Intrusion Detection Systems

Intrusione, controllo e contenimento nel digital warfare

Raistlin @ s0ftpj – Hackmeeting 2002 – Bologna, 22/05/02

I tre obiettivi della sicurezza

- *Confidenzialità:* solo le persone autorizzate possono accedere al sistema informativo
- *Integrità:* solo persone autorizzate possono modificare componenti del sistema, e solo nelle modalità per cui sono state autorizzate a procedere
- *Disponibilità:* il sistema deve fornire i servizi richiesti in un tempo "ragionevole" secondo i requisiti
- Obiettivi addizionali (casi specifici dei tre sopra):
  - Non-repudiation: un messaggio spedito deve poter essere "provato" e non poter essere negato
  - Privacy: forma particolare di confidenzialità che mira a proteggere uno specifico utente.
  - Identificabilità degli utenti
  - Accountability (logging)

Il formalismo della sicurezza

- **Confidenzialità:** Accesso alle risorse *only if* utente autorizzato (relazione molti – a – molti)
- **Integrità:** Modifica delle risorse *only if* utente autorizzato (relazione molti – a – molti)
- **Disponibilità:** *if* utente_autorizzato *and* richiesta_accesso(t) *then* accesso_consentito(t+k) and k < delta, con delta nelle specifiche.
- *La relazione di sicurezza è (deve essere) un IF AND ONLY IF !*

Implementazione Classica

- *Confidenzialità – Integrità – Disponibilità* sono definite da una relazione tra:
  - UTENTI opportunamente identificati
  - RISORSE cui possono accedere
- Meccanismi di *identificazione all'accesso*, utilizzo di *permessi*, dispositivi di *auditing* per verificare il buon funzionamento del tutto
- Meccanismi pervasivi di controllo dei permessi a runtime (esempio: processi in ambiente UNIX)

Un problema difficile

- Purtroppo il parallelo con la logica regge fino a un limite, ovvero fino al limite in cui i programmi si comportano conformemente alla specifica
- Per il teorema di Rice non è possibile “provare” in modo automatico la correttezza di un programma (in particolare rispetto ad ingressi eccezionali)
- La verifica manuale dei programmi e la loro scrittura in modo corretto (mediante corretti procedimenti di software engineering) teoricamente funziona ma nella pratica ha dimostrato dei limiti
- Esistono numerosi tipi di “exploit” per sfruttare vulnerabilità del software.

Forza e debolezza: il punto chiave

- Negli ultimi anni, il paradigma classico ha mostrato i suoi limiti
- Il principio dell’identificazione e dell’associazione ai permessi è ancora fondamentale ma non “scala” facilmente alle dimensioni di una WAN, o dell’Internet
- Gli hacker non utilizzano la forza, ma sfruttano le debolezze intrinseche dei sistemi. Il paradigma classico su scala di rete è insicuro, ma non può essere “aggiornato”.
- Logica KISS: Keep It Simple, Stupid.
- Continuare con il paradigma del “Who are you ? What can you do ?” è spesso improduttivo o comunque insufficiente.

Why are you doing this ?

- Torniamo alle origini: *confidenzialità, integrità, disponibilità* hanno un comune denominatore
- Il sistema informatico ha uno *scopo,* e deve servire a quello scopo evitando compromissioni
- Ogni violazione del paradigma CID è visibile perché il sistema compie azioni “anomale”
- Invece di limitarci a chiedere “Who are you ? What can you do ?” cerchiamo di capire: “Why are you doing this?”
- INTRUSION DETECTION SYSTEM, rivelatore di intrusione

Intrusion Detection System

- Chi entra in un sistema informatico abusivamente compie alcuni tipi di azione che un utente normale non farebbe mai
- Identificando queste azioni “anomale” possiamo scoprire un intruso
- Un IDS non si sostituisce ai normali controlli, ma piuttosto cerca di scoprire i loro fallimenti
- Allo stesso tempo non deve essere iperreattivo (mai gridare “Al lupo!”)

Due metodi per gli IDS

- Anomaly Detection: determinare statisticamente modelli di comportamento dell'utente, e segnalare deviazioni "significative" dal modello; *a posteriori* (model-based detection)
- Misuse Detection: confrontare gli eventi con "schemi" predefiniti di attacchi; *a priori* (behaviour-based)

Anomaly Detection Model

- Determiniamo statisticamente dei modelli di comportamento dell'utente e segnaliamo deviazioni significative
- Generico: ha in generale il vantaggio di potersi adattare a "nuovi" schemi d'attacco
- Varie tecniche e approcci: Autoclass (Bayes); identificazione predittiva; uso di reti neurali; model-based detection.
- Problemi principali: scelta delle metriche e dei threshold; scelta dei modelli di base

Misuse Detection Model

- Utilizzo di linguaggi basati su regole per rappresentare la conoscenza relativa ai misuse case (es. in IDES, con P-BEST; oppure ling. RUSSEL)
  - Problema della sequenzialità
  - Solo vulnerabilità conosciute
  - Problemi nel mantenere la knowledge-base
  - Problemi nella uncertain-reasoning
- State–transition analysis (es. in STAT); ancora rule-based ma con approccio diverso
- Utilizzo del pattern-matching (es. in IDIOT)
  - Problema di interleaving degli eventi
  - Discrete Approximate Matching – Longest Common Subsequence

Combinare le feature

- Molti sistemi IDS combinano feature di anomaly e di misuse detection; ad esempio: IDES, Intrusion Detection Expert System
- Segnalazione di eventi "strani" (misuse) combinati con analisi statistiche delle user sessions (anomaly)
- Anomaly Detection: applicazione di metodi statistici (modulo STAT) basati sulla covarianza di un vettore di variabili d'intrusione associate a una determinata attività
- Misuse Detection: sistema esperto realizzato con P-BEST

On-line vs. off-line operations

- On-line operations: il sistema lancia delle alert analizzando gli eventi correnti; normalmente usa una finestra di dati. Spesso per problemi di complessità computazionale questi alert sono limitati a regole attivate da trigger.
- Off-line (batch) operations: il sistema analizza i log (registrati) degli eventi. Può generalmente utilizzare maggiore potenza di calcolo. Può analizzare finestre illimitate nel passato, anche l'intera sessione.
- Integrazione tra i due principi: ISOA; trigger per l'attivazione di alert on-line, l'analisi off-line viene utilizzata per completare le ricerche su eventi giudicati interessanti dall'operatore.

Network Based vs. Host Based

- HOST based: i primi IDS erano host-based; un IDS host-based si appoggia al sistema operativo e controlla le system call (esecuzione e controllo dei processi) e gli accessi (al sistema, ai device...)
- NETWORK based: controllano il traffico sulla rete cercando nel flusso di pacchetti le tracce di una intrusione
- Prossima frontiera: interoperabilità, correlazione, normalizzazione
- Entrambi possono essere Anomaly o Misuse based.

Ruolo dei sistemi esperti

- Quasi tutti gli IDS utilizzano sistemi esperti !
- Sembrerebbe logico immaginare l'utilizzo dei ruleset nella parte host-based, con un modello di misuse detection: si veda IDES, che usa un modello non-rule-based (STAT) per l'analisi d'anomalia.
- Controesempi: MIDAS, Multics Intrusion Detection Alerting System; Wisdom&Sense

MIDAS

- Sistema Real-time sviluppato dal National Computer Security Center
- LISP-based
- Algoritmo forward-chaining con quattro livelli di regole
- Le variazioni dai modelli statistici di comportamento vengono riassunte nella base dei fatti da apposite regole

Wisdom & Sense

- Sviluppato al Los Alamos National Lab
- Statistico e rule-based al tempo stesso !
- Riceve in input i log “storici” di operazioni “normali” e ne deriva un numero impressionante di regole (ordine di 10^4 o più!), tutte al di sotto dei 10 byte. La ricerca (dati del ’94) impiega 50ms, la generazione meno di un’ora.
- Classiche tecniche di espansione e pruning
- Nuovi algoritmi (IREP) potrebbero migliorare la derivazione di regole da una base di fatti !

DIDS e NSM

- DIDS, Distributed Intrusion Detection System; sviluppato a partire da NSM, Network Security Monitor
- NSM categorizzava ogni connessione assegnandole un fattore di rischio, sulla base di vettori multipli di parametri (network-based)
- DIDS aggiunge sensori distribuiti per intercettare intrusi che non accedono via rete (network+host based)
- Esiste un DIDS Director che raccoglie ed elabora dati
- Utilizzando un modulo di nome NID (Network-user ID) tiene traccia dell’utente man mano che si muove tra account multipli sulla rete.

DIDS Director: struttura KBMS

- DIDS usa un modulo knowledge based realizzato in CLIPS
- Le regole gerarchiche usate sono denominate Intrusion Detection Model (IDM) e descrivono la trasformazione dei dati in ipotesi d’alto livello sui tentativi d’intrusione.
- Livelli
  1. Dati specifici di piattaforma
  2. Descrizione platform-independant di eventi
  3. Intervento del NID a correlare gli eventi
  4. Inserimento di spazializzazione (provenienza) e temporalizzazione (assoluta e relativa)
  5. Categorizzazione in attacks, misuses, e suspicious acts
  6. Produzione di un livello overall di “pericolo”

Caratteristiche dei pattern

- Linearità dei pattern
  - Lineari se in ordinamento stretto
  - Non lineari se ordinati su grafo (STAT)
- Unificazione: necessità di pattern-matching con binding di variabile su spazi infiniti
- Occorrenza: possibilità di specificare un lasso di tempo tra due eventi
- Inizio: necessità di considerare il tempo assoluto di inizio di un evento
- Durata: necessità di considerare la durata di un evento
- Online / offline; pattern dinamici; all / one

I pattern tipici

- Esistenza: se evento allora intrusione
- Sequenza: se evento1, evento2, … eventoN allora intrusione
- Ordine parziale: se ordinamento parziale allora intrusione
- Durata: se evento dura per tempo t allora intrusione
- Intervallo: se due eventi separati da tempo t allora intrusione

I metodi tipici

- Regular Expressions
- Grammatiche non contestuali
- Grammatiche con attributi
- STAT (grafo)
- Reti di petri colorate
- L'ultimo modello "riassume" gli altri

Misuse detection: pessima idea?

- I sistemi di misuse detection hanno molti problemi ma ne presentano uno in particolare: la necessità di gestire una knowledge base degli attacchi
- Problemi di aggiornamento e di ingegnerizzazione delle signature (in qualsiasi modo vengano gestite...)
- Problema del polimorfismo negli attacchi: ADMutate, encoding UTF...

Nuove frontiere per gli IDS

- Verranno presentate ora le direzioni di ricerca in cui ci stiamo orientando

1. Utilizzo di tecniche proprie della behavior engineering
2. Utilizzo di algoritmi basati su riconoscitori a rete neurale

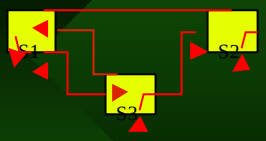
Behavior Engineering

- Campo nato dalle ricerche per lo sviluppo di agenti intelligenti e autonomi
- Tradizionalmente il compito della behavior engineering è studiare i comportamenti per provare a riprodurne versioni “sintetiche”
- Abbiamo proposto l’uso inverso della behavior engineering nella classificazione dei comportamenti

Classificare i comportamenti

- Esistono e sono in corso ricerche per la classificazione dei comportamenti dell’utente di un sito web mediante un modello di Markov, a scopo predittivo
- Catena di Markov (del primo ordine): un insieme di n stati, per ognuno dei quali è definita la probabilità di passaggio a un altro stato all’istante successivo (una matrice n x n di probabilità)
- Modello di Markov di ordine n: dipendente da n istanti precedenti (matematicamente molto più complesso)
- Modello di Markov nascosto: filtrato da una funzione di uscita

Catena di Markov: esempio



	s1	s2	s3
s1	0,6	0	0,4
s2	0,5	0,5	0
s3	0,5	0,2	0,3

- Semplice esempio di catena tra tre stati
- A sinistra il grafico a destra la rappresentazione matriciale

Etologia (?!?)

- Studio e classificazione dei comportamenti degli animali
- FAP: Fixed Action Patterns, sequenza di azioni atomica e non interrompibile
- MAP: Modal Action Pattern, una variante dei FAP proposta da Barlow
- Una intrusione è modellabile con un MAP, a grandi linee
- Il comportamento di un animale può venire modellato tramite un etogramma (elenco di comportamenti tipici) su cui si può costruire un modello di Markov

Modelli e riconoscimento

- Il modello di Markov del comportamento di un animale si modifica sensibilmente se sta svolgendo un MAP o se è in una particolare condizione psicofisica (es. fight or flight)
- Dal comportamento si può (con un test statistico e un determinato indice di confidenza) determinare la probabilità che appartenga a un determinato modello, e quindi che l'animale sia in una determinata condizione
- Modelli di Markov nascosti !!!

IDS Behaviorale

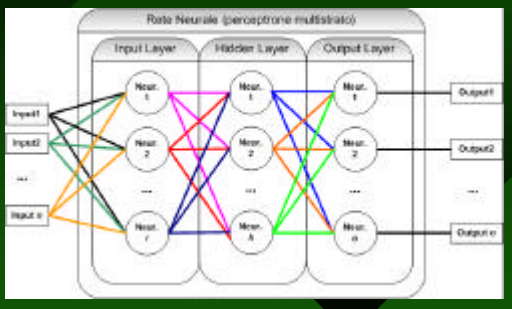
- Dal comportamento degli utenti costruiamo un modello markoviano (di ordine 1 ? Di ordine n ?)
- Cerchiamo di capire se il comportamento "corrisponde" al modello "utente normale", al modello "utente che cerca di fare qualcosa che non va", o al modello "ehi, questo non è un mio utente…"
- Algoritmo di forward-backward, altrimenti noto come "algoritmo di Viterbi": individuazione di modelli di Markov nascosti

Neural Networks Demistified

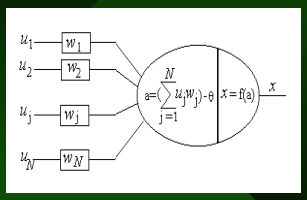
- Cos'è una NN ?
- Un algoritmo capace di:
  - interpolare funzioni
  - classificare oggetti

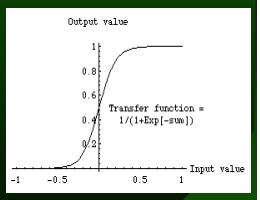
  basandosi su esempi
- Cosa non è una NN?
- Non è magia ma matematica
- Non è intelligenza artificiale
- Non riproduce il cervello o i neuroni

Come è fatta una rete neurale



Neurone Artificiale (zoom)





Il neurone somma gli input, moltiplicati ognuno per i pesi, e fornisce in output un valore filtrato dalla funzione a destra

Come si usa una rete neurale

- Si individua uno spazio di n variabili di ingresso (problema del "mapping")
- Si individuano n variabili di uscita (problema dei threshold)
- Si costruisce una rete (la struttura è un'arte, non c'è un metodo)
- Modelli base: percettrone multistrato, ma anche reti di Hopfield, Recurrent Networks, …
- La rete va addestrata, ovvero vanno selezionati opportunamente i pesi degli ingressi dei neuroni, affinché la rete fornisca l'uscita desiderata

Addestramento

- Calibrazione dei parametri affinché la rete neurale riproduca una particolare funzione tra gli ingressi e le uscite
- Questa funzione può anche non essere nota a priori
- La funzione può essere vista come un classificatore
- Teorema di approssimazione universale: un percettrone multistrato, in dipendenza dal numero dei neuroni, può approssimare qualsiasi funzione per quanto complessa con una precisione arbitraria

Addestramento supervisionato e non

- Supervisionato significa che esiste una conoscenza umana
- Fornisco alla rete neurale esempi di input ed output "corretti"
- Algoritmo di back propagation (numero di esempi almeno pari al doppio del numero dei parametri)
- Non sempre esiste o è formalizzabile una conoscenza umana
- Vorremmo che la rete individui "gruppi interessanti" nel dominio
- È possibile addestrare la rete "online", durante le operazioni; ad ogni modo "addestramento" in questo caso ha un significato molto diverso

Caratteristiche “desiderate”

- Robustezza e generalizzazione: un modello a rete neurale può approssimare il concetto di “simile”, purché vi sia un mapping appropriato
- Adattamento: la precisione della rete può migliorare con l’uso, se i dati vengono riutilizzati per l’addestramento
- Outlyer Detection: individuazione del concetto di “strano” all’interno dei fenomeni

Problemi, problemi…

- Una rete neurale è prona al fenomeno dei “false positives”, ciò nella intrusion detection è inaccettabile
- Una rete neurale non genera conoscenza umanamente comprensibile come risultato del training
- Problema di mapping: quali dati considerare ? In che formato ? Il numero di input di una rete neurale è fisiologicamente limitato

Alcuni spunti per la ricerca

- Supervised training: tentativo di riprodurre mediante una NN un sistema di tipo misuse detection: sembra inefficiente
- Unsupervised training: sembra promettente, tuttavia:
  - problema di definire il concetto di outlyer e delle soglie appropriate.
  - Problema di non reattività
  - Problema principale: mai implementato realmente
- Problemi computazionali per il throughput e per l’addestramento !

Questions ? :)

raistlin@s0ftpj.org