

Automated Security Testing

Un framework para testeo de
seguridad, mediante ataques
controlados.

alejandro david weil
matias eissler

Disclaimer

- ◆ Acerca de Core Security
- ◆ ¿Qué es penetration test?
- ◆ ¿Qué es un ataque?

Módulo en Impact

- Entities
- Módulos

Entities

Representan los objetivos atacables para Impact.
Nosotros estamos interesados, por ahora
unicamente en Host.

Así mismo, cada entity posee propiedades que
pueden pensarse como diccionarios anidados y se
utilizan para persistir la información que van
recolectando los módulos.

Módulos

- Son las **acciones** para colectar información acerca de los Entities.
- “Todos” (99.3%) los módulos (de IG, exploits, etc) están hechos en python.
- Impact, tiene un interprete de python embebido, y nos permite ejecutar uno o varios modulos a la vez.
- Un tipo especial de módulo son los exploits

Problemas

Automatizar IG y AP

Information Gathering

Pasos necesarios para conocer la mayor cantidad de información posible a cerca de todos los hosts de una red particular.

Information Gathering

- ◆ Descubrir las máquinas de la red
- ◆ Port Scannear (tcp/udp)
- ◆ Detectar OS
- ◆ Detectar servicios

Network Discovery

- Primer paso de IG
- Distintos métodos
- Eficiencia
- Requerimientos
- Futuros métodos

Port Scanners



Attack and Penetrate

Es el paso de lanzar ataques contra una máquina para lograr acceso a la misma.

Eligiendo el próximo ataque

- Como elegir el orden
 - Probabilidad de éxito
 - Exploits que no desestabilicen
 - Brute-forcecs
 - Privilegios
 - otros...

Automatizado de IG y AP

También conocido como
Rapid Penetration Test.

- ◆ Ejecución automática
- ◆ Facilidad de uso.

Objetivos de la automatización: Paralelización de acciones

Suponiendo una red con 150 maquinas, 120 exploits,
1 minuto por exploit:

$$120 * 150 * 1 \text{min} = 18000 \text{min} = 300 \text{ horas} = \\ 12.5 \text{ días.}$$

Ejecutar acciones en paralelo resulta mucho mas eficiente.

Customización

Brindar al usuario, la
posibilidad de que elija de
que forma quiere realizar
el penetration test.

Flexibilidad

Ante la aparición de un nuevo módulo que realiza alguna de las tareas ya descriptas, pero que tendrá, ventajas y desventajas (mayores requerimientos), la actualización automática de todo el framework.

Reutilizabilidad

Queremos un esquema de automatización que nos sirva para IG y AP actuales, pero que pueda servirnos para futuras versiones

Nuestro problema

¿Cómo ordenar el trabajo?

¿Cómo expresar las dependencias?

¿Qué trabajo ejecutar en paralelo?

Ejecución de tareas

- ◆ El problema de distribuir y ejecutar tareas (job scheduling) es bastante antiguo.
- ◆ En la industria se encuentra el mismo problema:
 - En que orden cortar piezas optimizando el tiempo
 - En que orden combinar la salida de una máquina con otra, si las distintas tareas que realizan toman distintos tiempos

Mas scheduling

- El ejemplo mas conocidos para nosotros es el scheduler de un OS
 - Distribuir el procesador entre distintas tareas
 - Prioridades de las tareas
 - Tareas bloqueadas por I/O “a la espera”

Automated Planning

- Consiste en planear como conseguir un objetivo, dado el conocimiento de las tareas atómicas que se pueden ejecutar
- Las soluciones mas comunes, consisten en hacer grafos de las tareas, y encontrar el camino hacia la solución buscada
- Quiero que un robot barra la terraza y se cuales son las acciones que puede realizar, la cuestión es como ordenar las tareas para obtener el resultado deseado

Automated Planning

- ♦ El plan seria:
 - Robot agarrar la escoba
 - Subir a la terraza
 - Barrer (*)
 - Bajar la escalera
 - Dejar la escoba

* Barrer se puede modelar nuevamente como otro objetivo a resolver con tareas, etc..

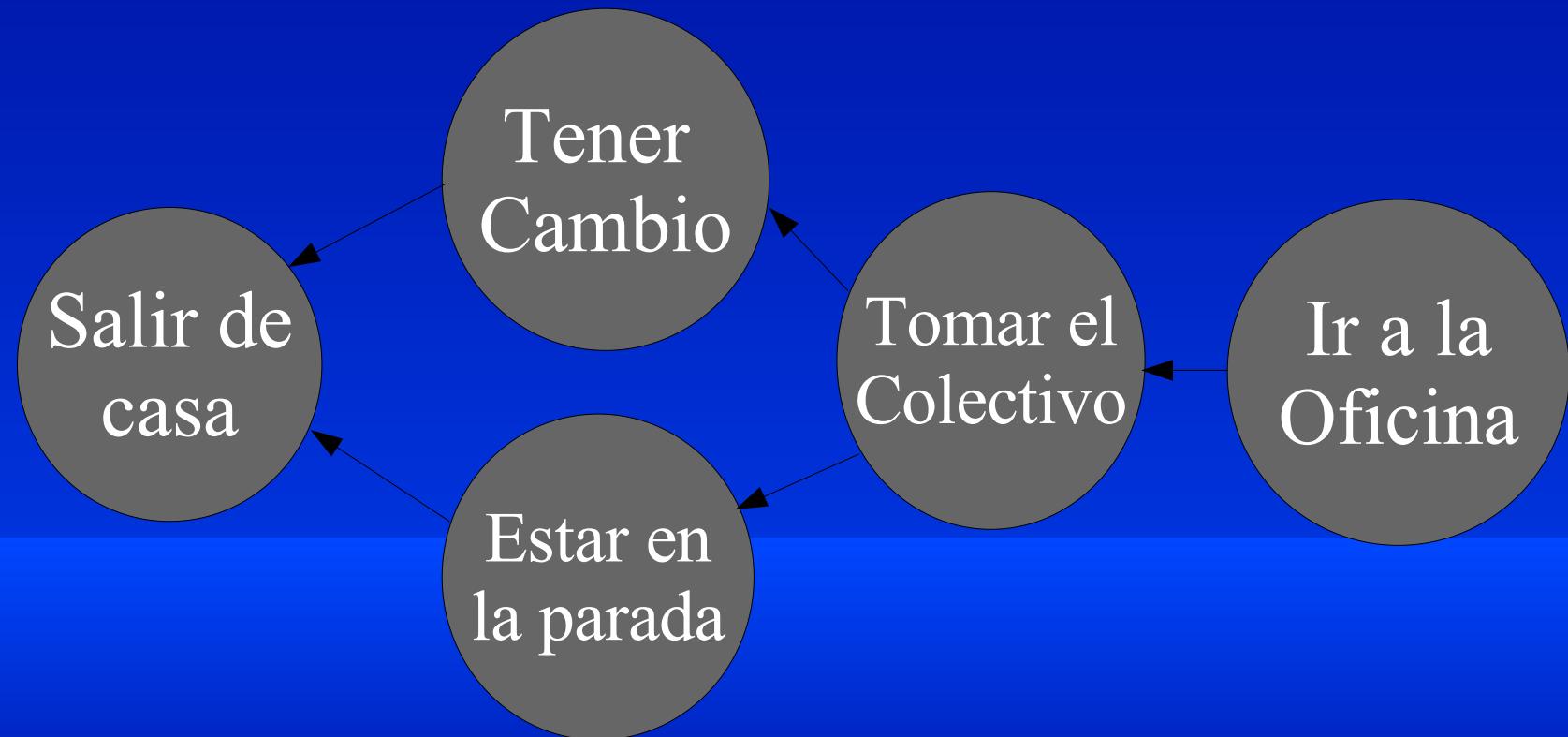
Planning y Scheduling

- La diferencia entre un planner y un scheduler por ahi no esta clara a primera vista:
- El Planner, especificara en que orden realizar que tareas
- El Scheduler, priorizara su ejecucion en el momento en el que las tareas ya han sido designadas para su ejecución

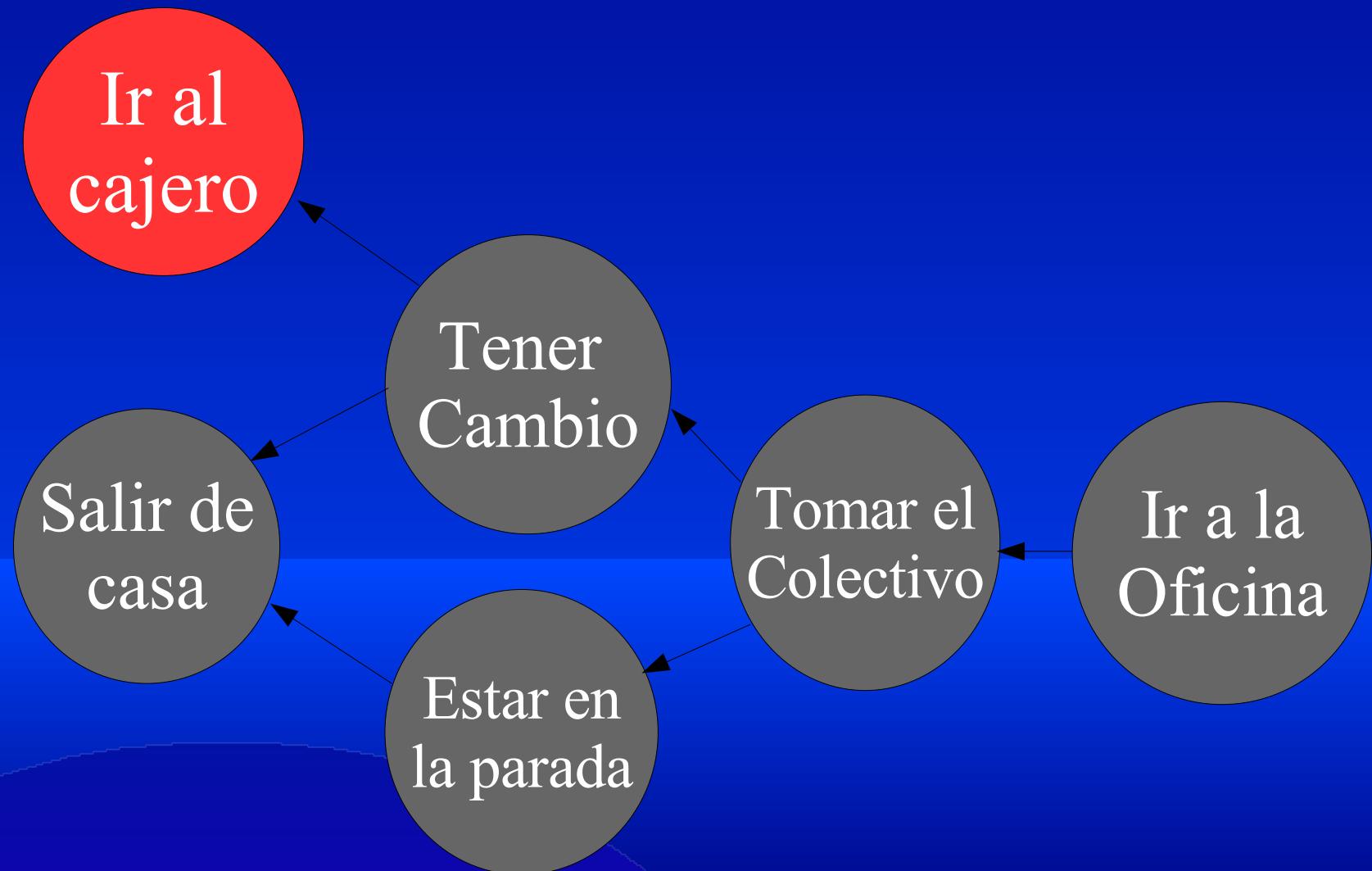
Solución

- Grafo de Goals
 - Dependencias
 - Verificación de estado

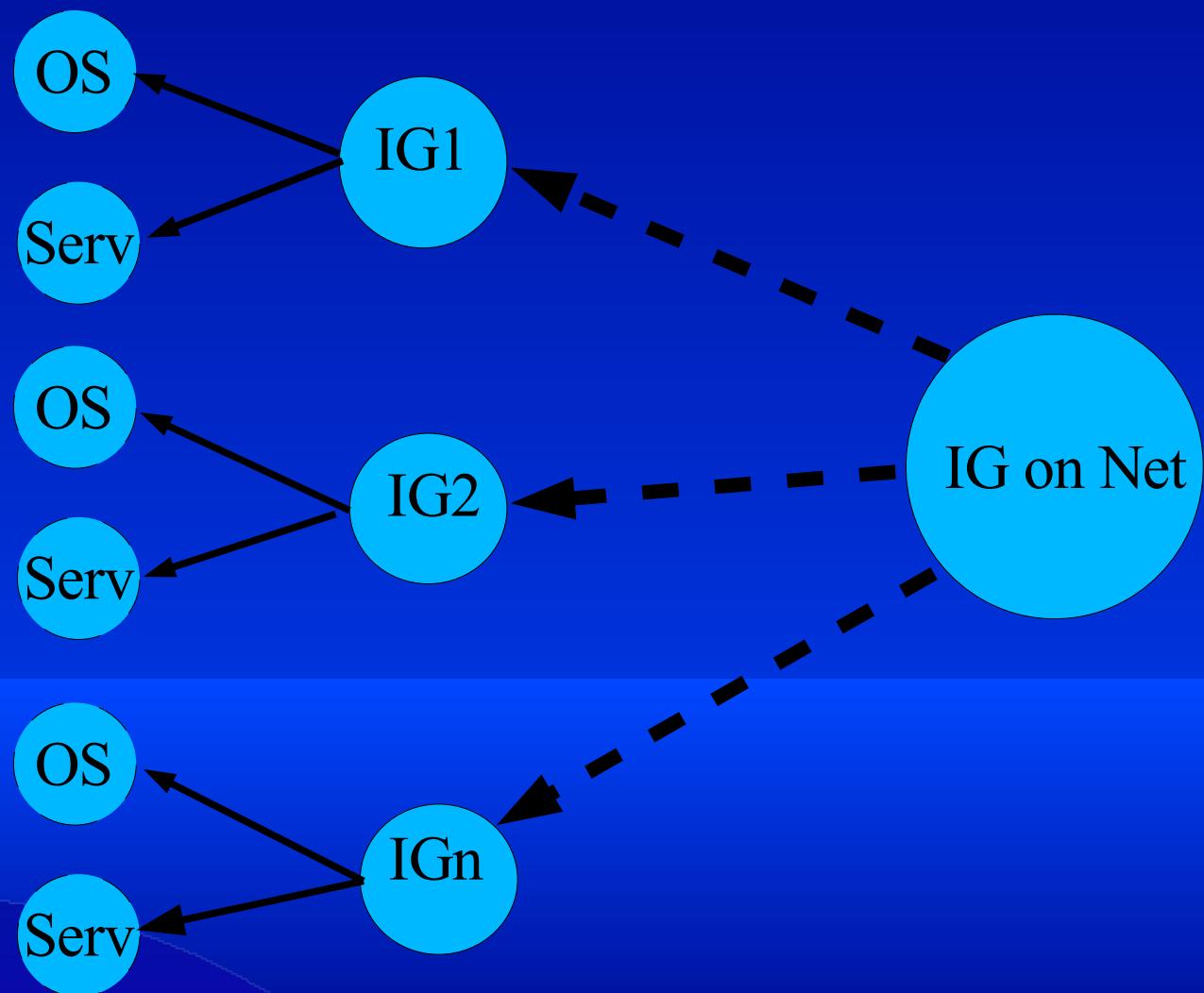
Dependencias Plantime



Dependencias Runtime



Dependencias en paralelo



Estado de los goals

- Válido
- Inválido
- Desconocido

Goals Compuestos

- All
- Most
- Any

Planner

Final Goal

Dependencias



Grafo

Strategy → Tareas → Scheduler

Strategy

- ◆ Distintas formas de hacer lo mismo
- ◆ Dependencias en run time
- ◆ Tareas no existosas

Scheduling

- Maximizar ejecución en paralelo
- Minimizar la cantidad de tareas

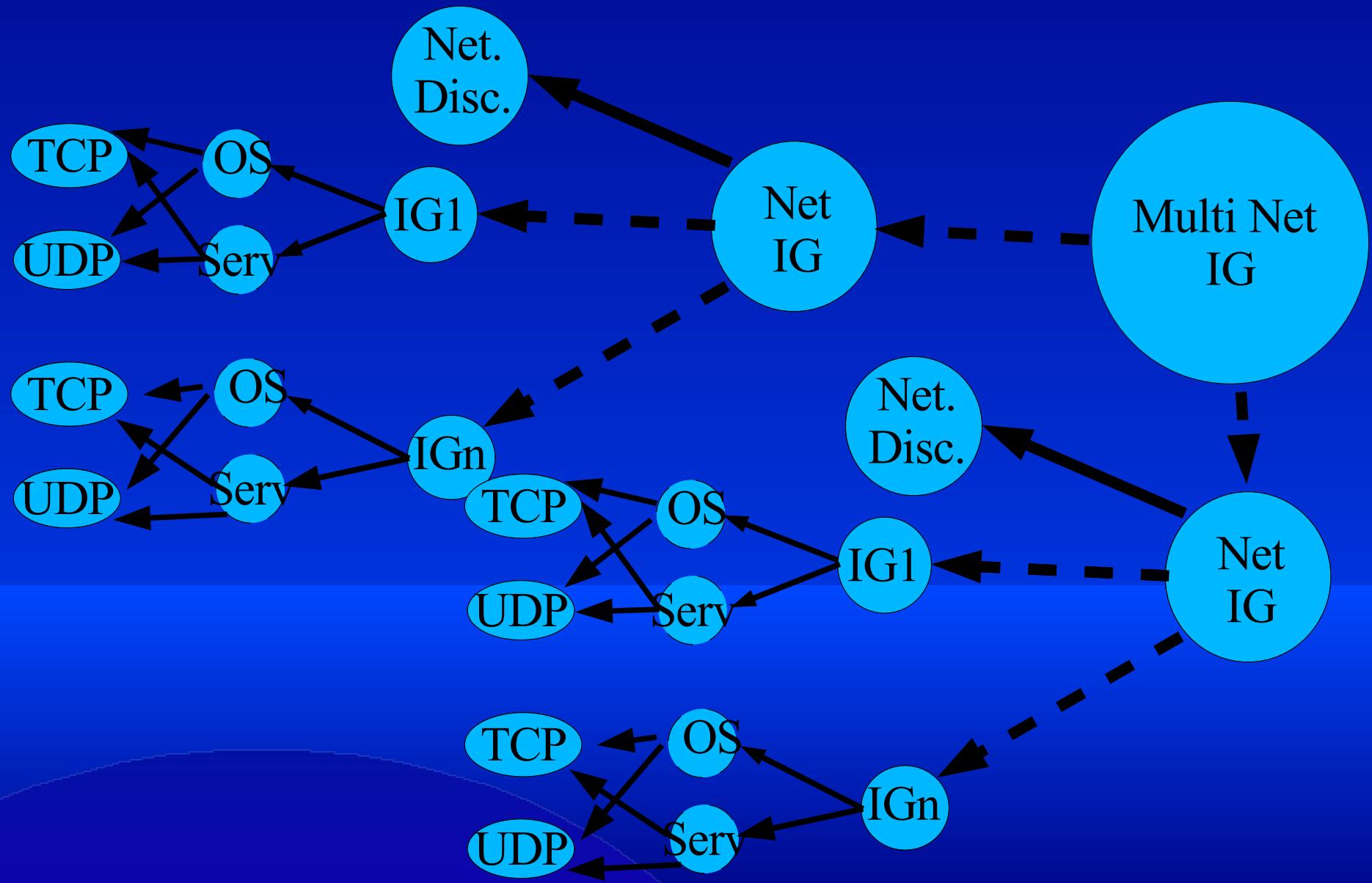
Interacciones

Scheduler Planner Strategy

Push
↔
Señal de Fin

Goals
↔
Tarea

Ejemplo de IG completo



nota sobre Python

- Usar Python nos brindó las librerías necesarias de grafos
- Nos permitió trabajar con el prototipo mientras se trabajaba el diseño y luego “portarlo” a usar las api del sistema
- Nos permitió tener el framework funcionando en tiempo mínimo

Agradecimientos

- ◆ LugLi
- ◆ UTN (Santa Fé)
- ◆ Core Security Technologies
- ◆ PyAr