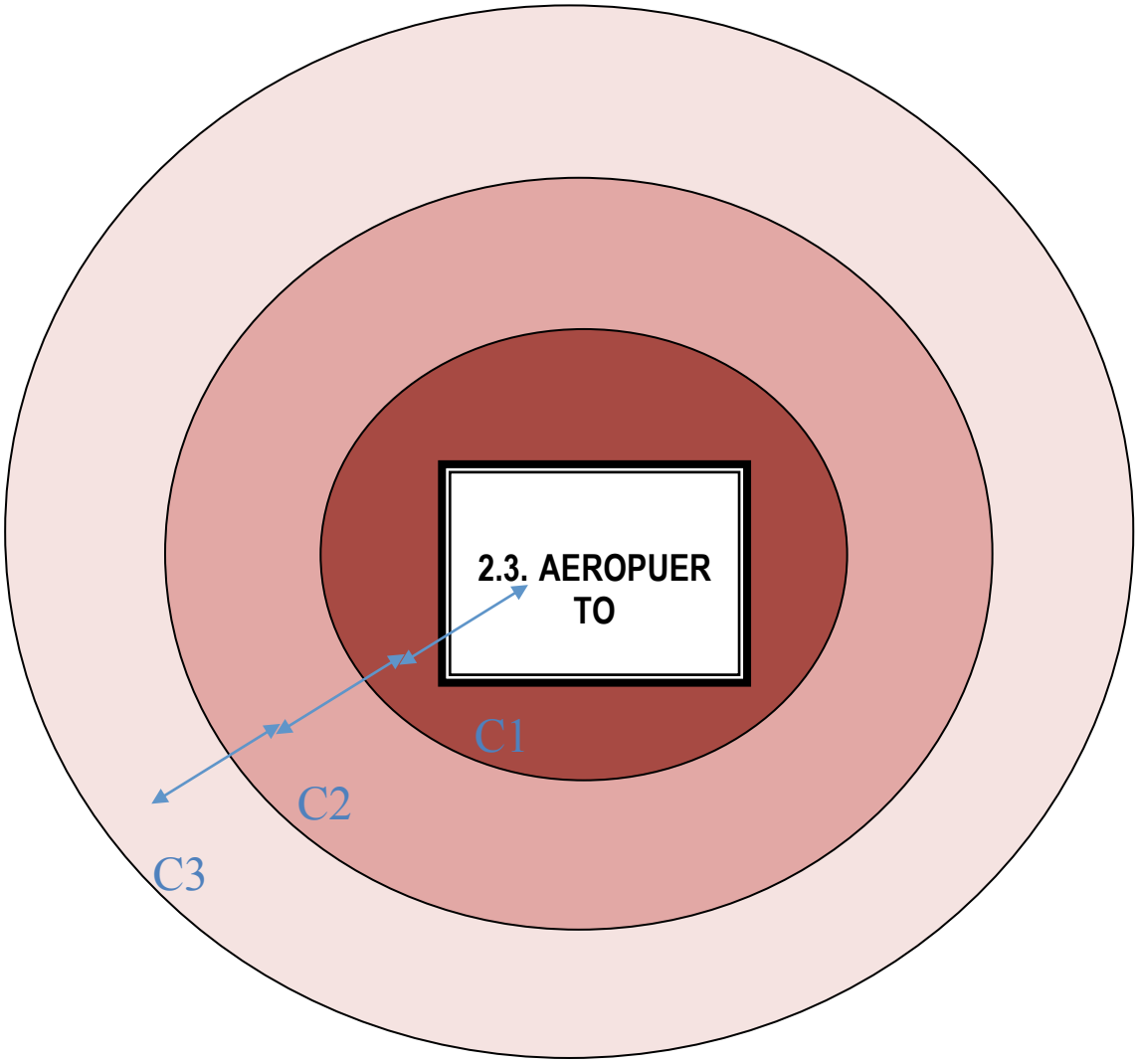
Así tenemos 8 desigualdades

$$\left(\sum_{i\in X} {C_{ij}}^{l} \left(\sum_{h\in Q} {z_{ih}}^{l}\right)\right) \leq a_j;\ \text{j=1,…8.}$$

La conclusión luego de correr el programa nos da como solución x1=0.625 y x5=0.375 con un beneficio de 5,125. Estrategia combinada. q1=1 o q6=1 como mejor ataque teniendo un beneficio máximo de 0.5 en ambas.

4. 2. Ataques Físicos

Para analizar los posibles ataques físicos, dividiremos el perímetro de vigilancia del aeropuerto en tres círculos de manera que conforme estemos más cerca del aeródromo, la vigilancia debe ser más fuerte. La criticidad de los perímetros identificados va aumentando conforme nos acercamos al punto central del mismo (terminal), pensando en el impacto que puede tener un ataque físico contra dicha terminal, zona de más afluencia de pasajeros. De una manera gráfica tendríamos el siguiente planteamiento de vigilancia:



Se elabora un plan de vigilancia de manera que el círculo 1 siempre esté vigilado. En este sentido, tendremos tres estrategias para defender el aeropuerto:

ALERTA 1→ Estrategia para defender el perímetro del círculo C1

ALERTA 2→ Estrategia para defender el perímetro del círculo C1 y el círculo C2

ALERTA 3→ Estrategia para defender el perímetro del círculo C1 y el círculo C3

Por otra parte, se han identificado los posibles ataques físicos que puede sufrir el aeropuerto:

ATAQUE 1→ Estrategia para atacar el perímetro del círculo C1

ATAQUE 2→ Estrategia para atacar el perímetro del círculo C2

ATAQUE 3→ Estrategia para atacar el perímetro del círculo C3

A continuación evaluamos la gravedad de estos ataques y las soluciones relativas. Estimamos la recompensa para el líder y el seguidor, así como, el coste de las acciones. Para ello utilizamos la siguiente tabla de impactos y costes del Líder y del Seguidor:

LÍDER			
Estrategia Defensa	Impacto	Coste/Fallo	Éxito
ALERTA 1	3	-3	0
ALERTA 2	2	-2	0
ALERTA 1	1	-1	0

SEGUIDOR			
Estrategia Ataque	Impacto	Coste/Fallo	Éxito
ATAQUE 1	3	-3	0
ATAQUE 2	2	-2	0
ATAQUE 3	1	-1	0

	ATAQUE 1	ATAQUE 2	ATAQUE 3
ALERTA 1	0/-3	0/-2	0/-1
ALERTA 2	0/-3	0/-2	4/1
ALERTA 3	0/-3	2/2	0/-1

Maximize[0.2(2 z23+2z32)]

SUJETO A

$$\sum_{IJ} ZIJ = 1, I: 1, \dots, 3; J: 1, \dots, 3. \qquad \sum_{I} ZIJ \leq 1, \forall J: 1, \dots, 3$$

y con tres restricciones más surgidas del programa para el atacante.

$$min_a a$$

s.t.

$$a \geq \sum_{i \in X} C_{ij}\, x_i, \qquad j \in Q$$

j=1 x1=1 y a=-3

j=2 x3=1 y a=2

j=3 x2=1 y a=1

Luego las tres desigualdades son

$$\left(\sum_{i \in X} {C_{ij}}^{l} \left(\sum_{h \in Q} {z_{ih}}^{l}\right)\right) \leq a_j;\ \text{j=1,2,3.}$$

Haciendo los mismos cálculos llegamos a

Z23=1 con ganancia 0.4. Es decir la estrategia optima para el defensor es la 2 y para el atacante la 3. La ganancia es de 0.6 para el defensor.

In[13]:= Maximize[{0.3 (2 Z23 + 2 Z32), Z11 + Z12 + Z21 + Z22 + Z13 + Z23 + Z31 + Z32 + Z33 == 1, Z11 >= 0, Z12 >= 0, Z13 >= 0, Z21 >= 0, Z22 >= 0, Z23 >= 0, Z31 >= 0, Z32 >= 0, Z33 >= 0, -3 (Z11 + Z12 + Z13) - 3 (Z21 + Z22 + Z23) - 3 (Z31 + Z32 + Z33) <= -3, -2 (Z11 + Z12 + Z13) - 2 (Z21 + Z22 + Z23) - 2 (Z31 + Z32 + Z33) <= 2, -1 (Z11 + Z12 + Z13) + (Z21 + Z22 + Z23) - (Z31 + Z32 + Z33) <= 1}, {Z11, Z12, Z13, Z21, Z22, Z23, Z31, Z32, Z33}]

Out[13]= {0.6, {Z11 -> 0., Z12 -> 0., Z13 -> 0., Z21 -> 0., Z22 -> 0., Z23 -> 1., Z31 -> 0., Z32 -> 0., Z33 -> 0.}}

En definitiva la estrategia conjunta de defensa óptima es usar ER con problabilidad 0.625 y ES con probabilidad 0.375. Esto junto a más la vigilancia de los anillos 1 y 2 con probabilidad 1. el ataque más pernicioso es el ataque SQLi y el ataque SL más el ataque al anillo 3. El defensor obtiene un beneficio de 0.6+4.9=5.5. Es la máxima expectativa de defensa y un coste máximo de 0.7*0.5+0.3*(-3)=-0,55 expectativa máxima de recompensa por parte del seguidor.

5. CONCLUSIONES

Hemos tratado en este trabajo un problema entre dos jugadores de forma que uno de ellos tiene información incompleta.

El primer jugador es un aeropuerto que debe defenderse de los ataques web a su sistema. Los atacantes conocen los sistemas de defensa del aeropuerto pero este no sabe como va a ser atacado.

La misma situación la hemos tratado para una defensa "llamada física" en la cual son susceptibles de ser atacados 3 anillos de control del aeropuerto defendidos por policías y perros.

Hemos modelado la situación como un **Juego Bayesiano Stackelberg:** "es un juego de Stackelberg donde el líder tiene incertidumbre sobre los tipos de adversarios que puede encontrar."

Para solucionarlo hemos elegido el sistema DBOS que es un algoritmo que permite dividir en problemas de optimización lineales un sistema de optimización no lineal y que permite hallar siempre una solución óptima porque por sus características evita las sub-optimalidades.

Como conclusión hemos encontrado una solución mixta de defensa del sistema web y una solución no mixta del física, es decir, una defensa posible sobre las otras.

Este trabajo es susceptible de ampliarse a situaciones de incertidumbre y particularizarse a sistemas más complejos o detallados.

6. REFERENCIAS

[1] M. Tambe, M. Jain, J. A. Pita and A. Jiang, "Game theory for security: key algorithmic principles, deployed systems, lessons learned", in 50th Annual Allerton Conference on Communication, Control, and Computing, pp. 1822-- 1829, 2012.

[2] P. Paruchuri, J. P. Pearce, J. Marecki, M. Tambe, F. Ordonez and S. Kraus, "Playing games with security: an efficient exact algorithm for Bayesian Stackelberg games", in Proc. of The 7th InternationalConference on Autonomous Agents and Multiagent Systems (AAMAS), pp. 895--902, 2008.

[3] D. Korzhyk, Z. Yin, C. Kiekintveld, V.Conitzer and M.Tambe, "Stackelberg vs nash in security games - an extended investigation of interchangeability, equivalence and uniqueness", Journal of Artificial Intelligence Research, vol. 41, pp. 297--327, 2011.

[4] V. Conitzer and T. Sandholm, "Computing the optimal strategy to commit to", in Proc. of the 7th Association for Computing Machinery(ACM) Conference on Electronic Commerce (EC), pp. 82--90, 2006.

[5] M. Jain, E. Kardes, C. Kiekintveld, M. Tambe and F. Ordonez, "Security games with arbitrary schedules: A branch and price approach", in Proc. of Association for the Advancement of Artificial Intelligence (AAAI) Conference on Artificial Intelligence, pp. 792--797, 2010.

[6] (2014) Kaspersky Lab website. [Online]. Available: http://www.kaspersky.com/ 80 (2014) Department for Business and Innovation Skills, UK website. [Online]. Available: https://www.gov.uk/government/organisations/department-for-business- innovation-skills/

Stackelberg Security Games for Information Security Mangement of Financial System