Alma Mater Studiorum · Università di Bologna

facoltà di scienze matematiche, fisiche e naturali
Corso di Laurea Triennale in Informatica

Nuove Tecnologie di Sicurezza: il Trusted Computing

Tesi di Laurea in Sicurezza

Relatore:
Chiar.mo Prof.
Ozalp Babaoglu

Presentata da:
Alessandro Di Teodoro

Sessione I
Anno Accademico 2010/2011

Dedico la tesi:

ai miei genitori, Olga e Roberto,

per come sono riusciti a trasmettermi

la loro lealtà e il loro affetto

alla mia ragazza, Sara,

con la speranza di vivere

sempre al suo fianco

Indice

Elenco Delle Figure

Introduzione

In questi ultimi anni stiamo assistendo ad un grande interesse, quasi maniacale, da parte delle aziende impegnate nell'Information Technology, e altresì degli utenti delle loro creazioni, nell'ambito della sicurezza informatica.
Questa smodata attrazione verso le difese si è resa necessaria in quanto la terza rivoluzione industriale ha creato un nuovo scenario sociale in cui è sempre più raro scovare individui, o aziende, non "informatizzati". Se è vero però che la vita è diventata più comoda, si pensi ad esempio a servizi quali l'online banking o l'e-commerce, è comunque anche vero che in tutte queste attività si è perso il contatto diretto, fondamentale circostanza in grado di garantire un elevato livello di fiducia tra due controparti.
È proprio questa mancanza la causa della ricerca sfrenata della sicurezza nei servizi informatici, una persona non viene più riconosciuta da un suo documento e prende posizione grazie ad una sua firma, ma bensì esegue queste operazioni tramite appositi protocolli studiati specificatamente per la sua attestazione. Questo modus operandi apre le porte ad utenti malevoli, che sfruttando difetti di progettazione, cercano di attaccare i servizi, od i fruitori degli stessi, sperando di appropriarsi di dati sensibili con cui poter recitare la parte di qualcuno sul web. Oltre alla difesa di individui, che utilizzano assiduamente servizi online, la branca che si occupa della sicurezza impiega le sue energie nel contrastare minacce a sistemi che potrebbero portare al malfunzionamento degli stessi: è il caso della

lotta a virus, backdoor, trojan e qualsiasi altro malware. Questo software malevolo è sviluppato da "simpatici" programmatori che hanno come scopo quello di rendere una macchina non funzionante, essi solitamente sono spinti da una sfida personale (contro i colossi dell'informatica) tesa a dimostrare che nessun software può essere impenetrabile.

Terzo ed ultimo motivo per il quale le società dell'IT si spingono nello sviluppo della sicurezza è il lato economico. Più precisamente potremmo parlare della tutela dei diritti di chi vende il proprio materiale in formato digitale (sistemi operativi, musica, etc.). Si rendono necessarie alcune nuove tecniche per lo sbaragliamento di chi cerca di utilizzare materiale "pirata", ovvero software di cui non si possiede la licenza di utilizzo.

Analizzando la situazione si rende evidente come i problemi, a cui può portare un'informatica eretta su basi non sicure, possano recare grattacapi sia ad utenti finali che a società impegnate nello sviluppo, è proprio questa melodia bipartisan che ha orchestrato la produzione di nuove tecnologie in termini di affidabilità.

All'inizio di questa introduzione si è fatto riferimento agli ultimi anni anche se la questione in oggetto esiste da sempre, il motivo è semplice da spiegare. In questa tesi, infatti, non si tratteranno le classiche e conosciute tecniche di sicurezza (a meno di un piccolo excursus) ma ci si focalizzerà sul cosiddetto *Trusted Computing*.

Il TC (acronimo di Trusted Computing appunto) è definibile come *l'insieme delle tecnologie volte a garantire un adeguato grado di affidabilità per tutto ciò che riguarda il mondo dell'Information Technology*, propriamente è l'insieme delle specifiche tecniche che dettano la realizzazione di determinate componenti hardware e determinato software.

Ma da chi sono stilate queste specifiche? Nel 2003 è stata fondata una corporazione con sede a Portland (USA) grazie al contributo di alcune multinazionali impegnate nell'informatica, essa è nata dalle ceneri della vecchia *Trusted Computing Platform Alliance* ereditandone l'atteggiamento e cercando di innovare il metodo operativo: questa società è stata chiamata *Trusted Computing Group*, ed ormai, da otto anni, è il punto di riferimento per la progettazione di sistemi di sicurezza.

Saranno proprio le specifiche di questa azienda, dopo averla introdotta, che andremo ad analizzare di seguito e non mancheranno le riflessioni su queste, in modo da comprendere se la nuova tecnologia è (o comunque sarà) in grado di soddisfare le richieste che l'universo dell'Information Technology reclama a squarciagola.

Capitolo 1

La Sicurezza Informatica

Questo primo capitolo si concentrerà nell'elargire una panoramica generale sulla sicurezza informatica. Per prima cosa dobbiamo definire in maniera non banale ciò che intendiamo quando facciamo riferimento all'affermazione "Sicurezza Informatica", essa non è altro che un ramo dell'informatica che si occupa di rendere sicuri i sistemi, delineando una situazione in cui questi siano immuni agli attacchi di cui possono essere soggetti. È opportuno sottolineare che con sistema non si identificano solo calcolatori che eseguono operazioni (tra le più varie), ma anche le comunicazioni tra loro. Per intenderci possiamo dire che lo scambio di informazioni (anche se non nella stessa intranet) è un "sistema di comunicazione" e come tale deve essere tenuto al sicuro.
Negli anni questo ramo dell'informatica si è profuso nel concederci tecnologie in grado di difenderci dalle minacce nelle quali è possibile imbattersi.

1.1 Crittografia simmetrica ed asimmetrica

La crittografia non è altro che un metodo, tra l'altro utilizzato ben prima dell'invenzione del calcolatore, che si propone di rendere "illeggibile" un messaggio a chi non è autorizzato a conoscere il contenuto di esso.
Iniziamo dicendo, in senso stretto, che con crittografia intendiamo la creazione di cifrari in grado di tenere al sicuro i nostri messaggi, e quindi algoritmi abbastanza complessi che se applicati ad una missiva ed ottenuta la loro risultante crittografata (ciphertext) non permettano di tornare al testo "in chiaro" (plaintext), amenochè non si possegga una ben determinata chiave. Quest'ultima non è altro che una serie di bit che ci permette di applicare la funzione inversa a quella di cifratura per ottenere il plaitext.
Gli algoritmi a cui ci siamo riferiti sopra non sono altro che due funzioni permettenti la creazione, con la giusta chiave, del ciphertext dal testo "in chiaro" o viceversa. Queste due funzioni sono la "funzione di cifratura", che indicheremo con $\mathbf{C_k(m)}$, e la "funzione di decifrazione", che indicheremo con $\mathbf{D_k(m)}$, k e m stanno ad indicare rispettivamente la chiave usata per applicare la funzione e il messaggio che si deve codificare/decodificare, è opportuno però indicare il testo cifrato con **c** in modo da non ammettere ambiguità, la seconda funzione diventerà quindi $\mathbf{D_k(c)}$.

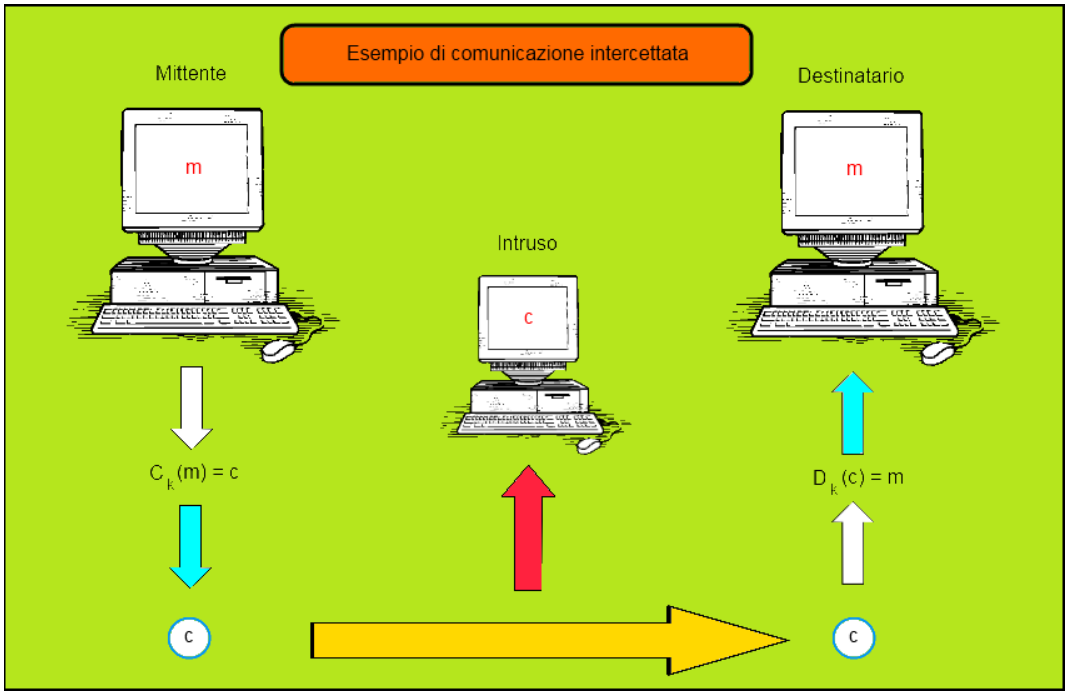


Figura 1: Comunicazione Intercettata

L'immagine (Figura 1) illustra la tipica situazione che si può verificare ogni qual volta si cerca di comunicare tra due calcolatori, come possiamo vedere le frecce azzurre rappresentano l'applicazione delle funzioni sopra citate mentre la gialla rappresenta l'invio dell'informazione. Il mittente del messaggio **m** applica la funzione $\mathbf{C_k(m)}$ ottenendo **c** per poi inviarlo verso il destinatario che applicherà $\mathbf{D_k(c)}$ riottenendo il plaintext.. In un qualsiasi punto della comunicazione può inserirsi un intruso che cerca di rubare i dati (freccia rossa). Come possiamo notare però l'intruso riesce a catturare solo **c**, ovvero il ciphertext, ma non avendo la chiave **k** non riesce commutare il messaggio in **m**, la comunicazione risulta così sicura.
È importante fare una precisazione, e cioè notare che la crittografia si divide in *simmetrica* e *asimmetrica.*

Per crittografia simmetrica intendiamo lo scenario precedentemente descritto, in cui la chiave condivisa (unica) dai due utenti è conosciuta solo da loro, mentre le due funzioni sono a conoscenza di chiunque. La "forza" di questo sistema è infatti

la segretezza della chiave, essa viene usata sia nella "funzione di cifratura" che nella "funzione di decifrazione", si noti che spesso le due funzioni sono le stesse.

La cifratura asimmetrica invece si basa sul fatto che a ogni utente interessato alla comunicazione sono associate due chiavi: *la chiave privata* e *la chiave pubblica*. In questo sistema il mittente di un messaggio deve contattare il destinatario in modo da ottenere la sua chiave pubblica con la quale poi procederà a crittografare il messaggio e infine spedirlo. L'unico modo per ottenere il plaintext a questo punto è decodificarlo applicando la "funzione di decifrazione" utilizzando la chiave privata corrispondente alla chiave pubblica usata dal mittente. Questo sistema rende la comunicazione più sicura, essendo solo il destinatario a conoscenza della chiave e, a meno di sua negligenza, nessun altro potrà mai entrarne in possesso. Riepilogando:

- Il mittente "**A**" richiede la chiave pubblica (**Kpub-B)** del destinatario "**B**";
- "**A**" cripta il messaggio **m** con la funzione $\mathbf{C_{Kpub\text{-}B}(m)}$;
- "**A**" provvede all'invio della risultante della funzione, ovvero **c**;
- "**B**" riceve il ciphertext **c**;
- "**B**" decifra **c** con la funzione $\mathbf{D_{Kpriv\text{-}B}(c)}$ ottenendo il plaintext **m**.

1.2 Algoritmi RSA e DES

Dopo aver chiarito cosa è la crittografia (simmetrica ed asimmetrica) possiamo spingerci oltre, ovvero possiamo descrivere gli algoritmi che vengono usati per applicare le funzioni di cifratura e decifrazione. In questo capitolo approfondiremo la conoscenza dell'algoritmo RSA e dell'algoritmo DES.

L'algoritmo RSA (crittografia asimmetrica), dal nome degli ideatori Ron Rivest, Adi Shamir e Leonard Adleman [1], si propone di riuscire a codificare un messaggio ottenendo una buona robustezza utilizzando funzioni one-way con trapdoor, nient'altro che funzioni di semplice applicazione ma difficili da invertire senza l'utilizzo di informazioni aggiuntive. Cerchiamo di spiegare al meglio come RSA cifri un messaggio, per fare questo però occorre richiamare la *Funzione di Eulero*.

Φ **La funzione di Eulero** (indicata con φ) è definita, per ogni intero positivo n, come il numero degli interi compresi tra 1 e n che sono coprimi con n. [2]

Richiamata questa funzione possiamo passare a spiegare come l'algoritmo di cifratura si applichi ad un messaggio. Come visto in precedenza, se ci troviamo in una situazione nella quale si vuole usare la crittografia asimmetrica sarà indispensabile creare un'accoppiata di chiavi (pubblica e privata) che grazie alle funzioni cifrino e decifrino i messaggi.

Per creare le due chiavi si procede così:

- Si scelgono due numeri primi **p** e **q** molto grandi;
- Si calcola **n** come prodotto dei due numeri primi;
- Si calcola un numero **e** tale che **M.C.D. (e , φ(n)) = 1**;
- Si calcola **d** tale che **d*e mod φ(n) = 1**;
- Si impone l'accoppiata **e** ed **n** come *Chiave Pubblica*;
- Si impone l'accoppiata **d** ed **n** come *Chiave Privata.*

Fatto questo procedimento di "preparazione" è possibile scambiarsi la chiave pubblica e provvedere alla cifratura del messaggio che sarà inviato.
Definiamo ora le due funzioni con le quali si applica l'algoritmo di "mascheramento" dell'informazione.

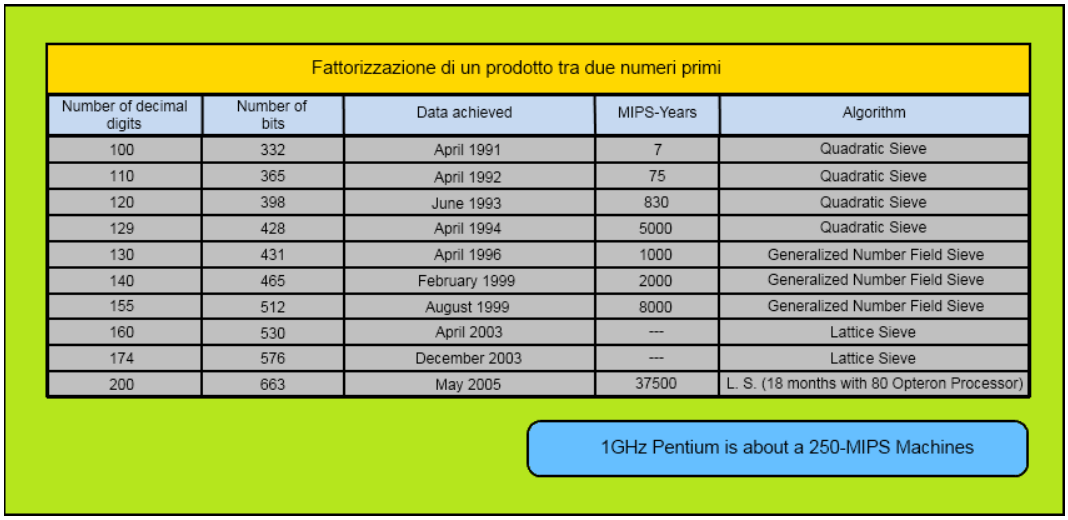
- **Funzione di cifratura:**

$$C(m) = m^e \bmod n$$

- **Funzione di decifrazione:**

$$D(c) = c^d \bmod n$$

Definite queste due funzioni in un calcolatore, sarà semplice per quest'ultimo applicarle nelle comunicazioni tra più utenti che quindi godranno di una sicurezza proporzionale ai numeri **p** e **q** scelti.
Questo algoritmo, attualmente, riesce a garantire una buona robustezza proprio grazie al fatto che i numeri scelti per generare le chiavi sono molto grandi (anche nell'ordine della centinaia di cifre) infatti "[...] la difficoltà di ricavare tale chiave (privata) a partire dalla chiave pubblica coincide con la difficoltà di fattorizzare il prodotto n nelle sue componenti." [1].



Fattorizzazione di un prodotto tra due numeri primi

Number of decimal digits	Number of bits	Data achieved	MIPS-Years	Algorithm
100	332	April 1991	7	Quadratic Sieve
110	365	April 1992	75	Quadratic Sieve
120	398	June 1993	830	Quadratic Sieve
129	428	April 1994	5000	Quadratic Sieve
130	431	April 1996	1000	Generalized Number Field Sieve
140	465	February 1999	2000	Generalized Number Field Sieve
155	512	August 1999	8000	Generalized Number Field Sieve
160	530	April 2003	---	Lattice Sieve
174	576	December 2003	---	Lattice Sieve
200	663	May 2005	37500	L. S. (18 months with 80 Opteron Processor)

1GHz Pentium is about a 250-MIPS Machines

Figura 2: Tempo di calcolo per fattorizzazione di due primi [3]

Sopra a questo testo è riportata una tabella (Figura 2) che rende l'idea di quanto sia complessa l'operazione di fattorizzazione di un prodotto di due numeri primi [3].

Passiamo ora a descrivere un altro algoritmo applicabile nei casi in cui si vuole cifrare un testo, **l'algoritmo DES** (crittografia simmetrica). Questo algoritmo fu sviluppato da IBM ed addirittura adottato come standard nel 1977 dagli Stati Uniti d'America [3]. Prima di spiegare come funzioni DES è opportuno introdurre una particolare funzione detta **funzione di Feistel**.

Questa particolare funzione, sviluppata appunto da Horst Feistel, opera su uno specifico blocco di bit e su una chiave, essa è esprimibile in quattro fasi:

1. *L'Espansione* – si prende il blocco di input e si duplicano alcuni bit fino a che il numero di questi non raggiunge la quantità (sempre di bit) di della chiave;
2. *La Disgiunzione Esclusiva blocco/chiave* – si procede all'operazione XOR tra il blocco espanso e la chiave;
3. *La Scelta* – in questo punto dell'algoritmo i bit vengono divisi in gruppi e successivamente, da ognuno di questi, ne vengono scelti un numero tale che la quantità totale sia pari a quella del blocco originale;
4. *La Permutazione* – nell'ultima operazione della funzione si procede a permutare secondo un preciso ordine i 32 bit risultanti dalle operazioni precedenti.

Avendo esposto Feistel possiamo ora descrivere l'algoritmo DES, si noti che quest'ultimo è utilizzato sia per cifrare che per decifrare un messaggio.
La prima operazione che DES descrive nel suo algoritmo (tra l'altro duale all'ultima operazione) non è altro che una permutazione che va sotto il nome di **IP**, essa non ha relativa importanza nella cifratura ma è una reminiscenza che si porta dietro dagli anni '70, **IP** serviva infatti a facilitare il caricamento dei blocchi. [4]
Dopo aver applicato questa prima funzione si procede a "spezzare" i bit ottenuti in due blocchi uguali (32 bit ciascuno avendo un plaintext di 64 bit), fatto questo si entra in un ciclo di 16 round (ripetizioni) che prende il nome di **rete di Feistel**. Ogni round di questa rete ha il compito di: prendere in input i due blocchi, applicare la funzione di Feistel al secondo blocco, e con questa risultante eseguire la disgiunzione esclusiva con il primo blocco, il risultato di questa operazione sarà il secondo blocco d'input del round successivo che avrà, a sua volta, come primo dato nient'altro che l'input numero due del round precedente. In fine si procederà ad applicare la funzione **FP**, che come abbiamo accennato sopra è l'inversa di **IP**.

Una schematizzazione di DES (img. a) e della catena (img. b) sono riportate in Figura 3.

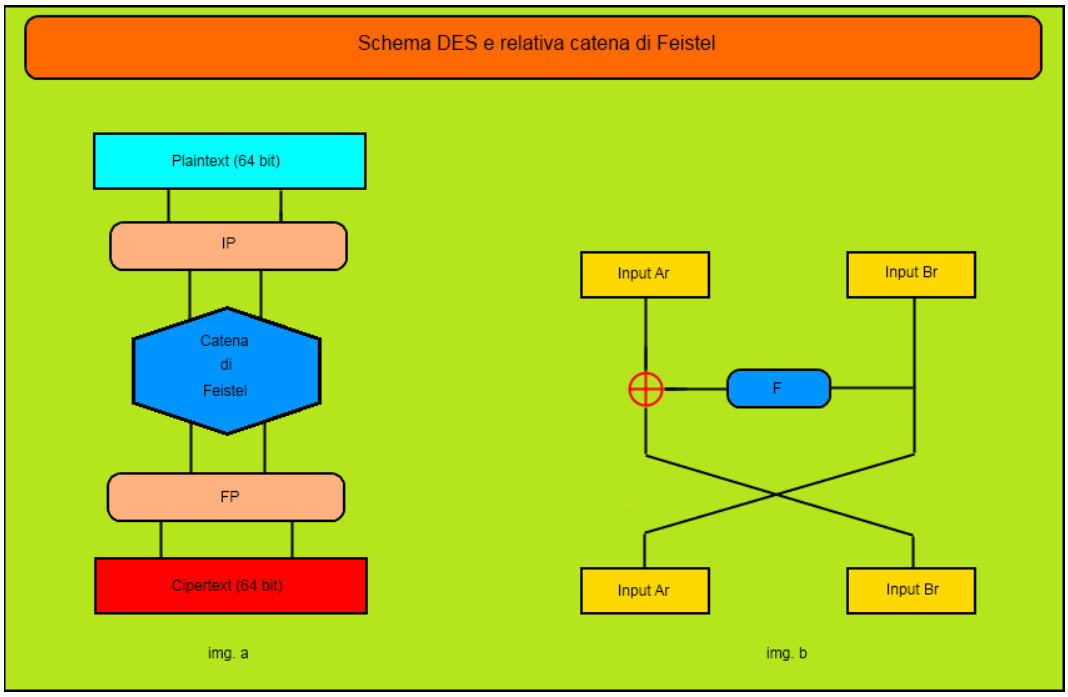


Figura 3: Schematizzazione DES e catena di Feistel

A causa della dimensione della chiave (soli 56 bit), ad oggi, DES non è ritenuto più sicuro ed è ormai stato ritirato dal *National Institute of Standards and Technology* nel 2005 [4] e successivamente nel 2007 si è potuto dimostrare anche "quanto" fosse insicuro, infatti "Una macchina parallela FPGA denominata COPACOBANA dell'università di Bochum e Kiel in Germania viola una codifica DES in 6,4 giorni con un sistema dal costo di $10.000." [4].

1.3 Key Escrow

Il più grande problema che si pone nella crittografia è quello di riuscire a tenere segreta la chiave che permette di trasformare il ciphertext in plaintext o comunque non perderla! Come si può ovviare a questo problema?
Semplice, è possibile ricorrere al *Key Escrow*. Questo metodo consiste nel dividere la chiave in *n* parti per poi inviare tutte le nuove semi-chiavi ottenute a persone fidate o ad *Istituti di Sicurezza*. È anche possibile cifrare queste parti con la propria chiave pubblica in modo che non sia possibile ricombinarle senza possedere la chiave privata corrispondente.
L'algoritmo usato nel *Key Escrow* è di semplice realizzazione ed è conosciuto a tutti visto che la sua robustezza sta esclusivamente nel fatto di riuscire a tenere le chiavi separate. Supponiamo ad esempio di voler dividere una chiave **K** in due parti, non dovremmo fare altro che creare una stringa **R** grande come **K** e poi calcolare lo XOR tra **R** e **K**, la prima semi-chiave non è altro che **R** mentre la seconda è la disgiunzione esclusiva. Per ricostruire la chiave originaria basterà uno XOR tra le due semi-chiavi.
Più genericamente se dobbiamo "splittare" il segreto in *n* semi-chiavi:

- Creeremo *n-1* stringhe di grandezza pari al segreto **K**;
- Daremo ai primi *n-1* ai affidatari le stringhe $\mathbf{R_1R_2...R_{(n-1)}}$;
- Daremo all'ultimo affidatario lo XOR tra tutte le chiavi e il segreto.

Considerando una semi-chiave per ogni affidatario avremo quindi *n* parti. Ovviamente anche in questo caso, per ricostruire il segreto basterà calcolare la disgiunzione esclusiva tra tutte le semi-chiavi.

1.4 La Firma Digitale

"Si può dire che mentre la firma autografa è orientata all'individuo, la firma digitale è orientata al documento" [5]. Questa è, a mio parere, la frase che esprime in maniera più breve e coincisa il concetto di firma digitale. Questa tecnica si propone di far riconoscere univocamente il mittente e allo stesso tempo di far sì che il documento spedito risulti non modificabile, essendo legato strettamente ad una firma derivante da esso.

Per spiegare meglio questo concetto possiamo prendere ad esempio la codifica **PGP** che non è altro che un ennesimo metodo di crittografia asimmetrica, usata però anche per firmare documenti digitali. Per ottenere questo risultato un mittente deve creare una *chiave di sessione* ($\mathbf{K_s}$) che verrà usata solo e unicamente per quella comunicazione. Con essa il mittente procederà a codificare il messaggio (**m)** precedentemente compresso (**m'**) e allo stesso tempo si procurerà la chiave pubblica del destinatario e con quest'ultima cifrerà $\mathbf{K_s}$. Ora il firmatario non dovrà fare altro che inviare al destinatario $\mathbf{C_{K\text{-}sessione}(m')}$ e $\mathbf{C_{kpub\text{-}Destinatario}(K_s)}$, in questa maniera avrà garantito in maniera fidata di essere lui in persona donando al documento la caratteristica di non ripudiabilità.

1.5 Infrastrutture per la Sicurezza

Il problema che sorge a questo punto, visto che sappiamo come comunicare in maniera segreta, è capire come fidarci di colui che sta comunicando con noi. In nostro aiuto intervengono le *Public Key Infrastructure*, ovvero istituti che ci permettono, grazie al loro lavoro, di renderci affidabili agli occhi del "mondo" o informarci sulla trustworthiness di terze persone. Questi istituti si dividono in due rami principali e ovvero: la *Certification Authority* (chi garantisce) e la *Registration Authority* (chi s'informa sulla trustworthiness per concedere i certificati).

I compiti principali di questi istituti sono quelli di creare, validare e revocare la sicurezza, sotto forma di certificati, a chi richiede i loro servigi. Si noti che anche le Certification Authority devono essere certificate da un istituto, si crea quindi una catena di certificazione nella quale una revoca di certificato ad alto livello potrebbe annullare la fiducia ad un numero elevatissimo di utenti.

Capitolo 2

Le Minacce Informatiche

In questo capitolo ci focalizzeremo sulle minacce informatiche. Dobbiamo per prima cosa riuscire a dividere i vari tipi di software malevolo in cui gli utenti possono incappare, questo perché, come il software che utilizziamo, anche le minacce negli ultimi anni si sono sviluppate e si specializzate in nuovi attacchi.
Cerchiamo quindi di classificare questi programmi (o tecniche) definiti minacce. Il primo tipo che incontriamo, considerando anche la linea temporale, è quello che tende a rendere un sistema inutilizzabile andando ad intaccare i file dello stesso. Il secondo tipo di minaccia in cui ci imbattiamo si concentra nel negare uno specifico servizio, mentre il terzo cerca in svariati modi di intercettare comunicazioni o dati sensibili.

2.1 Le Minacce al Sistema

Il cuore di un sistema è il software di base. Questo è in maniera continua esposto a vari tipi di attacco che cercano di renderlo inutilizzabile o di farlo operare in maniera malevola. Tutti questi attacchi vanno sotto il nome di virus, ma cosa è propriamente un virus? Esso è una porzione di codice, solitamente molto piccola, che viene eseguita, all'oscuro dell'utente, su di un sistema e la maggior parte delle volte è proprio lo stesso utente ad averla autorizzata.
La sua storia nasce addirittura nel 1949 quando John Von Neuman dimostrò che era possibile per del codice replicarsi in maniera malevola, ma solo nel 1984 quando Fred Cohen indicò Leonar Adleman come ideatore dell'espressione "virus informatico" fu utilizzato il termine come lo intendiamo oggi [6].
I virus operano su un computer in modo da rendersi invisibili e poter tentare la riproduzione di se stessi, il loro ciclo di vita infatti inizia con l'installazione e fino a che non vengono rimossi cercano all'infinito di replicarsi, "sperando" di riuscire ad accumulare abbastanza potere tanto da non poter essere combattuti. Queste minacce possono essere distinte in innocue e dannose. Sembra quasi un ossimoro dire "minaccia innocua" ma è proprio questa la prima divisione che dobbiamo fare parlando di virus, con questa affermazione si intende classificare tutti quei malware che occupano spazio nel sistema ma, al contrario dei "dannosi", non intaccano file vitali di quest'ultimo. Come abbiamo accennato, i virus dannosi cercano di mettere "KO" un sistema informatico, andiamo a vedere propriamente che cosa si intende per questa affermazione.

I virus più conosciuti negli anni si sono manifestati in questi vari modi:

- Impossibilità di eseguire particolari programmi;
- Impossibilità di accedere a file;
- Rimozione o modifica di file;
- Riavvii imprevisti del sistema;
- Inibizione di software anti-virus;
- Problemi alla connessione di rete;
- Messaggi di errore non attesi;
- Avvio del sistema non possibile.

In questi casi la soluzione più plausibile è quella di riuscire a trovare dei programmi *anti-virus* e *anti-malware* che riescano a contrastare il software malevolo presente nella macchina, purtroppo, alcune volte, se il virus si è espanso abbastanza questa operazione risulta inutile e bisogna procedere alla re-installazione della macchina.
Questo software riesce a installarsi sulle nostre macchine "nascondendosi" dietro ,o fingendosi, qualcos'altro. Basti pensare ai vari *Worm*, che si sono diffusi negli anni passati sulla rete internet, o anche ai *Trojan*, programmi contenenti in essi codice malevolo che se eseguiti corrompono il sistema.
Dovremmo pur difenderci da queste minacce, in nostro aiuto arrivano i programmi di "difesa" dai virus. Questi tendono, in base alla loro tipologia, di prevenire o cercare infezioni nel sistema. La prima tipologia di programma consiste nel monitorare costantemente la macchina e non farla entrare a contatto con software malevolo o semplicemente non attendibile, in modo da ridurre il più possibile il rischio. Il secondo tipo di software invece non fa altro che controllare, con tecniche di *scanning* e *detectioning*, se il sistema è ancora sicuro. Queste due tecniche di rilevamento-infezione usano le firme digitali e le funzioni hash per verificare che sia installato solo software attendibile.

2.2 La Negazione di Servizio

Per far capire cosa è una minaccia che consiste nella negazione di un servizio, ci concentreremo in quelli che vengono chiamati attacchi **Denial of Service** (DoS). Questa tecnica consiste nel richiedere ad una stessa fonte, nel medesimo momento, un identico servizio più volte tanto da farla collassare e metterla in condizione di non riuscire a "rispondere" a tutte le richieste. Come è possibile però attuare questo procedimento? È necessario introdurre il concetto di *Bot-net* esse sono "[...] reti di pc «schiavi» i cui proprietari sono ignari del fatto che svolgono in silenzio vere e proprie attività illegali" [7]. In pratica, grazie a del software malevolo, utenti non autorizzati entrano in controllo di macchine *zombie* che eseguono senza opporsi gli ordini impartiti. Per entrare in controllo di una Bot-net esistono due modi, si può infatti, con l'aiuto di altri utenti, diffondere software che faccia diventare zombi soggetti ignari (questo richiede una buona conoscenza dell'informatica) o si può ricorrere ad una soluzione più "semplice e veloce", esiste la possibilità di acquistare vere e proprie Bot-net dai creatori di esse che le offrono al migliore offerente.
Questi attacchi vengono eseguiti da cracker informatici per arrecare un danno economico a chi deve garantire un servizio, oppure vengono anche utilizzati per protesta contro un'istituzione o una società. Vale la pena citare l'attacco DDoS (**Distribuited Deniel of Service**) che nel febbraio 2011 ha colpito il sito del governo italiano, questo è stato rivendicato da un gruppo di utenti molto vasto che viene chiamato "Anonymous", uomini che dal 2008 si organizzano sulla rete internet per portare a termine le loro proteste.

2.3 Le Minacce ai dati

Da quando esistono gli elaboratori essi sono stati utilizzati, oltre che per il calcolo, anche per il mantenimento di dati sensibili ed è sempre esistito qualcuno intenzionato a "mettere le mani" su quei dati. Negli ultimi tempi si sono sviluppati alcuni software malevoli con lo scopo di entrare in possesso o spiare queste informazioni. Tra queste minacce spiccano le **Backdoor**, gli **Exploit**, lo **Sniffing** e il **Phishing** (non propriamente un software ma una tecnica).

Come possiamo capire dal nome, una Backdoor, viene paragonata ad un porta di servizio e cioè ad una porta che non sia la principale. Si pensi a cosa potrebbe succedere se un abitante di una casa se, oltre alla normale porta di entrata, avesse una seconda entrata e per qualche motivo non ne fosse a conoscenza, un disastro! È proprio questo che succede in un sistema operativo, da una operazione incauta di un utente si "spalancano" accessi non controllabili. Da questi ultimi, dei malintenzionati possono riuscire ad entrare nell'elaboratore e a piacere svolgere azioni contro la volontà dell'utente.

L'Exploit è una tecnica simile alla Backdoor, differisce da questa però perché è un vero e proprio codice che, sfruttando un bug del sistema o una sua vulnerabilità preesistente (non causata da un utente), permette al malintenzionato di impossessarsi del controllo di una macchina.

Con il termine sniffing si identificano le tecniche utilizzate da chi vuole intromettersi frammezzo alle comunicazioni tra macchine. Come sappiamo, per trasmettere dati all'interno di reti, è indispensabile dividere l'informazione in pacchetti, ciò che lo *sniffer* tenta di fare è intercettare questi, per poi tentare di decodificarli e entrare in possesso di informazioni riservate.

L'ultima minaccia, forse la più diffusa, è il Phishing, questa non è nient'altro che la richiesta esplicita di dati sensibili ad un determinato utente, che come un pesce "abbocca" all'amo lanciato dal malintenzionato. Il Phishing è nella maggior parte delle volte diffuso per e-mail nelle quali chi vuole impossessarsi di dati si finge una persona (fisica o giuridica) fidata e induce il finto amico o cliente a divulgargli i sui dati. Possiamo sotto vedere un'immagine che rappresenta uno dei

casi più diffusi (Figura 4). Si nota facilmente che è un caso di phishing visto che non veniamo chiamati con il nostro nome e soprattutto invece di "linkarci" alla home page della ditta in questione veniamo collegati ad una pagina che ci chiede direttamente nome utente e password.

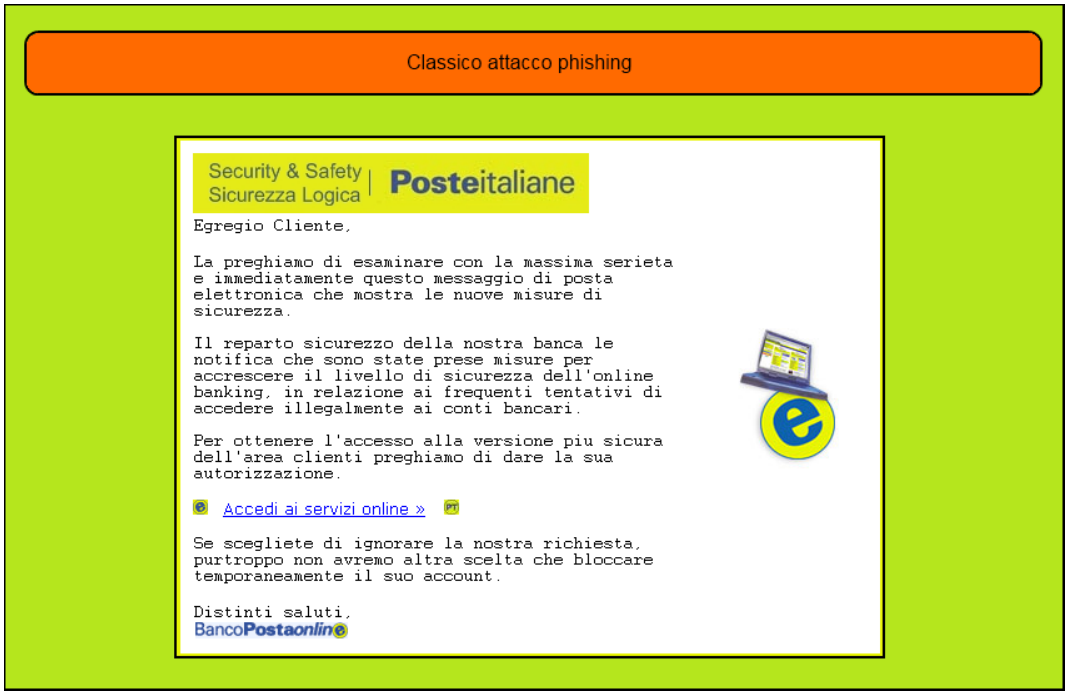


Figura 4: Esempio di Phishing

Capitolo 3

Panoramica sul Trusted Computing

Dopo esserci dilungati sulle minacce che si possono incontrare avendo a disposizione un calcolatore ed aver spiegato cosa significa cifrare le informazioni, possiamo ora dedicarci al Trusted Computing e cioè varie tecniche che permettono all'utente di operare in sicurezza e affidabilità su una macchina. In questo capitolo andremo ad analizzare come è nata questa tecnica e il modo in cui opera concentrandoci sui concetti generali che essa si è proposta nel suo lavoro.

3.1 Nascita

Nel momento in cui utilizziamo un sistema informatico, qualunque esso sia, è necessario munirsi di buoni sistemi di difesa che riescano a bloccare le minacce esistenti al momento, e allo stesso tempo non limitino le nostre operazioni. Questo è proprio quello che si proponevano, e si propongono, aziende produttrici di hardware o software che nel 2003 diedero i natali ad una nuova società chiamata Trusted Computing Group creata per raccogliere le teorie e le tecnologie della Trusted Computing Platform Alliance.

La Trusted Computing Group è una corporazione fondata da *Microsoft*, *Intel*, *AMD*, *Sun Microsystem*, *Hewlett-Packard*, *IBM* e *Sony* con sede a Portland in Oregon, a questa società, successivamente, si sono aggiunte anche altre aziende importanti come Lenovo e Seagate, questo ha dato le basi per una "potenza decisionale" sul mercato mondiale che doveva essere controllata da qualcuno, infatti la TCG si organizzò in modo da essere governata vita natural durante da alcuni membri fissi e altri eletti dagli stessi appartenenti. Fin dall'inizio la società si è concentrata nello sviluppare chip, software e protocolli di rete in grado di lavorare assieme come un normalissimo sistema già esistente, ma che rendesse più sicuro il modo di operare.

3.2 Ambizioni

Come detto precedentemente la Trusted Computing Group si è profusa, in questi anni, nell'ambire a nuove tecnologie nel campo della sicurezza. Questo si è tradotto in vari studi e varie applicazioni tese a far conoscere all'utente, tramite l'uso di software scritto in una determinata maniera tale da "comunicare" con hardware sicuro, l'affidabilità che un sistema può garantirgli.
Queste tecniche creano e monitorano continuamente il "Perimetro di Fiducia" [8], che non è altro che l'insieme di risorse "fidate" che proteggono l'accesso ad alcune zone critiche ed a dati sensibili. Come è possibile fare questo? Semplicemente progettando software ed hardware, collaboranti tra loro, che impediscano operazioni ed accessi subdoli, controllando il modo di operare della macchina in alcuni punti quali: il software di base, il software applicativo, i file, il bios e i driver di periferiche. Qualcuno potrebbe domandarsi per quale motivo devono essere monitorati tutti questi elementi quando è noto che i problemi partono quasi esclusivamente dal sistema operativo, ma rispondere a costui non dovrebbe essere un problema visto che il software di base, oltre ad essere controllato da se stesso, è succube di decisioni che riguardano la sicurezza da parte dell'utente che lo controlla, questi infatti potrebbe dare via libera ad operazioni tutt'altro che sicure. È ovvio quindi immaginare perché la TCG si sia focalizzata su tutti gli apparati del sistema e non solo sul "cuore" di esso. Cerchiamo ora di capire come si fa ad applicare nella realtà questo "fortino inespugnabile" annunciato dalla società. Il TC si propone di riuscire a cifrare e firmare tutte le risorse citate prima, in modo che l'utente, attraverso il software e l'hardware utilizzato, sia sicuro di operare in piena trustworthiness grazie al fatto che il sistema permette l'esecuzione di programmi, e l'uso di periferiche (attraverso i driver), solo se questi sono dotati di un grado di affidabilità sufficiente, riconoscibile appunto grazie alla firma apposta in essi. Per riuscire in questa impresa (ancora in sviluppo) è necessario implementare nuove metodologie da inserire affianco e "internamente" a quelle già esistenti che permettano l'operazione di cifratura e firma anche a basso livello, in modo da

allargare la recinzione di sicurezza alla massima ampiezza che essa può sopportare.

3.3 L'Organizzazione

Come abbiamo potuto capire da come è nata la Trusted Computing Group e da cosa essa si prefigge di riuscire a creare, non è un caso pensare che essa sia divisa in più commissioni, ognuna di queste inerente ad un ramo ben preciso della sicurezza informatica. Le commissioni che si sono create fino ad oggi sono 9, ovvero:

- Organizzazione per lo sviluppo del TPM

  In questo gruppo lo scopo è quello di riuscire a creare specifiche che portino alla costruzione di chip "Trusted Platform Module", integrati alla scheda madre, che possano garantire la sicurezza del sistema;

- Organizzazione per lo sviluppo del TSS

  Nel secondo gruppo, per esteso Trust Software Stack, l'intento è quello di realizzare interfacce di programmazione che permettano alle macchine di usare i moduli fidati specificati dal gruppo TPM;

- Organizzazione per il TNC

  Lo scopo del gruppo Trusted Network Connect è quello di definire nuove architetture di rete che possano rendere sicura la comunicazione;

- Organizzazione per sistemi client

  Come possiamo capire dal nome del gruppo, in esso si prendono decisioni che possano rendere compatibili le nuove tecnologie nei sistemi client;

- Organizzazione per sistemi server

  Simile al gruppo precedente ma, in questo caso, ci si occupa dei sistemi server;

- Organizzazione per sistemi mobili

In questo ennesimo gruppo si cerca di applicare le tecnologie TC nei dispositivi mobili (ad esempio Smartphone);

- Organizzazione per la memorizzazione di dati (storage)

Chi partecipa a questa è chiamato a decidere in che modo si dovranno immagazzinare dati su supporti di memorizzazione in modo che la privacy delle informazioni sia garantita, qui vengono quindi prese anche le decisioni che influenzano la memorizzazione di file su reti o il backup di sistemi;

- Organizzazione per le Infrastrutture Informatiche

In questo gruppo si studia il modo di creare infrastrutture che supportino le tecnologie TC;

- Organizzazione per Hard Copy

La Hard Copy non è altro che la copia in formato fisico (una stampa ad esempio) di documenti digitali, nel gruppo si creano le basi per utilizzare questa tecnologia in modo da essere la più affidabile possibile.

Queste commissioni lavorano in maniera distinta ma si riuniscono in periodicamente per accordarsi su parti comuni del loro lavoro e promulgare gli standard che verranno poi adottati.

3.4 Concetti generali del Trusted Computing

L'ambizione del TC viene tradotta nel concreto con alcuni concetti generali che si propongono di utilizzare la cifratura per rendere il sistema il più sicuro possibile e allo stesso tempo in grado di far fronte alle richieste dell'utente senza limitargli le operazioni eseguibili. I concetti dichiarati dalla Trusted Computing Group sono descritti nei seguenti sotto-paragrafi.

3.4.1 Endorsement Key

Questa tecnologia, abbreviata EK, ha lo scopo di riuscire ad identificare ogni Trust Platform Module, in pratica permette al sistema di riconoscere in maniera univoca qualsiasi chip montato sulla scheda madre e quindi di identificare anche quest'ultima. Le aziende produttrici, al momento della costruzione dell'hardware, imprimono due chiavi RSA a 2048 bit in ogni chip facendo in modo che queste siano impossibili da consultare o rimuovere, sarà quindi improbabile eludere i meccanismi di sicurezza eretti da questo "muro". La coppia di chiavi RSA residente in hardware viene usata dai programmi e dal sistema operativo per identificare il calcolatore sul quale stanno operando, viene quindi garantito al codice di essere su una piattaforma certificata.

3.4.2 Attestation Identity Key

Essa è un ulteriore duplice chiave RSA usata come marcatore di identità per gli utenti che utilizzano un determinato software su piattaforma TPM. Questa coppia di chiavi identifica un utente attivo nel calcolatore come un Universal Unique Identifier identifica un programma avviato da un fruitore in un sistema operativo. A differenza delle Endorsement Key le AIK non hanno una

corrispondeza univoca bidirezionale (un chip una sola chiave, una chiave un solo chip), queste chiavi utente identificano si univocamente un soggetto, ma lo stesso soggetto può avere un'infinità di chiavi di attestazione: per esempio se ne possono creare una coppia ad ogni programma applicativo che viene lanciato.
L'affidabilità di queste chiavi è garantita dal fatto che queste vengono create direttamente dal chip TPM in base all'utenza che ne fa richiesta.

3.4.3 Sealed Storage

Come è possibile apporre un sigillo veramente "infrangibile" su dati sensibili quando esistono macchine dalla potenza talmente elevata che potrebbero "crackare" le chiavi che proteggono essi? Come è possibile credere di essere al sicuro quando esiste anche una sola remota possibilità che qualcuno indovini la chiave? La risposta è il Sealed Storage, questa tecnologia cifra i dati presenti su un dispositivo di archiviazione di massa usando chiavi pubbliche derivanti dalla configurazione del sistema HW/SW in uso in quel momento. Questo rende praticamente impossibile l'accesso ad informazioni se non si possiedono le chiavi private del software, dell'hardware e dell'utente, ciò si traduce in concreto nell'inattuabilità di provvedere alla decodifica di informazioni se non si è localizzati materialmente sulla macchina che mantiene fisicamente i dati. É importante sottolineare che questa politica restrittiva è controllata dal solo dispositivo di memorizzazione così da permettere ad esso di essere decifrato da un range di macchine ben preciso definito a priori.

3.4.4 Memoria Separata

La tecnologia di separazione di memoria deriva dalla già nota protezione di memoria, stiamo quindi parlando di memoria volatile ovvero quella memoria atta a contenere tutto ciò che è in esecuzione in un sistema. Per protezione di memoria intendiamo tutte le metodologie che permettono ai dati, e alle applicazioni, di non creare conflitti tra loro attuando politiche di segmentazione o paginazione. Con l'evoluzione di questa tecnica si cerca di rendere inaccessibili alcune zone di memoria RAM a determinate operazioni da parte di alcuni software che potrebbero agire in maniera non "sicura" su dati. Neanche il sistema operativo o l'utente *root* tramite esso può accedere a queste zone.

3.4.5 Remote Attestation

L'attestazione remota è una funzione che permette ad un sistema di essere riconosciuto sulla rete nella sua configurazione. Questa procedura non fa altro che interrogare il nostro HW/SW per capire se esso è stato modificato. La tecnica della Remote Attestation è resa in atto grazie al TPM, infatti il chip in cui sono residenti le chiavi RSA certifica il sistema ad un certo tempo, come se ne facesse una fotografia, da questo punto in poi ogni applicazione, che richieda affidabilità sulla rete, "fotograferà" anch'essa la configurazione HW/SW per poi lavorare insieme al chip della macchina in una procedura di matching tra le due istantanee. Un esempio che rende l'idea dell'attestazione remota potrebbe essere un software che si connette ad un catalogo multimediale on-line dal quale possiamo effettuare acquisti di qualsiasi genere. Il gestore del server potrebbe volere che il suo materiale venga in contatto solo ed esclusivamente con software originale e particolari configurazioni hardware (come quella standard di base). Per fare ciò il software residente nel server controlla il sistema attraverso la RA, e solo se questa restituisce una risposta di confronto positiva al controllo si può avviare la comunicazione vera e propria.

3.4.6 Protected Input/Output

La protezione di I/O si focalizza nel cifrare tutti i flussi che sono generati dalle richieste di programmi o tutti gli output che gli stessi generano quando vengono interrogati (o vengono richieste informazioni). Questa tecnologia mantiene la cifratura del flusso fino a che questo non raggiunge la sua destinazione, possiamo pensare ad un semplice "click" del mouse, questo verrà cifrato appena sarà immesso nel sistema e decifrato non appena raggiungerà la sua meta. Per applicare questa tecnica però è necessario possedere hardware (come tastiere, mouse, schede video, etc.) che supporti la cifratura e la decifrazione di flussi di dati inviati e/o ricevuti. Questa tecnologia è tesa a proteggere i dati da utenti, o software malevoli, che cercano di intercettare e copiare parti di essi.

Capitolo 4

Presente e Futuro del TC

Essendo il Trusted Computing una tecnologia ancora in sviluppo possiamo facilmente intuire che ci siano tecnologie che si sono di già "presentate al mondo" e altre che invece che sono ancora sotto studio e quindi non ancora stabili. Andremo in questo capitolo a descrivere queste tecnologie.

4.1 Applicazioni attuali del TC

4.1.1 Il Trust Platform Module

Il Trust Platform Module, sviluppato dall'omonimo workgroup del TC, non è altro che un microchip che fornisce le basi per l'applicazione delle politiche trustworthiness nei sistemi che utilizzano questa tecnologia. Esso è il modulo che permette di cifrare (e eventualmente firmare) quello che le procedure del TC richiedono di codificare e rendere riconoscibile a terzi sia nel disco che nella rete. Questa operazione di codifica dati, chiamata on-the-fly encription, è attualmente utilizzata da software, per quando riguarda il disco, quali TrueCrypt o FreeOTFE. Cerchiamo di capire come funzionano per farci un'idea di come il TPM agisca sul sitema. Con il software TrueCrypt abbiamo la possibilità di criptare intere porzioni di Hard-Disk o addirittura partizioni bi boot, così da proteggerle in caso che un programma, o un utente malevolo, non possa modificarle e rendere il calcolatore non più fidato o addirittura non funzionante. Questo è in pratica quello che fa il TPM, si noti che a differenza del software, il chip, "lavorando" a basso livello rende le operazioni molto più veloci.
Come detto in precedenza con le chiavi pubbliche e private è possibile firmare messaggi. La seconda funzionalità di questo microchip è proprio questa, infatti, tramite la coppia di chiavi RSA residenti in esso si può certificare il sistema al tempo *t*, ciò rende possibile l'implementazione della Digital Certification. Come è possibile tramite le chiavi applicare la DC? Introduciamo il concetto di Hash, una funzione che fornisce una stringa di bit partendo da un file, anche se questo è funzionale al nostro sistema, riuscendo così a creare la famosa istantanea del sistema. È garantita così l'affidabilità di questo, ad esempio se un programma modificasse un file vitale, la tecnologia Digital Certification riconoscerebbe subito la situazione e bloccherebbe tutte le operazioni protette da essa.
La variante di questa seconda tecnica consiste nel firmare, al posto di cifrare, i file del sistema, ciò serve ad attestare la propria identità in una rete o meglio

l'identità della macchina. Tutto questo è strettamente legato al TPM e sostituendo esso, infatti, dovremmo firmare di nuovo ciò che ci identifica, l'operazione comporterà una nuova firma e quindi non appariremo più con la stessa identità, potrebbero attestarci come utenti non autorizzati nell'operare in una rete.

Andiamo ad esaminare l'architettura che permette il funzionamento, cioè l'applicazione dei vari algoritmi, del Trust Platform Module. Potremmo schematizzare il TPM come una serie di componenti (Figura 5).

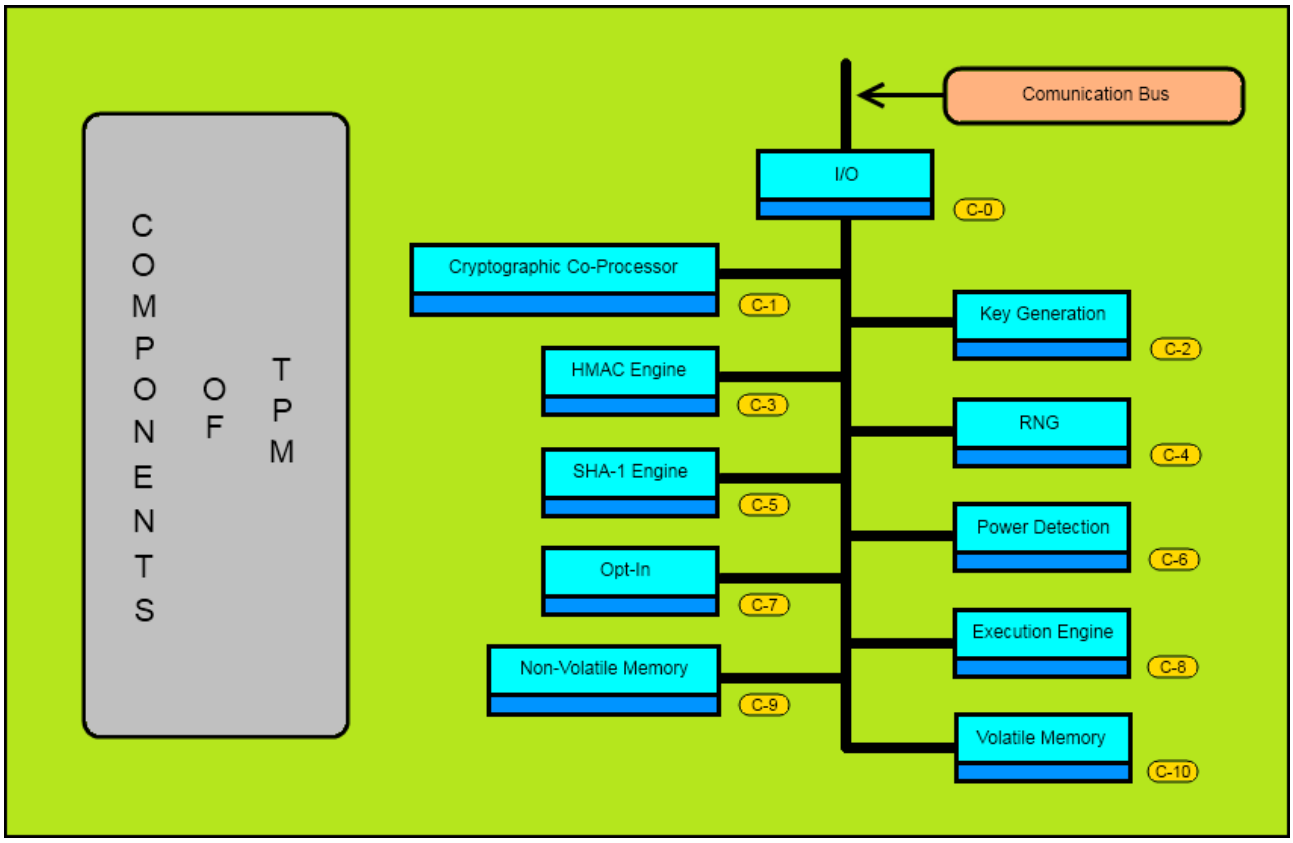


Figura 5: Componenti del TPM [9]

Si nota dall'immagine come il TPM sia diviso in undici componenti, con ognuna un ben preciso compito da svolgere, queste vengono identificate dalla lettera C con una costante a seguito (si parte dallo zero per la numerazione). Il primo

livello che incontriamo in questa architettura è l'**Input-Output**, una componente che si occupa del protocollo di codifica, permettendo di comunicare anche con bus esterni e in secondo luogo imporre le politiche d'accesso (del componente Opt-In) alle funzioni a cui viene passato il controllo. Seconda componente del TPM è il **Co-Processore Crittografico** che implementa le funzioni di crittografia asimmetrica del chip stesso e altre funzioni come SHA-1, generalmente questo segmento esegue i sui calcoli con chiavi a 2048 bit e quindi permette la generazione, in uscita, di un solo blocco, tuttavia nel caso in cui questo non fosse possibile (applicazione di algoritmi diversi) e si generassero più blocchi, questi, grazie a meccanismi di concatenazione gestiti dal TPM, riuscirebbero comunque a mantenere la propria integrità. Il Co-Processore Crittografico svolge tre diversi compiti, infatti è possibile notarlo operare sia come motore degli algoritmi RSA sia come motore degli algoritmi simmetrici e infine come il gestore che svolge operazioni di firma. Seguono poi quattro livelli che permettono la gestione degli algoritmi necessari al TPM e sono i generatori di **Chiavi** e **Numeri Random** e i motori di **SHA-**1 e **HMAC**, cioè la componente che permette di autenticare i messaggi sul bus. Ennesimo componente è il **Power Detection** che si limita ad informare il TPM dei cambi di stato di alimentazione che subisce il sistema, con questa operazione, in maniera indiretta, è possibile riconoscere i momenti nei quali si rende necessaria l'applicazione di alcuni algoritmi. Come accennato prima, il componente **Opt-In** è quello che gestisce le varie politiche del chip (anche la sua disabilitazione), ciò è permesso dal fatto che questo salva dentro di se stati di varie flag (persistenti e volatili) in grado di informare il sistema sulla configurazione-macchina e sulle operazioni da attuare. Dopo aver esaminato le flag è possibile eseguire i comandi richiesti dal TPM, entra in gioco il componente C8 chiamato **Execution Engine** che non è altro il cuore del Modulo in grado di eseguire il codice. Ultimi due componenti che possiamo distinguere sono quelli che riguardano la memorizzazione **Non-Volatile** e **Volatile**, come possiamo immaginare nella parte di memoria persistente sono salvate tutte le informazioni che riguardano lo stato fissato del TPM, mentre invece la "volatile" serve a eseguire i calcoli di cui il chip necessita. Questi calcoli vengono svolti in dei registri chiamati PCR di dimensioni pari a

160 bit che appaiono nel Trusted Platform Module in numero mai inferiore a sedici.

4.1.2 Il BitLocker Drive Encryption

La tecnica BitLocker è stata introdotta per la prima volta nel sistema operativo *Windows Vista* [10]. Con essa Microsoft si è proposta di riuscire a proteggere al meglio i dati degli utenti del loro sistema tramite un particolare protocollo di cifratura, questo rende arduo il furto dell'informazione anche se si possiede direttamente il supporto che la contiene. La procedura si basa su due punti focali e cioè, la cifratura del volume in cui è contenuto il sistema operativo e la verifica sia dell'integrità delle componenti sia delle configurazioni di avvio. La tecnica BitLocker è stata progettata per utilizzare il TMP (versione 1.2) come hardware di cifratura, ma le macchine che non sono dotate di questo chip possono comunque usare tale tecnologia tramite un supporto di memoria (per esempio una chiave USB), questa scelta è comunque considerata "scomoda" visto che non è spalleggiata da un dispositivo di cifratura preposto ed è legata, in modo indivisibile, all'utilizzo di sistemi che possano caricare supporti di memoria removibili in ambiente pre-OS.

I modi in cui BitLocker opera sono principalmente tre, e ovvero:

- Controllo dell'integrità dei file di avvio, in modo che si realizzi se questi siano stati modificati dannosamente;
- Aumento della protezione, cifrando i file che permettono l'avvio del sistema e quindi "impossibilità" del controllo di questo da non autorizzati;
- Blocco del sistema, in caso di alterazione di esso.

È possibile utilizzare la modalità di avvio protetta in modi diversi che mutano al variare del nostro hardware e delle impostazioni di sicurezza. Possiamo distinguere l'applicazione di BitLocker con solo TPM oppure con l'ausilio di PIN, o dispositivi di archiviazione removibili.
In un sistema provvisto esclusivamente di Trusted Platform Module, il disco viene partizionato in due volumi (Volume di Sistema e Volume di Sistema Operativo) e per decifrare i dati c'è bisogno di usare due chiavi, infatti il TPM cifra la Chiave Master (intero volume) che a sua volta cifra un'ulteriore Chiave di volume in grado di cifrare e decodificare i dati sensibili.
Nello scenario in cui vengono usati codici quali un PIN, o supporti removibili quali chiavi USB, le modalità di cifratura non vengono alterate rispetto alla situazione precedente, viene infatti immessa solo una modifica, e cioè l'autorizzazione al TPM di operare come se fosse "solo" nel sistema. Sono proprio questi supporti che danno "via libera" alle funzionalità del chip, questa modalità operativa si applica calcolando lo Hash SHA-256, da questo punto vengono "presi" i primi 160 bit della risultante che verranno usati come dati di autorizzazione del TPM.
In assenza di TPM il sistema operativo, tramite uno script, crea e cifra la Chiave di Avvio memorizzandola su un supporto, in assenza di esso sarà impossibile accendere la macchina per accedere al sistema.
È di vitale importanza per il sistema creare la Chiave e la Password di Ripristino, questi due elementi servono, come suggerisce il loro nome, a "ripristinare" il sistema in caso di aggiornamento di parti Hardware o Software che cambino lo stato del sistema. Gli scenari di ripristino sono molteplici e vanno dal semplice aggiornamento del BIOS alla disinstallazione del TPM dal sistema, o addirittura alla perdita delle chiavi (PIN e supporti). È opportuno sottolineare che nelle operazioni di ripristino non è coinvolto il TPM, quindi in caso di rottura di scheda madre, in cui sono integrati questi particolari chip, non si dovrà sostituire tutta la macchina ma basterà avviare un'operazione rigeneratrice che sistemi il tutto (dopo aver sostituito la parte non funzionante ovviamente).
L'introduzione di BitLocker nel sistema Vista, e Seven poi, ha movimentato gli utenti e creato richieste tese a cercare un modo per amministrare in maniera

semplice questo nuovo tipo di sicurezza, nel febbraio 2009 Microsoft ha annunciato "Microsoft BitLocker Administration and Monitoring", un tool che permette la configurazione di BitLocker in maniera più *user friendly* possibile [11].

4.1.3 Trust Zone

La TrustZone è una tecnologia che da specifica è implementata direttamente dal microprocessore stesso, essa è stata introdotta dalla società britannica "ARM Holdings". Questa modalità di sicurezza è stata pensata ed ideata per piccoli sistemi e promette di essere in grado di far fronte agli attacchi diretti ad essi e, soprattutto, di riuscire a rendere più sicure le operazioni di commercio in rete. Questa tecnologia infatti riesce a distinguere dati e codici sicuri da quelli malevoli, è necessario che non sia attivo lo standard TPM del Trusted Computing, infatti la TrustZone è una sorta di TC proprietario implementato dalla ARM. L'API software della TZ riesce ad abbattere i costi per la sicurezza in sistemi informatici, essa infatti è in grado di implementare tecniche per il Digital Rights Management.
La TrustZone è stata introdotta appunto per far fronte alla sempre più innovativa fabbricazione di dispositivi digitali, per lo più mobili, e all'ideazione di nuovi metodi (e software) di comunicazione in rete, dichiarava infatti John Bruggeman: "Poiché l'emergere di dispositivi di nuova generazione continua a suscitare l'immaginazione del mercato con particolari combinazioni di applicazioni e servizi, la sicurezza si sta affermando come una necessità critica per i nostri clienti". perché proprio Bruggerman, capo marketing di Wind River, fece questa dichiarazione? Semplicemente perché proprio la società a cui appartiene è stata la prima che ha deciso di integrare la tecnologia TrustZone.
Cerchiamo di spiegare come ARM ha implementato la TrustZone. La società britannica ha diviso, secondo la sua concezione, software e hardware (SoC) in due parti separate: il "Mondo Normale" e il "Mondo Sicuro" (Figura 6) e proprio sulla

distinzione dei due "mondi" si basano i concetti della TrustZone. Il "Sicuro" non dovrà mai entrare in contatto con il "Normale", ciò è garantito dall'hardware residente nel bus *TrustZone-enabled AMBA3 AXI™* con il quale si riesce a creare software robusto (da eseguire nel processore sicuro) in grado di proteggere il sistema dagli attacchi informatici. L'idea vincente di questa procedura è stata estendere alcuni processori ARM tanto da renderli in grado di gestire software di entrambi i "mondi" senza riservare core per la sicurezza, è stato quindi possibile implementare la TrustZone anche in processori mono-core. Si creano in pratica due processori virtuali che "switchano" tra il "Normale" e il "Sicuro" attraverso un monitor che controlla cosa eseguire e dove (virtualmente parlando) in un determinato istante di tempo. Ovviamente passare dal "Mondo Sicuro" a quello "Normale" non comporta rischi, ma al contrario il passaggio opposto potrebbe innescare una serie di eventi funesti e incontrollabili. Qui entrano in gioco le istruzioni del Secure Monitor Call, una procedura di accesso al monitor che controlla se si hanno i giusti privilegi e se non si è intenzionati ad azioni malevole (come potrebbe esserlo un malware).

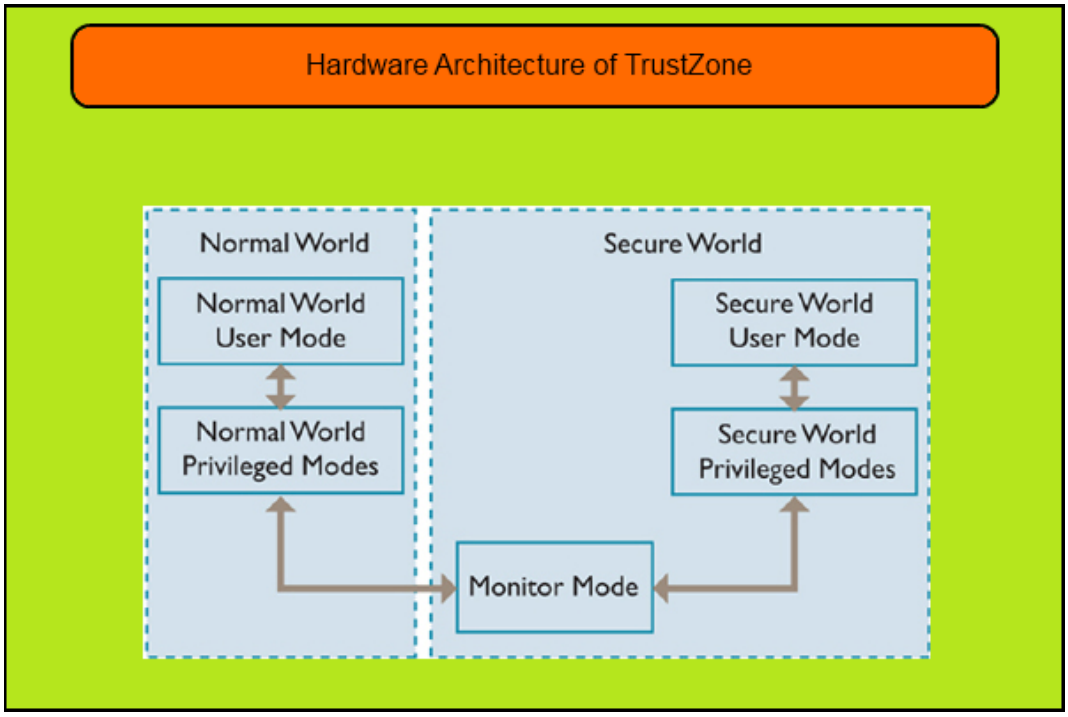


Figura 6: Schematizzazione Hardware TrustZone [12]

Abbiamo visto come è necessario costruire l'hardware per la TZ, è ragionevole pensare che si debba implementare anche un certo tipo di software in grado di essere compatibile questa tecnologia (Figura 7) [12]. È possibile introdurre diverse implementazioni di software abbastanza robusto da collaborare con la TZ che vanno dal mantenere tutto il sistema operativo nel "Sicuro" ad usare librerie sincrone residenti sempre in questo. Sicuramente la prima delle due opzioni è l'implementazione più potente da immaginare, visto che mantenere tutto nel "mondo" più blindato ci permette di eseguire più operazioni allo stesso tempo senza correre alcun tipo di rischio. Ovviamente questa soluzione è la più complessa da implementare e le sole forze di ARM non potrebbero bastare, nel 3 febbraio 2009 infatti la *Giesecke & Devrient*, una compagnia tedesca, ha dichiarato di voler lavorare affianco alla società britannica per la costruzione di dispositivi mobili ad alta sicurezza, in grado di supportare la prima possibilità elencata.

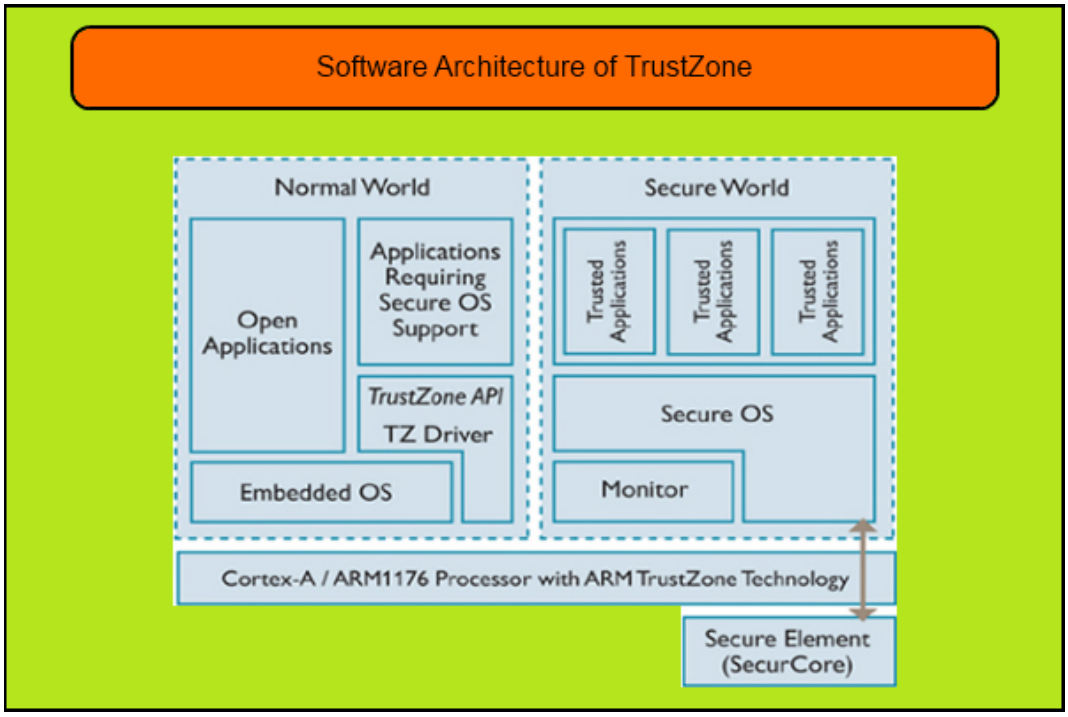


Figura 7: Schematizzazione Software TrustZone [12]

4.1.4 Padlock – Via Technology

Il sistema di sicurezza Padlock implementa gli algoritmi di cifratura in hardware insieme ad alcune funzioni extra che i progettisti del sistema possono usare per rendere i dati e la macchina irraggiungibili da terzi. Tutto ciò crea un grande vantaggio a livello di risorse, infatti utilizzare l'hardware al posto del software alleggerisce e rende più veloci le operazioni per il sistema. In secondo piano possiamo dire che utilizzare il livello hardware ci permette di avere un grado di sicurezza maggiore, visto che un utente malevolo non può, in questo caso, studiare un software di protezione, per poterlo poi "bucare", non essendo esso presente.

Le innovazioni che ha portato con se Padlock sono stare rese possibili da alcune caratteristiche di base che ne permettono il funzionamento, andiamole a descrivere.

L'uso degli algoritmi **SHA-1** e **SHA-256** in modo che i dati sensibili o funzionali al sistema non possano essere manomessi, o intercettati, in scenari di comunicazione. Il calcolo delle risultanti di questi algoritmi richiede potenza di calcolo elevata ma essi ci ripagano dandoci una sicurezza adeguata al dispendio di energie, SHA-256 è considerato in pratica indecifrabile.
L'uso dell'algoritmo **AES**, in grado di cifrare informazioni per 25gb/s, per comunicazioni in canali non protetti.
Molto importante anche la tecnica che permette di "aiutare" i cifrari visti in precedenza che va sotto il nome di **Montgomery Multiplier**, essa si basa sull'algoritmo RSA a chiave pubblica/privata.
Si aggiunge ai cifrari descritti sopra la **NX Execute Protection**, in pratica una "difesa" contro i worm. Questi ultimi non sono altro che del codice che cerca di riprodursi all'infinito, all'immisione, questi worm, sono visti dal sistema come dati, ma riproducendosi essi tentano di entrare nella zona di esecuzione del software "sperando" di essere loro stessi ad essere eseguiti. Con la NX EP non si fa altro che separare virtualmente le zone di memoria volatile, in modo che nessuna serie di bit considerata come dato possa essere eseguita, a meno che non sia stato controllata e dichiarata affidabile dallo stesso sistema.
In ultimo, nei processori ARM, troviamo un'unità chiamata **Via PadLock RNG**, questa provvede a generare numeri casuali ad una velocità di anche 12 milioni al secondo per poi usarli negli algoritmi di cifratura.
Attualmente PadLock è utilizzato dalle famiglie di processori Via C7 e VIA Eden.

4.1.5 LaGrande – Intel Vs. Presidio - AMD

La Trusted Execution Technology di Intel, conosciuta precedentemente con il nome in codice LaGrande, è un'ulteriore tecnologia di sicurezza realizzata in hardware che propone l'affidabilità del sistema senza intaccare la sua velocità e stabilità. La TXT, tra l'altro come ogni altra procedura di sicurezza via hardware, è resa in atto dall'integrazione di estensioni per i processori. Intel riesce con essa

a garantire principi fondamentali del TC quali la cifratura degli elementi sensibili con il Sealed Storage, il Memory Curtaining (Separazione di Memoria) e la possibilità di riconoscere se TXT è attiva e configurata al meglio. Questo sistema di sicurezza può quindi garantirci la verifica locale, quella remota e il **Multi-Level Operation**. Quest'ultimo non è altro che la capacità del sistema, utilizzando il memory curtaining, di riuscire ad utilizzare più software nello stesso istante, garantendo che questi non vadano in conflitto o restino separati nel loro ciclo di vita.
A "LaGrande" si contrappone Secure Execution Mode, nota come "Presidio", di AMD. SEM è una tipologia di sicurezza ibrida, ovvero è implementata via software e hardware, che viene eseguita ad un livello di privilegio più basso del Sistema Operativo stesso in grado di proteggere gli accessi in memoria di operazioni non lecite che corromperebbero il sistema [13].

4.1.6 Trusted Software Stack jTSS

Come accennato nel capitolo 3 esiste una organizzazione della TCG denominata Trusted Computer Stack, che ha l'intento di realizzare specifiche interfacce in grado di consentire l'interoperabilità su diversi sistemi e allo stesso tempo implementare un buon grado di sicurezza.
Queste interfacce sono indispensabili per sfruttare quello che il TPM ha da offrirci, infatti in ogni sistema che usa il chip "fidato" è necessario poter accedere ad esso per vie sicure utilizzando le API che ci vengono fornite proprio dal TSS. Questo Software Stack è formato da più moduli che, lavorando assieme, garantiscono che la nostra persona sia messa in sicurezza di fronte alle minacce, gestisce quindi le politiche di privacy e quelle di interfaccia (comunicazione con il TPM). Tra i moduli presenti nel TSS ne spiccano soprattutto due e sono: la **Trusted Gui** e il **Compartment Manager**.
Come possiamo immaginare il primo dei due moduli riguarda l'interfaccia grafica, esso infatti viene inizializzato ogni qual volta l'utente necessita di inserire input o

anche quando è interessato a controllare gli output del software che sta eseguendo. In pratica la Trusted Gui gestisce la protezione dei dati da parte delle operazioni di I/O che potrebbero corromperli in maniera irreversibile. Una Trusted Gui è stata implementata dallo *European Multilaterally Secure Computing Base* nel progetto *Turaya* definito come segue: "Turaya is an implementation of the EMSCB security architecture, providing strong isolation and multilaterally secure policy enforcement of legacy applications." [14]. In questo modello di architettura la Trusted Gui è usata in due principali modi: autenticazione *just-in-time* (quindi richiesta di una password quando dati sensibili entrano nel dominio dell'utente) e *amministrazione centralizzata* (nient'altro che la possibilità data agli amministratori di rendere accessibili delle particolari risorse).

Il modulo *Compartment Manager* invece si occupa di "chiudere" parti del sistema ai fruenti dello stesso, ciò non fa altro che assegnare ad ogni utente il set minimo di privilegi che garantiscono l'esecuzione normale di un'operazione. Queste operazioni di controllo privilegio non entrano in atto solamente quando l'utente decide di avviare un software, ma ogni volta che viene creata un'istanza di avvio per qualsiasi programma del sistema, è necessario quindi che il modulo sia in grado di eseguirsi anche senza che l'utente richieda (indirettamente) assistenza. Il modo più semplice di creare le istruzioni per il Compartment Manager è quello di suddividere (in maniera variabile) il sistema in "compartimenti" logici così da riuscire a controllare gli accessi ad essi in base ai livelli di privilegio.

Visti i due moduli di "spicco" torniamo a parlare del TSS vero e proprio aiutandoci con un esempio e ovvero il Trust Software Stack implementato nel linguaggio Java, meglio conosciuto con il nome di *jTSS IAIK*.

Come detto IAIK è un implementazione del TSS per JAVA, esso è stato sviluppato ed è mantenuto dalla *Graz University of Technology* con l'appoggio della commissione europea in materia del progetto openTC. Questo progetto differisce dai suoi simili perché si è proposto l'intento di implementare tutti i livelli che richiede direttamente in JAVA.

Attualmente IAIK supporta tre diverse funzionalità:

- La TSS Device Driver Library
  Questo è ciò che ammette il supporto a Linux e Windows permettendo l'accesso rispettivamente al TMP device file e al TBS Microsoft;
- La TSS Core Services
  La seconda funzionalità invece permette di "coprire" le funzioni e le strutture da specifica 1.2 del TPM, inoltre supporta, tramite le funzioni Authorization Manager e Key Cache Manager, il riconoscimento del privilegio utente;
- La TSS Service Provider
  In questa specifica il Trusted Software Stack implementa l'interfaccia SOAP per la comunicazione a oggetti tra software inoltre a gestire le chiavi RSA utilizzate dal sistema e le funzioni hash. Degna di nota, nel TSS Service Provider, è l'implementazione delle funzioni per il Key Migration, ovvero ciò che permette di far fronte alle "rotture" del TPM permettendo il ripristino delle operazioni anche su un chip diverso.

Come possiamo notare il primo punto non è una vera e propria funzionalità ma semplicemente una specifica che permette di far funzionare lo jTSS di IAIK sul sistema corrispondente. I secondi due punti invece (TCS e TSP) permettono il vero e proprio utilizzo del Trust Software Stack in questione, in pratica il Service Provider permette, tramite un API, la richiesta delle funzionalità del chip, ovviamente per accedere al TPM è necessaria una ben definita procedura che non è altro che la Core Service. Possiamo immaginare questa come un demone in continua esecuzione che, a tempo debito, soddisfa le richieste d'accesso al TPM avendo solo lei i privilegi per farlo. Per supportare questa tecnica sono richieste delle parti comuni tra il TSP e il TCS che non sono altro che le strutture e le costanti per il Trust Platform Module, l'attuazione della memoria persistente e alcuni blocchi ausiliari per la cifratura e decifrazione. Abbiamo detto che è

importante che questi moduli comunichino tra loro , ma come è possibile fare questo in IAIK? Si usa il web service SOAP (Simple Object Access Protocol), un protocollo basato su XML, ma molto più "potente" visto che fornisce funzionalità aggiuntive rispetto alle classiche *remote procedure call*, in questa situazione il demone Core Service è visto come un server web (per esempio in Windows si configura, come da specifica del Trusted Computing Group per TSS, la porta TCP all'indirizzo 30003 ma in IAIK viene usata la "successiva" 30004 per evitare collisioni che farebbero venir meno le funzionalità del sistema).

4.1.7 TrustedGRUB

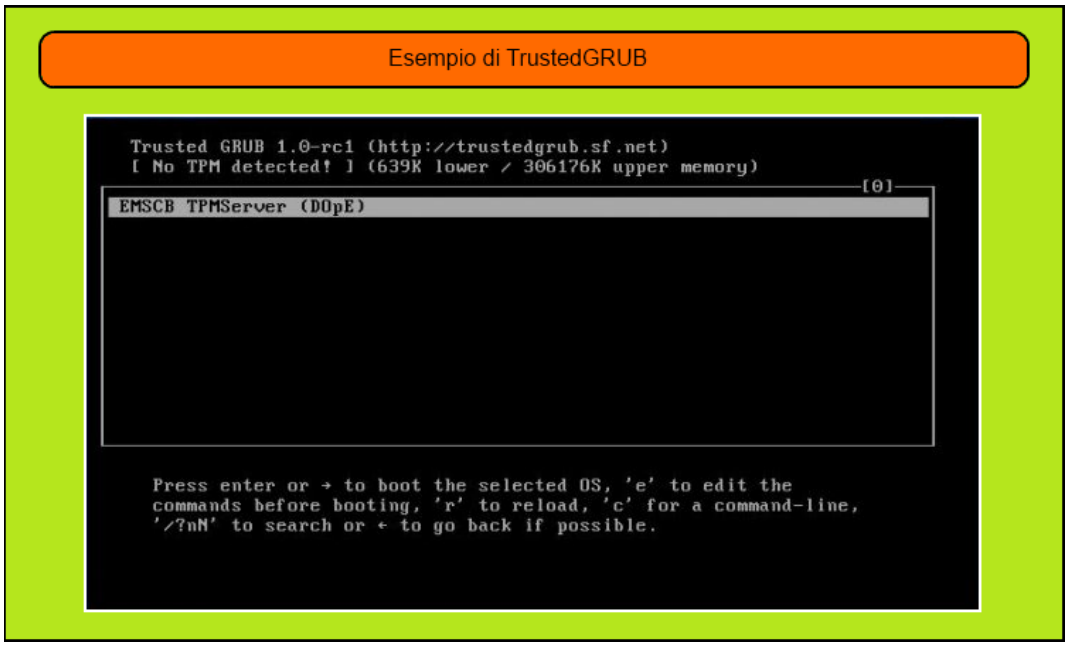


Figura 8: Schermata di TrustedGRUB

L'immagine sovrastante ci mostra una ben nota situazione che si crea ogni volta nella quale si accende una macchina linux, il primo programma a caricarsi in fase pre-boot dopo il Bios è proprio GRUB, acronimo di Grand Unified Bootloader. Questo tool ci permette di caricare il kernel del sistema operativo e avviare così la

macchina.

Come possiamo notare però nella Figura 8 siamo in presenza di una variante di questo programma chiamata TrustedGRUB, non è altro che un modo più sicuro per "fare boot" del sistema. In pratica se ci troviamo davanti a questa funzionalità sappiamo di essere collegati al Trusted Platform Module e questo ci permette di controllare la configurazione dei moduli da caricare, questa operazione è accordata tramite il calcolo delle funzioni hash (SHA-1) dei moduli e la loro archiviazione nel Platform Configuration Registers, una zona sicura del TPM per lo store di dati. Questi, una volta salvati, possono essere utilizzati per confrontare le configurazioni del sistema e quindi riuscire a capire in fase pre-boot se andremo ad operare in maniera fidata.

Il TrustedGRUB può sembrare un controllo "inutile" visto che esso risiede nello MBR, quindi in una zona disco già sottoposta a sicurezza grazie alle specifiche del TC quali la separazione di memoria, ma così non è. Infatti se venisse a mancare questa forma "modificata" di Grub si verrebbe a spezzare la cosiddetta "Chain of Trust", cioè quel collegamento di software e modi di operare sicuri che riescono a rendere il sistema invulnerabile sempre e comunque (si parla di minacce conosciute e a meno di bug). Esaminiamo il modo in cui un Trustworthy System carica il kernel.

Come detto in precedenza il TPM non può "costruire" una situazione sicura senza l'ausilio di software programmato ad hoc, si definisce quindi il "root of trust" e cioè la radice con la quale inizializzare la catena di fiducia, dovendo l'anello n della catena essere controllato dall'anello $n-1$ è importantissimo che la radice sia la più sicura possibile. Quest'ultima viene realizzata mediante l'aggiunta pre-bios di un Core Root of Trust for Measurement, come si può notare in Figura 9, il CRTM quindi si va ad inserire come primo codice ad essere eseguito avendo il compito di controllare l'integrità del Bios, usando uno dei sedici registri (Platform Configuration Register) messi a disposizione del TPM. Successivamente il controllo viene passato proprio al Bios che essendo stato controllato fa "store" del suo stato e a sua volta passa il controllo al bootloader. A questo punto la situazione è stata sempre controllata e salvata nei PCR ma proprio ad un passo dal caricamento del sistema operativo ci troviamo in un

ennesimo anello che deve essere controllato anch'esso per rimanere in condizione di affidabilità. È proprio qui che ci viene in aiuto il TrustedGRUB che riesce ad estendere la "chain of trust" fino alla selezione del sistema, rendendo questa la più sicura possibile impedendo (o comunque visualizzando un messaggio di avviso) il normale caricamento di un sistema che è stato manipolato/modificato in maniera untrust.

Si noti che il TrustedGRUB non si limita soltanto a verificare l'integrità del sistema ma riesce a controllare qualsiasi file di sistema l'utente necessiti di sorvegliare, come ad esempio file contenenti password. Il tGrub infatti, ha un'ulteriore funzione chiamata "checkfile" con la quale calcola le funzioni hash degli archivi da controllare per poi, ad ogni accensione, fargli fare match con i file stessi. Ennesima, ed ultima, funzione di questo Grub è la possibilità di crittografare dati per poi poterli decifrare non con una classica chiave ma grazie ad una chiave-configurazione che non è altro che la configurazione del sistema, i dati crittografati saranno accessibili quindi solo se il sistema non è stato modificato da quando è stata applicata la codifica.

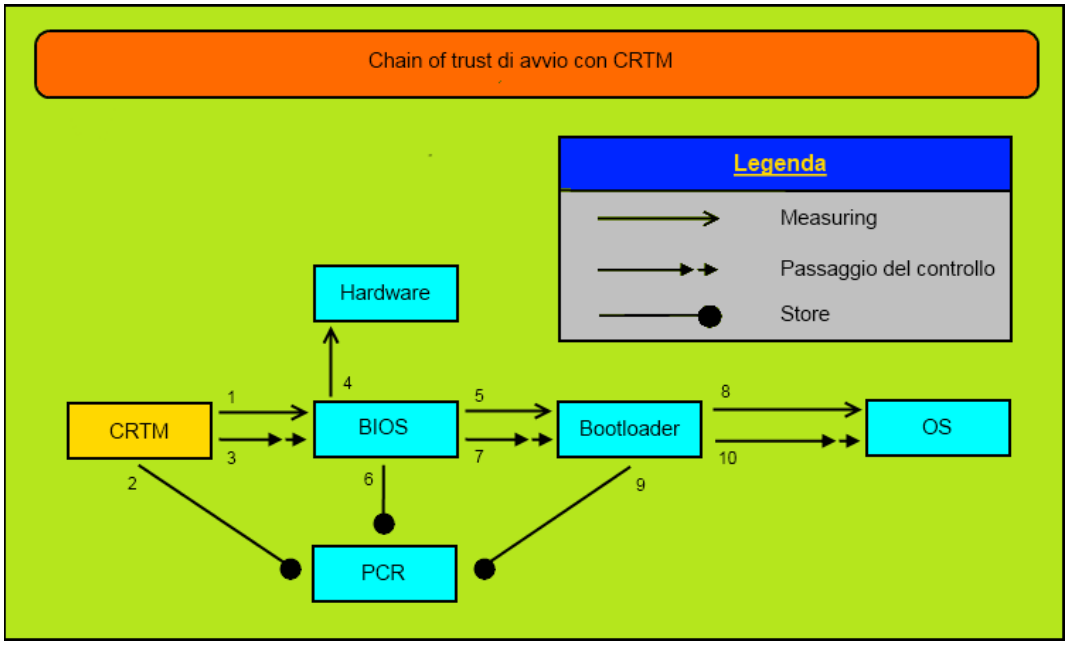


Figura 9: Catena di fiducia con CRTM

Abbiamo detto che per "settare" questa catena di fiducia è necessario fare "store" di dati nei registri PCR (o comunque utilizzarli come aiuto per il match delle funzioni SHA-1), andiamo a vedere come il tGrub utilizza questi 16 registri forniti dal TPM:

- Registro 4: Informazioni sullo MBR;
- Registro 8: Informazioni sul Bootloader (prima parte);
- Registro 9: Informazioni sul Bootloader (seconda parte);
- Registro 12: Argomenti a riga di comando e di shell;
- Registro 13: Tutti file sottoposti al controllo "checkfile";
- Registro 14: Tutti i file che devono essere caricati fino al sistema operativo.

Si noti che questa configurazione la si trova in una macchina sulla quale il sistema è fatto partire dal disco fisso, ma come sappiamo è possibile fare boot anche da un altro dispositivo, in questo caso il PCR numero 8 rimarrà vuoto mentre il 9 conterrà l'immagine di boot.
Ulteriore agevolazione derivante dall'uso dei Platform Configuration Register consiste nel velocizzare l'operazione di boot, infatti i primi TrustedGRUB utilizzavano il calcolo SHA-1 messo a disposizione dallo hardware del sistema ma questo portava a rallentamenti, si creava un vero e proprio collo di bottiglia dato dal numero dei file da controllare, rendendo la situazione insostenibile per un numero elevato di questi. Si preferisce, ad oggi, farsi supportare dai PCR ottenendo una buona soglia di velocità.

4.1.8 Trusted Network Connect di protocollo AAA

Le reti informatiche sicure non sono altro che reti nelle quali non è possibile essere messi sotto attacco dai cracker informatici e comunque si è certi (questa è almeno la speranza) di non imbattersi con software malevolo. Come si può però creare una di queste reti a rischio tendente a zero? È possibile sfruttare i già esistenti protocolli di rete detti AAA e magari entrare a far parte della rete tramite DAA (Direct Anonymous Attestation) per continuare a celare la propria identità.

Conosciamo ora meglio i protocolli da utilizzare per implementare questa specifica TCG. Il protocollo AAA, detto così perché crea le sue basi nei tre concetti *authentication*, *authorization* e *accounting*, ci permette di instaurare connessioni nelle quali gli utenti e le loro operazioni, possono essere monitorate. AAA non è un vero e proprio protocollo ma una famiglia di protocolli, possiamo infatti collocare in essa le specifiche relative a RADIUs, DIAmETER e TACACs (tre protocolli per l'autenticazione).

I principi fondamentali sui quali si basa l'idea di una rete "fidata" sono principalmente quattro e vanno sotto il nome di: **Platform-Authentication**, **Endpoint Policy Compliance**, **Access Policy** e **Assessment, Isolation and Remediation**, andiamo ad analizzarli univocamente.

Quando parliamo di *Platform-Authentication* vogliamo intendere che un utente con il desiderio di connettersi ad una determinata piattaforma deve necessariamente rendersi riconoscibile, ma allo stesso tempo, dopo essersi autenticato in maniera del tutto automatica, la stessa piattaforma deve sincerare l'utente relativamente alla sua integrità tramite la procedura chiamata "Integrity Check Handshake". Si ha quindi una sicurezza bi-direzionale tra il fruitore e il fornitore di servizio.

Nel secondo principio invece (*Endpoint Policy Compliance*) si cerca di stabilire quale sia il grado di fiducia da attestare all'endpoint in essere, quando questa fase è in esecuzione la piattaforma remota è in grado di "leggere" alcune informazioni

del terminale che vuole connettersi ad essa. Sono proprio queste informazioni che fanno variare il livello di fiducia della macchina presso la piattaforma, ed è facile capire il perché semplicemente elencandone alcune: lo stato del sistema, la versione del sistema, le patch applicate allo stesso, la versione dei software interessati alla comunicazione, il rilevamento intrusioni tentate sul sistema, la versione database virus e persino il numero di volte in cui il sistema si è dovuto "difendere" (anche se l'operazione è andata a buon fine!) . Questo concetto si lega alle *Access Policy* che creano la connessione solo se gli utenti erano in condizione sicura anche a priori della connessione.

Ultimo principio è quello che va sotto il nome della tripla *Assessment, Isolation and Remediation*, essa non fa altro che descrivere come la rete si comporta davanti ai tentativi di connessione. Infatti appena un utente invia la richiesta, la rete valuta il suo sistema, e nel caso questo sia giudicato non valido per entrare a far parte della TNC verrà all'istante isolato dalla rete stessa (come se fosse in quarantena), ovviamente l'operazione non si limita a questo ostracismo ma prosegue dando allo stesso utente una serie di "consigli" da eseguire se vuole rientrare in possesso del diritto di accedere.

Andiamo a vedere come è stata creata l'architettura delle Trust Network Connect dal TCG. Possiamo distinguere cinque "soggetti" principali che vanno sotto i nomi di **Access Requestor** (AR), **Policy Enforcement Point** (PEP), **Policy Decision Point** (PDP), **Metadata Access Point** (MAP) e il **MAP Client** (MAPC), queste funzionalità collaborano insieme per garantire la sicurezza della rete in questa maniera: l'AR richiede l'accesso alla rete protetta, successivamente il PDP confronta le credenziali ottenute con le politiche di sistema, a questo punto il PDP comunica le informazioni ricevute al PEP che decide di fatto se AR può entrare a far parte della rete, in maniera facoltativa il Client Map può coordinare le operazioni di PDP e PEP per rendere la rete ancora più sicura monitorando lo scambio di informazioni che passano per MAP.

Viene mostrata nella Figura 10 un'immagine che rende l'idea di come TNC sia stata implementata, si noti che questa architettura è volutamente generalizzata per rendere l'idea di quanto possa adoperarsi per gestire l'interoperabilità, in ogni colonna è possibile distinguere uno dei cinque soggetti ed in ogni riga di

riferimento è possibile vedere le funzioni che entrano in gioco.

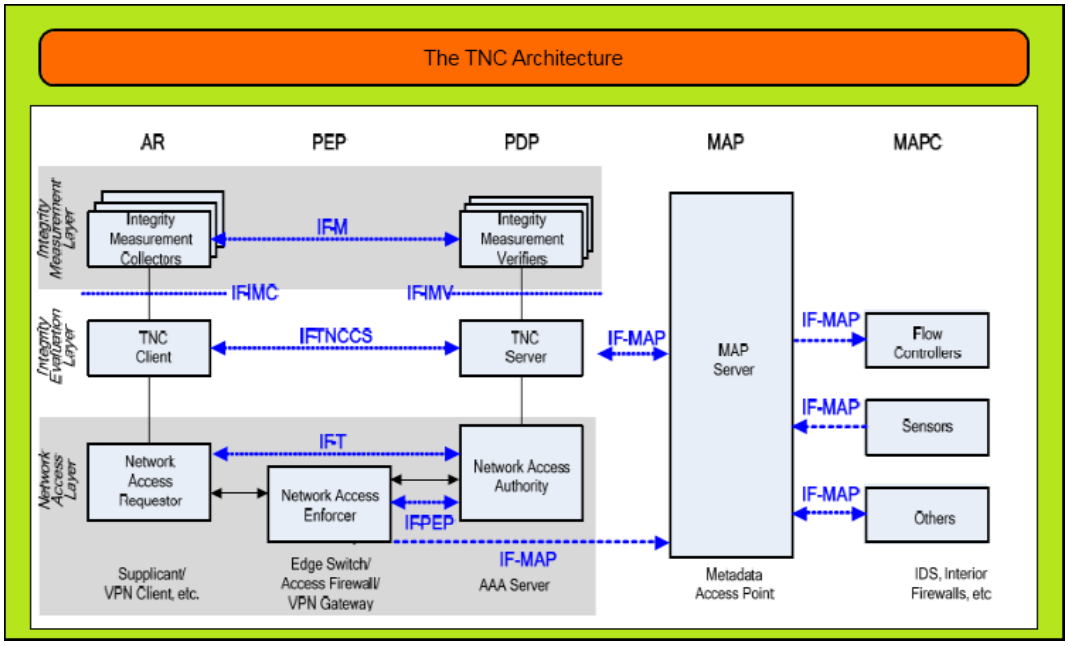


Figura 10: Architettura di una Trusted Network Connect [15]

Cerchiamo ora di definire le funzioni che ogni "soggetto" ha a disposizione per far funzionare questo macro-concetto. Per poterlo fare abbiamo bisogno di definire tre gruppi di funzioni che permettano di identificare queste, dal Trusted Computing Group questa selezione è definita come Layer (livello):

- The network access layer: fanno parte di questa categoria le funzioni che stabiliscono la connettività di rete, supportandola maggior parte di esse (quali 802.1x, AAA), troviamo infatti qui il **Network Access Requestor** (NAR), il **Network Access Enforcer** (NAE) e il **Network Access Authority** (NAA);

- The integrity evaluation layer: come immaginiamo dal nome in questa categoria troviamo le funzioni che permettono la verifica dell'integrità e sono il **TNC Client** (TNCC) e il **TNC Server** (TNCS);

- The integrity measurement layer: questo è un semplice livello di plug-in per la verifica dell'integrità del sistema, troviamo quindi la funzione **Integrity Measurement Collectors** (IMCs) e la funzione **Integrity Measurement Verifiers** (IMVs).

Visti i livelli delle funzioni, andiamo ad esaminarle singolarmente e in base al loro domini di interesse (i soggetti citati). Per quel che riguarda l'AR avremo una *Network Access Requestor* che provvederà ad esaminare le definizioni di accesso alla rete, una *TNC Client* che non è altro che una componente software dell'AR stesso in grado di fornire supporto allo handshake e infine una *Integrity Measurement Collector* con il compito di salvare e comunicare lo stato del sistema che richiede l'accesso. La funzione che fa parte del PEP è quella derivante dal primo layer e cioè la *Network Access Enforcer* che infatti, in accordo con le operazioni del *Policy Enforcement Point*, non fa altro che calcolare se l'accesso può essere fornito al richiedente. Tre sono invece le funzioni che riguardano il PDP e cioè le rimanenti la NAA la TNCS e la IMV, rispettivamente queste valutano se il richiedente raggiunge la soglia minima per avere un accesso consentito, gestiscono il flusso di messaggi e infine verificano l'integrità dell'AR mediante le misurazioni ottenute dalle funzioni IMC. Per quel che riguarda MAP e il suo Client sono definite la funzione *Metadata Access Point Server*, in MAP, per aiutare PEP e PDP nell'operazione decisionale d'accesso e le funzioni Flow Conter e Sensor in MAPC, usate rispettivamente per monitorare la rete accertandosi se questa rispetta le decisione prese e per spedire questi flussi di monitoraggio al MAP. [15]

4.2 Applicazioni future del TC

4.2.1 Infrastrutture Multi-tenant Fidate

Una delle future applicazioni della filosofia "fidata" alla base della Trusted Computing Group sarà quella di rendere sicure le infrastrutture condivise e il loro modo di operare. Possiamo comunque evincere, direttamente dalla TGC, che questo nuovo working group (Trusted Multi-Tenant Infrastructure) non si limiti agli obiettivi prefissati ma sia aperto a sviluppare le proprie idee per un qualsiasi nuovo sviluppo, infatti: "Our philosophy is to release these for discussion as groups of related use cases are ready, in order to generate more insight and input from enterprises deploying or considering employment of cloud services." [16]
In ogni caso possiamo dire che questo gruppo è stato fondato per la creazione di standard e modelli per la creazioni di infrastrutture di fiducia, così da rendere sicuri servizi quali il cloud computing o semplicemente la gestione di sistemi multi-tenant.
L'idea di creare un nuovo settore apposito per monitorare le questioni d'interesse dello stesso non è stata partorita a caso, ma si è resa necessaria grazie allo sviluppo, sempre più crescente, di infrastrutture condivise e di ciò che queste possono offrire, basti pensare al sempre più diffuso cloud computing che in pochi anni ha conquistato i cuori degli internauti. Parole d'ordine sono *Monitorare* e *Real-Time*, rendere sicuro un servizio cloud, o comunque riuscire a garantire un buon grado di fiducia su una piattaforma condivisa, non è come rendere protetto un sistema che "vive dentro quattro mura", infatti, oltre a dover garantire quest'ultimo all'utente, si rende necessario garantire, nello stesso istante di tempo, una fiducia bi-direzionale. L'aspetto può sembrare già visto avendo parlato di TNC, ma questa affermazione non è vera, si deve qui familiarizzare con un concetto nuovo, ovvero la possibilità che ha l'utente di sfruttare un servizio in una nuova maniera che esula dall'insigne essenza client/server.
Il lavoro del working group TMI si basa sulla costituzione di specifiche

riguardanti la creazione di un nuovo framework aperto in grado di implementare nella migliore maniera possibile i servizi che abbiamo citato sopra, cercando, dove è possibile, di colmare le lacune che questi hanno creato in riferimento al modo di procedere della pluri-menzionata Trusted Computing Group. Sono proprio queste lacune che hanno, nella mente del TMI, fatto scattare le parole chiave di cui parlavamo prima, l'utente non deve essere in grado di operare senza che venga "controllato", pena, a patto di operazioni illecite, il mal funzionamento del suo sistema o addirittura di quello centrale!
Andiamo ora ad esaminare come sarà possibile ottenere queste soluzioni. Il Trusted Multi-Tenant Infrastructure si propone di affrontare il problema fissando la sua attenzione su tre punti principali, e cioè: riuscire a stabilire un contesto di fiducia, riuscire a scambiare informazioni mantenendo questo contesto e cercare di applicare tutte le policy di sicurezza create per il servizio richiesto. È proprio in base ai servizi che otteniamo la prima classificazione resa nota dalla TCG, che si basa sugli scenari in cui può essere applicata questa nuova struttura framework, distinguiamo:

- TCG Generic Use Cases:
  Questo è il caso d'uso generico in cui non si applicano le accortezze apportate dal TMI;
- TCG Multi-Tenant Use Cases:
  Il secondo caso d'uso si applica quando si è in presenza di una struttura condivisa e si vogliono far valere le idee inserite dal TMI;
- TCG Multi-Tenant Use Case Scenarios:
  Terzo ed ultimo caso, questo si differenzia dal secondo perché tenta di esaminare uno scenario particolare quindi una specifica istanza di un caso d'uso che può interessare ad uno specifico servizio oppure ad una specifica azienda;

I partecipanti a questo scenario non sono altro che il fornitore e il cliente dei servizi richiesti, per ognuno di essi possiamo distinguere sei casi che non sono altro dei modi di operare, ben specificati, attualizzati quando viene fatta una richiesta o viene effettuata una prestazione di servizio.

Le cause che caratterizzano il fruitore sono:

- Richiesta di modifica delle policy in atto al momento;
- Gestione del Trusted System Domain tramite il Consumer Management Agent;
- Uso del Consumer Management Agent dopo la modifica dello stato stabile del Trusted Systems Domain;
- Uso del Provider Management Agent dopo la modifica dello stato stabile del Trusted Systems Domain;
- Revoca dalla rete protetta del componente richiedente il servizio (Asset);
- Revisione delle policy del dominio di fiducia.

Mentre le cause che caratterizzano il fornitore sono:

- Sancire le relazioni di fiducia tra i fornitori;
- Modifica delle policy di fiducia dei fornitori accorsi;
- Garanzia delle operazioni consentite all'asset;
- Trasferimento di un asset se questo si trova in un dominio di errore;
- Rifornitura dei servizi all'asset trasferito;
- Revisione delle policy del dominio di fiducia.

Per capire meglio come questa architettura è stata pensata dal TCG accludiamo un'immagine che rende l'idea di come sia possibile realizzare in futuro uno di questi sistemi, figura 11.

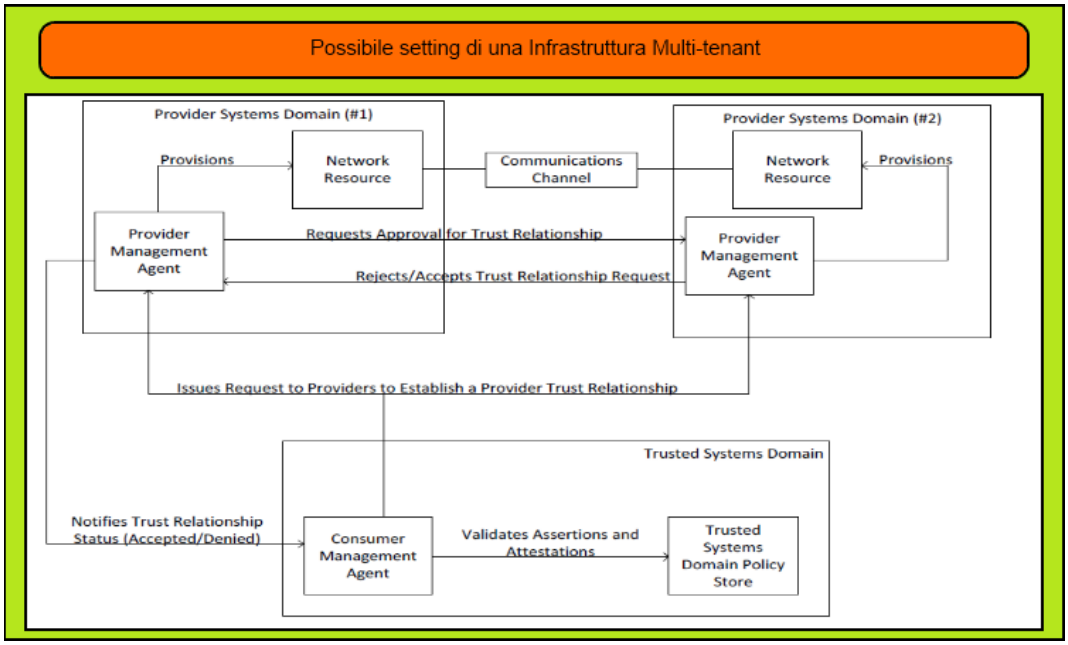


Figura 11: Settaggio TMI per uno Use Case specifico [17]

4.2.2 Digital Rights Management

Inserire i DRM in un capitolo in cui si parla di applicazioni future può sembrare un errore, ma si deve considerare il fatto che questo metodo di difesa è in continuo sviluppo e questa evoluzione non sempre, come accade per una qualsiasi altra forma di difesa informatica, inizia una nuova fase di progetto partendo dai punti raggiunti o comunque con la scoperta di nuovi tipi di supporti. Il progresso in questa tecnologia è dettato, tanto è vero, più da "mode" del momento che dall'usufruire delle ultime tecniche informatiche esistenti.

Cerchiamo di inquadrare meglio il concetto di *Digital Rights Management*. Questi non sono altro, come da traduzione, i diritti che un autore può vantare nella creazione di un'opera, e con questa ultima parola intendiamo proprio un qualsiasi genere di opera, per capirci meglio: non vogliamo indicare solo del codice di programmazione.

Questo è uno dei fattori principali che "frena" chi cerca di tutelare questi diritti e

crea quell'effetto "ballerino", di cui parlavamo prima, nello sviluppo di queste tecnologie. Si pensi infatti ad una mostra d'arte di un pittore vivente, questi avrà tutta l'accortezza nel far pagare il biglietto a chi vuole entrare a visitarla ma allo stesso tempo, di certo, non incriminerà chi crea rappresentazioni dei suoi dipinti facendo accrescere il suo nome. Questo è il limite "variabile" creatosi nei DRM. Per quel che riguarda la difesa di opere, ovviamente digitali, soggette a diritti d'autore sono state sperimentate varie tecniche negli anni e tutte hanno avuto alla base il concetto di acquisto del diritto di utilizzo come la procurazione di una chiave in grado di decodificare (o di rendere funzionante) un articolo a disposizione.

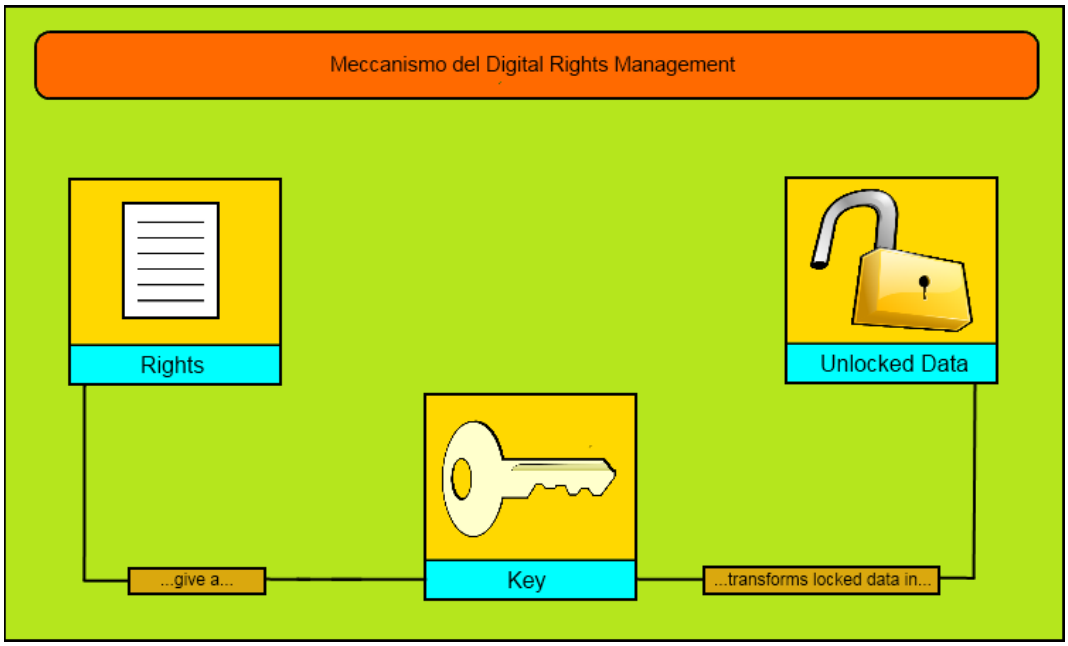


Figura 12: Modus Operandi DRM

Tra queste tecniche che permettono di fruire di un bene spicca il *Watermarking*, questa è una tecnica che si è proposta di proteggere le opere tramite l'inserimento, nelle stesse, di flussi digitali che contengano informazioni. I modi in cui è possibile implementare questa scelta sono principalmente due: il *visible*, che consiste nel inserire un logo (come quello televisivo) che identifica immediatamente il diritto d'autore, e lo *hidden*, ovvero quel metodo in cui grazie a

tecniche, quali la steganografia, è possibile inserire flussi di dati non riconoscibili che non permettono l'esecuzione se non si posseggono i diritti.
E per quel che riguarda lo sviluppo futuro? Negli ultimi tempi stanno prendendo piede i linguaggi Rights Expression Language, linguaggi di programmazione basati su XML, che permettono l'attuazione di tecniche DRM. Uno di questi linguaggi è quello che va sotto il nome di ODRL (Open Digital Rights Language), che solo da poco è giunto alla versione 2.0 avendo la ODRL Initiative pubblicato solo a febbraio 2011 la quinta specifica preliminare relativamente al suo modello di base. Le informazioni che possiamo astrarre da questo modello riguardano proprio la configurazione e il funzionamento di questo, possiamo notare infatti come il linguaggio sia stato definito in otto entità primarie, appoggiate da due entità secondarie fungenti da classi di associazione che non fanno altro che colmare le informazioni tra le relazioni esistenti, di seguito possiamo notare una formalizzazione del modello di ODRL 2.0 (Figura 13) definito come segue: "The ODRL Core Model is designed to be independent from implementation mechanisms and is focussed on the optimal model and semantics to represent policy-based information."[18].

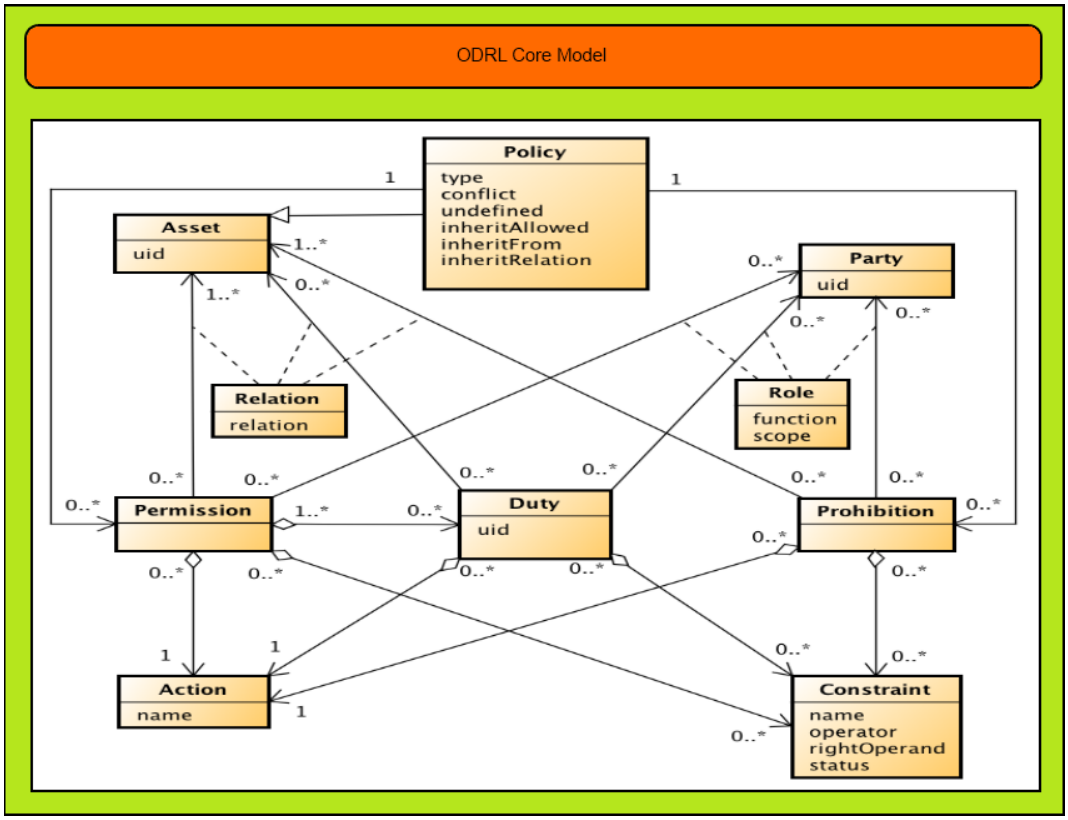


Figura 13: Modello del nucleo operativo ODRL [18]

Come si può notare dall'immagine esiste una macro entità chiamata **Policy** in cui gli unici attributi richiesti sono lo ID personale e il tipo di politica di gestione che dovrà essere messa in atto. Le prime due relazioni che saltano all'occhio sono quelle che Policy ha con **Permission** e **Prohibition**, altre due entità che stanno ad indicare ciò che è concesso e ciò che è negato ad fruitore da parte di un fornitore, si noti che situazioni in cui un cliente non abbia proibizioni esistono e sono da considerare, non vale però il rovescio della medaglia, infatti un compratore che non ha permessi non sussiste, questo porterebbe al decadimento del modello, è quindi sensata la scelta di non implementare situazioni di questo genere. A questo punto è necessaria un'ulteriore entità che figuri il bene di interesse, fa le veci di questa **Asset** che grazie al suo unico attributo, che funge da chiave primaria, identifica univocamente un oggetto di diritto. Dualmente, dopo aver definito la "merce" è necessario definire il soggetto a cui questa deve essere

collegata, a questa catalogazione pensa **Party** che non si limita a singole persone ma si estende anche a veri e propri gruppi. Queste cinque entità potrebbero già bastare per definire un linguaggio XML in grado di gestire un sistema DRM, ma per completezza si è deciso di aggiungerne altre tre (per arrivare finalmente ad otto), completezza in questo sistema infatti significa poter rendere valido il modello nel tempo, portarlo quindi a riuscire a sostenere situazioni in cui i diritti di un soggetto variano. Vengono in nostro aiuto quindi **Duty**, **Action** e **Constraint**, rispettivamente queste entità indicano i requisiti da soddisfare per avere dei permessi, inviano le azioni (ben definite in un vocabolario) che devono essere svolte e segnalano le restrizioni da applicare. Con questo scenario il nostro codice è in grado di soddisfare anche i casi in cui una licenza d'uso estingua la sua validità dopo un certo periodo di tempo od anche dopo un certo numero di impieghi. Infine le classi di associazione **Relation** e **Role** fungono da espansione per le classi riguardanti i beni e i fruitori, creano infatti nuove informazioni riguardanti i parametri che un cliente ha verso i suoi averi, settando appunto i permessi del bene e i ruoli che i fruitori acquisiscono nel passaggio.
È comunque opportuno notare che le tecnologie Digital Rights Management, soffrono di una debolezza insita in ogni sistema anti-copia chiamato *Analog Hole*, come abbiamo sopra accennato, infatti, ogni sistema DRM non fa altro che convertire il flusso di dati (prodotto più tecniche) dal formato digitale in analogico, ma una volta ottenuto questo è possibile riconvertirlo in digitale senza apporre elementi di cifratura, guadagnando così il fine di eludere i sistemi coercitivi per la tutela del copyright.

4.2.3 Difesa dagli attacchi phishing

Come detto in precedenza, uno degli attacchi più diffusi sulla rete è proprio quello va sotto il nome di phishing, questo attacco cerca di appropriarsi dei dati di autenticazione di un utente indirizzandolo verso una finta pagina di login. Sarebbe possibile risolvere questo problema sfruttando il protocollo crittografico

SSL ma il prerequisito per l'utilizzo di questo è la proprietà di un certificato che si comporti come una chiave pubblica nella comunicazione client-server e non è possibile costringere tutti gli utenti a possederne uno. In che modo potrebbe essere possibile sfruttare il Trusted Platform Module in maniera che questo vigili sui possibili casi di phishing? Andiamo a vedere una possibile applicazione.

Innanzitutto dobbiamo avere un browser che riesca a gestire una *certificate table*, nient'altro che una tabella in cui sono associati dei certificati validi a siti internet che richiedono un'autenticazione. I certificati presenti in questa "memoria" sono scindibili in due tipi: i classici e quelli del TPM. Come dal nome, i certificati classici, sono quelli che sono stati sempre utilizzati anche senza l'apporto del Trusted Computing, per capirci sono tutte quelle attestazioni che possono essere ricevute iscrivendosi a servizi web che ne richiedono e ne forniscono. È possibile però che non si abbiano certificati per accedere a particolari servizi web anche perché questi non ne richiedono essendo magari stati sviluppati da utenti malevoli (appunto siti che utilizzano la pratica del web spoofing), come potremmo procedere in questa situazione in modo tale che gli attacchi siano evitati direttamente dal nostro software e non dal nostro intelletto?

La tecnica consiste nel forzare un'autenticazione con certificati in ogni "luogo" che necessiti di scambio di dati sensibili. Fanno la loro entrata in scena il TPM e le sue funzioni, in particolare la *TPM MakeIdentity* che riesce a creare una nuova Attestation Identity Key da utilizzare come autenticazione. Appena creata la nuova accoppiata di chiavi, la parte pubblica viene inviata ad una CA di fiducia che provvede a certificarla apponendo la sua firma (si noti che la CA verifica se la chiave pubblica inviata è stata creata da un TPM). Questo certificato firmato viene spedito indietro e si provvede a certificarlo ulteriormente con una chiave privata del TPM (chiave estratta da un ulteriore accoppiata), a questo punto avremo quindi un certificato basato sulla nostra chiave pubblica derivante da un AIK "a", attestato da un'autorità fidata e infine cifrato con una chiave privata derivante da un AIK "b".

Questo sarà il certificato con il quale si potrà rispondere all'handshake della richiesta del web service. A questo punto sarebbe possibile, tramite i protocolli della Trusted Network Connect, costringere le autenticazioni sui web service

basandosi su questi punti:

- Permettere solo le doppie attestazioni;
- Costringere i web service a certificarsi, pena la mancata visualizzazione di loro stessi su sistemi trust;
- Costringere ad una sorta di aggiornamento dei certificati posseduti dai web service (non a doppia attestazione perché vecchi);
- Blocco dell'invio delle informazioni a web service fasulli (vengono visualizzati perché sono normali pagine web ma non ricevono informazioni visto che il TPM client non permette l'invio);
- Impossibilità dei web service di richiedere password prima della fase di handshake.

L'unico problema potrebbe essere la richiesta di un utente di potersi autenticare da macchine diverse. Questo però può essere facilmente risolto ricorrendo alla Certified Migratable Keys, una chiave creata dal TPM ma non certificata da una CA ma da una Migration Authority (MA), in questo modo è possibile, da parte del proprietario del sistema, autorizzare la migrazione della chiave, in questa maniera si avrà la certezza di operare in sicurezza.

4.2.4 Difesa dallo spamming

In questo paragrafo discuteremo di un metodo anti-spam presentato al *Mit Spam Conference* nel marzo 2010 [19] che si occupa appunto di limitare o cancellare del tutto lo spam nella posta senza pero incappare in disagi quali confondere messaggi innocui con altri da fermare.
La tecnica in questione viene chiamata Trustworthy Self Regulation, in acronimo TSR, e crea le sue fondamenta nel fatto di permettere all'utente di riconoscere il protocollo di invio dei messaggi, in questo caso un protocollo esente da spam. Tutta questa "architettura" è resa in atto tramite dei

microprogrammi che si interfacciano tra loro chiamati *Law-Governed Interaction* (LGI), assieme alla possibilità di riuscire a capire se i messaggi inviati sono *L-Trust* (inviati con tecniche TSR).

L'operazione che un mittente deve eseguire per assicurare al destinatario che un messaggio risulti fidato, verrà chiamato chiamato *L-Mail*, è inserire un tramite fidato, indicato T^{L}, in grado di dimostrare che i dati inviati sono sicuri e conformi alle "leggi" della tecnica in uso. Come è possibile immaginare, non è fattibile utilizzare un solo controller T^{L} per tutto il traffico di messaggi sulla rete ma si rende necessaria una decentralizzazione degli intermediari, in modo da non congestionare il traffico. Si parla quindi di DTCB (Decentralized Trusted Computing Base). Il problema però non è solo la decentralizzazione ma anche il modo di "costruire" questi controlli, infatti essi devono essere implementati in modo che godano di fiducia dei destinatari e, allo stesso tempo, in maniera da riuscire ad interpretare e applicare le varie leggi applicabili ai dati viaggianti nella rete, si osservi che queste leggi sono in numero elevato e soggette a repentini cambiamenti, quindi i controller dovranno essere il più duttili possibile per sopportare le modifiche. Per ottenere questo, i tramiti vengono eretti in modo da essere più generici possibile e operare in modo da poter ottenere le leggi direttamente dal destinatario, provvedendo poi alla gestione del protocollo.

Nel corso della presentazione al congresso sono state esposte cinque tipi di leggi in grado di supportare la Trustworthy Self Regulation, o meglio quattro tipi di leggi ben definite più una ibrida. Le leggi sono:

- Payed Postage;
- Rate Limiting;
- IO (con supporto a Opt-In e Opt-Out);
- GC (supporto alla comunicazione nelle comunità);
- Altre leggi ibride.

Andiamole ad analizzare una per volta.

La *Payed Postage* consiste nell'inviare ad un certo destinatario, dei messaggi chiamati PP-Mail che si ottengono tramite uno specifico controller chiamato T^{pp}, si avrà così una sorta di messaggio con un timbro digitale, fornito da un Payement Service Provider (PSP), che accerterà la verifica del controller suddetto (questa è, in pratica, la digitalizzazione della classica lettera spedita per posta dove il timbro elettronico fa le veci di un francobollo). Il mittente potrà effettuare un grande acquisto di "francobolli" dal PSP proprio grazie alla divisione del protocollo, che certifica il destinatario solo nella fase gestita dal controller e non in quella dell'acquisizione di timbri, che risultano quindi assoggettati al mittente e impersonali per il destinatario. Tramite questa tecnica è possibile delimitare un numero di mazzi m di n messaggi inviabili ogni tempo t (solitamente un giorno), ovviamente non c'è limite al numero di m, ma per dare il via al mazzo successivo è possibile richiedere una password. Questo accorgimento, rendendo n variabile con il variare di m, può essere usato come difesa negli attacchi di tipo zombie non permettendo l'invio di PP-Mail (e quindi la ricezione di messaggi da parte di un destinatario) in un sistema corrotto.

Discorso simile è quello del *Rate Limiting* che permette l'invio di un numero di messaggi controllato, in pratica il TSR blocca (meglio dire non inoltra) i dati in uscita dal mittente se questi superano un limite massimo, solitamente giornaliero, o comunque se vengono inviati troppi messaggi nello stesso istante, per fare un esempio si potrebbe dire che i prelievi di alcuni conti correnti sono soggetti a Rate Limiting, infatti non è possibile prelevare più di una cifra x' mensilmente e allo stesso tempo non è possibile prelevare più di una cifra x" per volta.

Successivamente troviamo la legge *IO*, che si riferisce a Opt-In e Opt-Out due concetti opposti che indicano rispettivamente l'obbligatorietà di avere un dichiarazione esplicita a inviare dati e viceversa una possibilità di inviare dati a chi non esprime la volontà di non volerne ricevere (il classico "chi tace acconsente"). Per gestire questa legge entra in scena il controller T^{io} che va a gestire le politiche di invio, supportato dal mittente del messaggio che deve mantenere due liste separate nello stesso sistema, appunto la "opt-in" e la "opt-out". Proprio l'imposizione di dover mantenere le liste nello stesso sistema

(e la loro inscindibilità) regala a questa legge la sua sicurezza. Un utente con una lista opt-out molto estesa potrebbe avere la malsana idea di sbarazzarsene, ma in questo caso dovrebbe cambiare sistema e rinunciare anche alla opt-in, ciò tende a scoraggiare lo stesso in modo che chi ha richiesto un blocco di messaggi da parte di quel destinatario non possa tornare alla situazione di partenza in futuro.
Quarta legge è quella che si occupa di monitorare sui gruppi di discussione (chiama *Group Comunication*) o meglio utilizza questo approccio. È realistico pensare che gruppi di lavoratori, amici e quant'altro non siano soggetti ad attacchi spam, nessuno per esempio invierà pubblicità in un gruppo di lavoro se lui stesso ne fa parte. Si è scelto questo modo di relazionarsi come ispirazione di questa legge, in pratica più entità che vogliono essere gruppo, non permettendo ad altri di inviare messaggi al loro interno, si certificano presso una CA come singoli facenti parte di un'aggregazione ed a sicurezza attivata non sono in grado di inoltrare mail certificate ad utenti non membri del gruppo o dei gruppi di appartenenza. Si noti che un individuo che invia un messaggio non consono alle regole viene subito espulso dal gruppo inerente ed essendo l'espulsione eseguita tramite la revoca del certificato della CA, lo stesso individuo sarà buttato fuori da tutti i gruppi in cui è certificato dalla stessa autorità (un'autorità che non può più fidarsi di un soggetto revoca tutte le istanze di certificazione per esso).
Per portare le tecniche Trustworthy Self Regulation in auge c'è bisogno di settare nuovo sistema di comunicazione via rete che permetta la gestione di questo, una buona idea potrebbe essere configurare gli account esistenti in modo che riescano ad accettare tutte le leggi esistenti e inviino messaggi con una legge *non-legge* automatica (una prassi individuata come legge ma che non consista in nessun modus operandi), in questa maniera tutti gli utenti farebbero parte di una "rete" TSR senza però dover gestire nulla di nuovo, avrebbero l'impressione di operare con i classici attuali sistemi.

Capitolo 5

Critiche al Trusted Computing

In questo ulteriore capitolo andremo a citare le massime critiche che sono state mosse al TC focalizzandoci sui rischi e i disservizi in cui l'utente potrebbe incappare utilizzando le tecniche proposte dalla Trusted Computing Group.

5.1 Owner Override

Immediatamente dopo l'applicazione dei primi sistemi di Trusted Computing si è resa nota una delle prime problematiche derivanti da questa tecnologia, ovvero la perdita di controllo sulla propria macchina, può succedere infatti che questa, oltre ad essere protetta da minacce, sia anche protetta da alcune operazioni volute dal proprietario stesso. Questa è una situazione che non può andare di certo a genio a chi "predica" la totale libertà sui sistemi. La soluzione che venne presentata fu quella dell'Owner Override. Questa tecnica permette al proprietario del sistema di bypassare alcuni controlli, o comunque di avere esito positivo da essi, fingendo di essere cosa non si è in realtà modificando il proprio stato. Ciò può sembrare un'operazione illecita, ma se si ragiona sui controlli che la remote attestation permette di eseguire, in un attimo si comprende perché l'Owner Override nasca non per imbrogliare qualcuno ma bensì per garantire dei diritti inequivocabili per il proprietario. Immaginiamo, ad esempio, una situazione in cui il proprietario di una macchina voglia eseguire su di essa del software, magari scritto proprio da lui, e non sia in grado di farlo perché le politiche decisionali del costruttore non lo permettono, si sarebbe davanti ad una situazione paradossale: una macchina che **non è in grado** di eseguire del codice!

Come è possibile rendere nulle le verifiche del sistema assieme alle sue decisioni tramite Owner Override? Esistono due modi di operare in questo senso e ovvero, la totale disabilitazione del sistema TPM oppure la falsificazione delle chiavi. Chiedersi perché l'approccio viene fatto tramite due filoni così differenti è tanto normale quanto sia elementare rispondere alla stessa domanda. Disabilitare il Modulo di fiducia in un sistema rende questo libero ma comunque vincolato, non si ha la possibilità di far "girare" software che richiede un'attestazione da parte del fruitore di questo, la soluzione quindi può essere accettata solo ed esclusivamente da chi non ha interesse ad utilizzare uno specifico software di una ben determinata casa produttrice. Proviamo a pensare a chi però non può rinunciare a questo, un utente che possiede dell'hardware (come un dispositivo mobile) che può funzionare appieno solo se è gestito su una particolare

piattaforma SW, è letteralmente impossibile impedire l'utilizzo di questo programma a questo individuo. Viene quindi in aiuto la falsificazione dei certificati che “truffano” i sistemi di controllo remoto facendo credere ad essi che stanno operando con una macchina perfettamente configurata per operare con loro stessi.

Tuttavia questa soluzione è stata proposta e subito bocciata dai membri della TCG, ufficialmente perché l'Ownel Override potrebbe far riconoscere un utente malevolo in un sistema come fidato e questi avrebbe la possibilità di operare in maniera illecita tanto da portare al malfunzionamento, o addirittura ad un crash, del sistema stesso. È comunque lecito pensare che si sia scelto di non inserire l'Owner Owerride per scopi commerciali, visto che l'attuazione di questa, in maniera non del tutto controllata, potrebbe portare all'improduttività di controlli che permettono di verificare se l'utente “naviga” nell'illegalità, come l'inefficacia di un metodo DRM.

5.2 Lock-In

Ulteriore problema che inserisce la tecnologia di fiducia è quello del Lock-In e come ci suggerisce la parola stiamo parlando di un blocco, una chiusura che riguarda il software.
Abbiamo accennato nel paragrafo precedente ad utenti che non potrebbero, anche nel caso fosse possibile, utilizzare tecniche Owner Override visto che sarebbero impossibilitati a servirsi di software che richiederebbe un'attestazione remota. Il problema sembrava arrestarsi qui in merito all'impiego di programmi sulle nuove piattaforme, ma invece la smentita è proprio dietro l'angolo.
Come ormai specificato più volte, il cuore del TC è il Trusted Platform Module che provvede a tenere la situazione sotto controllo e decide cosa è possibile "fare" e cosa non lo è. Proprio qui sta il problema, come da specifica TCG un sistema affidabile non deve poter operare in situazioni untrusted, il punto è che a definire le situazioni con scarso grado di fiducia sono proprio le società che partecipano alla stesura delle specifiche. Con l'attuale implementazione del TPM il software è stato catalogato in due branche, quella sicura e quella non sicura, apparentemente questa divisione sembra non portare con sé dei problemi, infatti è sempre possibile eseguire software affidabile, e non si hanno problemi ad eseguire programmi dichiarati untrust, previa disattivazione delle politiche gestionali del TPM.
Se bene questa ultima affermazione sia sempre vera cela all'interno un difetto dato proprio dalle implementazioni targate TCG. Un utente, munito di un modulo di sicurezza, come detto prima, può deciderne la disattivazione e grazie a questa è logico pensare che costui si ritrovi ad usare una macchina che non ponga vincoli. In realtà non è così, il fruitore di un sistema trusted disabilitato è paradossalmente più vincolato di chi decide di non neutralizzare il famigerato sorvegliante (TMP). Dal Trusted Computing infatti possono crearsi delle situazioni di blocco chiamate Lock-In, situazioni nelle quali è impossibile eseguire, per presa di posizione (politiche con TPM attivo) o in via cautelare dell'utente, alcune operazioni.

Pensiamo al proprietario di un sistema che voglia eseguire in locale un software che si interfaccia con una stampante ma questo software è definito dal sistema come inaffidabile, a questo punto l'utente potrebbe riavviare la macchina e configurarla in modo da bypassare i controlli del TPM (disattivarlo), ignaro di cosa sta per succedere, questi installerebbe il nuovo programma e opererebbe di tutta tranquillità e magari non ri-attiverebbe neanche le politiche di sicurezza. Con questa operazione l'infausto "proprietario" (virgolettato a questo punto) potrebbe aver modificato lo stato del suo sistema, e magari potrebbe essere anche cliente di un software trusted di tipo word processor che permette l'apertura di file a soli utenti fidati. Lock! Il cliente nonché autore del file non è più abilitato ad aprire quest'ultimo, e l'unica cosa possibile da fare è rimediare riportando la macchina all'ultimo stato stabile salvato, pregando che nessun software abbia "giochicchiato" con le chiavi di sistema (questo renderebbe le operazioni di decifrazione dei file molto complesse). É una situazione invertibile certo, ma forse è anche inaccettabile. Logicamente, come per l'Owner Override, la TGC (o comunque le aziende membri) in via ufficiale dichiara che esistono programmi fidati per ogni operazione e quindi disabilitare il TPM non è necessario.
Per il momento questa è la classica situazione di Lock-In (scelte software obbligate), ma cosa potrebbe succedere se un'azienda mossa dalla voglia di spodestare qualsiasi concorrente sfrutti questo difetto del Trusted Computing per creare di proposito queste situazioni traendone vantaggio?

5.3 Richard Stallman e il Treacherous Computing

Cosa ne pensa Stallman del Trusted Computing che come abbiamo potuto vedere vincola l'utente? C'è da dire che Richard M. Stallman è uno dei principali attivisti del mondo informatico e sostenitore del software libero, nonché fondatore della Free Software Foundation e del tipo di licenza libera GPL. Detto ciò, è facile anticipare come il programmatore statunitense reagì ai primi concetti che nascevano già dalla Trusted Computing Platform Alliance (madre della TCG).
Il pensiero di Stallman riguardo l'informatica si è sempre fondato sul principio "la macchina è mia e la gestisco io", ovvero sulla possibilità di studiarne la composizione e di avere il modo di modificare i sorgenti per rendere il sistema più performante e magari evitare qualche restrizione di troppo. L'informatico si esprime verso il TC ribattezzandolo come Treacherous Computing (informatica infida), lui infatti ritiene che questa nuova tecnologia nasconda dietro il suo volto d'angelo una faccia da demone. Lo statunitense infatti crede che le politiche di gestione dell'informatica fidata celino a noi utenti il loro lato peggiore, quel lato che li spinge a creare tecniche non tanto per migliorare l'informatica ma quanto per fortificare i metodi di controllo e portare i clienti ad essere una sorta di automi senza libero arbitrio, costretti ad essere "governati" dallo stesso sistema acquistato. Purtroppo la mancanza di libero arbitrio non spaventa le generazioni del momento (o forse queste non si rendono neanche conto di cosa stanno perdendo, avendo magari subito un lavaggio del cervello paragonabile al progetto MK-ULTRA). Stallman però non è uno sprovveduto, ed è sempre stato consapevole che non è sempre possibile convincere tutti i fruitori di sistemi che una nuova tecnologia non è stata creata per gli scopi dichiarati, proprio per questo motivo è lui stesso, in una sua pubblicazione, a cercare di far aprire gli occhi a tutti gli individui, fornendo esempi su come potrebbero essere utilizzate in futuro alcune tecnologie nate dalla Trusted Computing Group: "Imagine if you get an email from your boss stating a policy that is illegal or morally outrageous, such as to shred your company's audit documents, or to allow a dangerous threat to your country to move forward unchecked. Today you can

send this to a reporter and expose the activity. With treacherous computing, the reporter won't be able to read the document; her computer will refuse to obey her. Treacherous computing becomes a paradise for corruption.” [20]. In pratica lo statunitense crede che il TC possa essere usato (abusato) dalle società in modo che questo possa trasformare una situazione palesemente illegale in una lecita e tutto questo “grazie” al fatto che sarebbe impossibile dimostrare di aver letto dei dati.

Altro punto in cui Stallman si focalizza è la circostanza, plausibile, in cui un governo, d'accordo con le società informatiche di turno, cerchi di limitare la libertà di parola dei cittadini sfruttando i principi di fiducia per creare una vera e propria censura. Questo ci farebbe retrocedere indietro di almeno mezzo secolo senza accorgercene, infatti questa operazione sarebbe spacciata come protezione per le genti da minacce informatiche, nascondendo l'indole per l'epurazione della notizia. C'è da dire, comunque, che i governi dei paesi più evoluti hanno dimostrato, negli ultimi anni (anche solo per non sfigurare purtroppo), di non voler applicare la censura dell'informazione (almeno quella via internet), si spera quindi che continuino su questo trend anche con l'inserimento, nel mercato, di tecniche che permetterebbero l'occultamento di notizie senza l'apparenza di un regime “dittatoriale”.

5.4 Effetti collaterali del Trusted Computing

Purtroppo oltre a difetti, più o meno voluti, che il TC si porta dietro, se ne aggiungono di altri molto fastidiosi a cui non è possibile far fronte, almeno con le scelte implementative che sono state effettuate per stilare le specifiche.
I principali difetti sono quattro, cerchiamo di descriverli brevemente.
In primo luogo troviamo la *Perdita di Controllo*, con questa affermazione vogliamo intendere che non è possibile, appunto, avere il controllo della propria macchina. Le aziende informatiche infatti hanno la possibilità di rendere un prodotto funzionante solo se questo rientra nei canoni della configurazione di fabbrica, ciò significa che è possibile creare un device hardware che supporti solo un tipo di software di base con uno specifico firmware, non permettendo all'utente di poter scegliere indipendentemente macchina e sistema operativo. Un esempio sono gli smartphone con sistemi operativi proprietari in cui è impossibile, infatti, installare software di base scelto dall'utente. Con Perdita di Controllo, comunque, indichiamo anche quelle situazioni in cui il proprietario del sistema viene privato della padronanza dei suoi dati, si pensi a un sistema DRM che permetta una riproduzione di un file controllata (non eseguibile su più sistemi).
Altro effetto collaterale è la *difficoltà della sostituzione* (parziale o completa) di hardware e software. In riguardo al software ci siamo espressi mentre parlavamo di Lock-In, cambiare un programma diventa un operazione complessa perché è possibile che questo stesso gestisca la cifratura, in questo caso si rende necessario un ennesimo software che permetta la transizione, visto che è impossibile pensare di dare a qualcuno la mansione che consista, ad esempio, nel riscrivere (anche con un banale copia/incolla) un numero enorme di dati. Come anticipato, questo difetto è presente anche quando si necessita di cambiare lo hardware, le chiavi di sistema infatti risiedono nel Trust Platform Module e quindi per sostituire la macchina senza voler perdere dati è richiesto un processo di migrazione, questa situazione potrebbe anche essere accettabile visto che anche senza il TPM nel momento della sostituzione di una macchina si procede al backup dei dati, ma se la macchina viene sostituita per mal funzionamento? In presenza di sistemi di

sicurezza, quali il sealed storage, non sarebbe possibile collegare un disco su una nuova macchina, questo perché non potrebbe essere letto, sarebbe quindi impossibile recuperare i dati (a meno di tecniche, delle stesse aziende, complesse che comunque dovrebbero "bucare" i sistemi di sicurezza).

Successivamente, nella nostra analisi, troviamo la *Perdita dell'Anonimato*, ci riferiamo al fatto che gli utenti che intendono utilizzare un software con convalida online devono necessariamente usufruire delle chiavi presenti nel TPM, ed essendo queste univoche da chip a chip, un server sarebbe in grado di tracciare le operazioni di un ben determinato individuo. La TCG sta comunque cercando di risolvere questo problema di attestazione remota, che ha sollevato un numero di critiche abbastanza elevato, introducendo nuove apposite specifiche nel Trusted Platform Module.

Ultimo effetto collaterale considerato *defective by design* da chi si oppone alle tecnologie "Trust" è ciò che scaturisce dal memory curtaining che, come detto nel capitolo 3, non permette di accedere ad alcune zone di memoria anche se a fare la richiesta è il sistema operativo, questo potrebbe avere ripercussioni sull'implementazione di alcuni software di base, rendendo complicata perfino un'operazione di debug.

5.5 La posizione dell'Anti-Trust

Come tutti sanno l'Anti-Trust è l'organo principale atto a controllare che vengano rispettate le regole del libero mercato e quindi che sia possibile per le aziende concorrenti partecipare al commercio interessato, senza essere spazzate via da scelte delle rivali non permesse dalla legge.
Questa descrizione allora potrebbe porre il dubbio sulla legalità, o meno, del Trusted Computing. Se infatti ci soffermiamo a meditare come le tecnologie delle aziende membri possano regolare il mercato in maniera non armonica, e impari rispetto alle concorrenti, sarebbe logico pensare che l'Anti-Trust non abbia fatto ritardo nel far sentire la sua posizione. Così però non è stato, la società anti cartello ha lasciato indisturbata la TCG di operare come meglio credesse, non considerando un pericolo il fatto che l'applicazione delle nuove tecniche avrebbe potuto (e potrebbe ancora) ledere a software liberi quali distribuzioni di sistemi operativi o interi pacchetti di programmi applicativi, ma anche ad aziende minori che cercano profitto nell'ambito informatico. Basti pensare a ditte che incentrano la loro produzione in software e hardware che non sfruttano il TC, semplicemente perché lo ritengono inutile o addirittura dannoso alla libertà del proprietario.
I "maliziosi" potrebbero pensare ad un accordo tra le multinazionali e l'Autorità Garante della Concorrenza e del Mercato, questa però è una realtà che in molti si sentono di escludere, ricordando come in passato l'Antitrust si è espressa contro le ricche società degli Stati Uniti che oggi sono membri della TCG (si ricordi ad esempio la maxi-multa commissionata a Microsoft nata dalla denuncia di RealPlayer). Più che di un accordo infatti si può parlare di un vero e proprio modo di aggirare le leggi che regolamentano il libero mercato, affiancate, sfortunatamente, si potrebbe dire, dallo sviluppo di società che in passato non entravano alla "spartizione della torta" con una "gran fetta". Fornire un esempio non è cosa difficile, si pensi ad un brano musicale (un singolo per intenderci) il cui prezzo è fissato su un certo valore e proseguendo si immagini che lo stesso singolo sotto forma di file (mp3) venga venduto online su una piattaforma

proprietaria ad un prezzo più basso del tot valore, i rivenditori non annessi alla piattaforma (in pratica tutti tranne il proprietario della stessa) risentirebbero della scelta in maniera negativa e gran parte del mercato sarebbe in mano ad una sola persona giuridica. Ciò sembrerebbe dare le fondamenta per un ingresso in scena dell'Antitrust, questo però non può avvenire a causa del fatto che legalmente il prezzo non è stato fissato dal rivenditore ma dal produttore, e questi non è perseguibile poichè sta commercializzando qualcosa che non ha concorrenti che possono essere ritenuti tali (può piacere più un cantante che un altro, ma nessun di questi pubblica la stessa melodia con più produttori nello stesso tempo). Questo è il motivo principale per il quale questo metodo non è considerato un cartello, se poi pensiamo anche al fatto che le due aziende principali fornitrici di sistemi operativi e programmi sono Microsoft e Apple e che quest'ultima, oramai, può vantare un grande successo, comprendiamo come la concorrenza tra le due cancelli il problema del monopolio (purtroppo non quello dell'oligopolio). Ora si ipotizzi uno scenario in cui queste compravendite avvengano solo grazie all'utilizzo di applicazioni TC e si integri questa visione alla precedente, ci troveremo davanti ad un contesto in cui la concorrenza a queste nuove tecnologia non potrebbe neanche esistere (magari file protetti da DRM che permettano la riproduzione e la copia solo su sistemi trust), in pratica un oligopolio non creato dalla predominanza di qualcuno, non dalla estinzione di qualcun altro, ma dalla esclusiva partecipazione al mercato di chi costituisce l'oligopolio stesso.

Una situazione di queste è completamente da evitare perché se si instaurasse priverebbe il commercio della libera concorrenza e i clienti, di questo, della libera scelta.

Capitolo 6

Riflessioni e Conclusioni

In questa tesi abbiamo introdotto le minacce in cui può incorrere un utente e allo stesso tempo abbiamo cercato di descrivere alcuni metodi in grado di contrastare gli attacchi informatici.

Nel nostro percorso ci siamo focalizzati, per quel che riguarda la sicurezza, nel descrivere specifiche e modi di operare della Trusted Computing Group, una corporation dell'Oregon alla cui fondazione hanno partecipato esponenti di multinazionali per lo sviluppo hardware e software. Le specifiche di questa società hanno aperto il mondo informatico alla creazione di un nuovo modo di fare sicurezza chiamato Trusted Computing in grado di bloccare sul nascere ogni tipo di software od operazione che porterebbe ad un malfunzionamento della macchina. Abbiamo notato che questo nuovo sistema differisce totalmente dai classici impianti di difesa (quali antivirus, firewall, etc.) e crea una forte scissione da questi per due fondamentali principi. Il primo si basa sul fatto che la sicurezza non viene più fondata interamente sul software, anzi, il cuore di questa diventa una parte hardware chiamata Trusted Platform Module dal quale è possibile gestire tutte le politiche di sistema, in secondo luogo notiamo che questa pratica non è un semplice "add-on" al sistema ma ne diventa parte integrante. Proprio questa è la grande novità inserita dalla corporazione statunitense, integrare al sistema un protocollo di sicurezza non permettente a questo di operare in casi

che non garantiscano un buon livello di tranquillità, infatti la macchina non autorizza calcoli di alcun genere se questi non sono certificati.
La grande novità però si è anche dimostrata un grande vincolo proprio per questa sua natura. Il problema che è saltato all'occhio, nello sviluppo di questo testo, è il fatto che a giudicare cosa è sicuro e cosa non lo è sono le stesse multinazionali fondatrici e non più l'utente proprietario della macchina. Ciò ha aperto, nel mondo informatico, una grande discussione attorno all'opera della TCG che tende a screditarla perché ritiene che le sue tecnologie possano essere un mezzo per mettere in condizione di non-scelta un fruitore di sistemi informatici, allineando quindi tutti gli utenti verso le stesse decisioni. Questa è una visione pessimistica dello sviluppo ma non è giusto metterla a tacere, immaginiamo anche per un solo attimo che le società partecipanti alla TCG siano "tentate" dal serpente del profitto e basino le loro scelte, per la sicurezza, in modo da sbaragliare la concorrenza, si aprirebbe uno scenario in cui un cliente di un sistema non potrebbe più utilizzare uno specifico software anche se la pratica di questo non manifesterebbe alcun tipo di rischio. La diatriba ormai va avanti da almeno sette anni ed ha visto comporsi i due schieramenti rispettivamente dalle multinazionali, pro-TC, e dai sostenitori dell'informatica libera, contro-TC. È fondamentale aver notato chi, e come, si schiera nei confronti del Trusted Computing; questa nuova "scuola", se utilizzata in maniera non corretta, potrebbe portare alla morte del software libero, e questo non è un eufemismo! Nel caso in cui tutti i produttori di chip si allineassero a specifiche malevole (nei confronti della concorrenza libera) per continuare a gioire del mondo dell'informatica libera due sole sarebbero le soluzioni. La prima potrebbe essere la costruzione, da parte di aziende minori, di chip privi di TC, tuttavia queste potrebbero non essere interessate ad un mercato così piccolo, oppure si potrebbe optare per lo sviluppo di un'architettura parallela a quella in uso, anche in questo caso però si dovrebbe cercare, come un rabdomante fa con l'acqua, gli investitori che, si può immaginare, non sarebbero molti. In ogni caso non sarebbe possibile sfruttare tutte le applicazioni web e comunicare con semplicità in un mondo informatico governato da una gestione malsana del TC.
Riassumendo, il Trusted Computing è un'arma a doppio taglio (almeno per quel

che riguarda l'utente) e ciascuno di noi dovrebbe informarsi in maniera molto maniacale dei rischi che la nostra libertà di scelta potrebbe correre. La parola d'ordine per noi semplici individui è quindi: informazione.

L'**Informazione** è l'unica arma che possiamo impugnare senza aver paura che si rivolti contro di noi.

Bibliografia

[1] M. Cinotti (2002), *Internet Security*, seconda edizione, Milano, Ulrico Hoepli Editore (Hoepli Informatica).

[2] Autori di Wikipedia, *Funzione φ di Eulero*, Wikipedia, L'enciclopedia libera, http://it.wikipedia.org/wiki/Funzione_φ_di_Eulero (controllata il: aprile 19, 2011).

[3] O. Babaoglu (2001-2009), Slides tratte dal corso in *Sicurezza* (laurea in informatica), Bologna.

[4] Autori di Wikipedia, *Data Encryption Standard*, Wikipedia, L'enciclopedia libera, http://it.wikipedia.org/wiki/Data_Encryption_Standard (controllata il: aprile 20, 2011).

[5] A. Lisi, L. Giacopuzzi / M. Scialdone, F. Bertoni (2006), *Guida al Codice dell'Amministrazione Digitale*, prima edizione, Camerino (MC), Halley Editrice.

[6] M. Giuliani (2006), *Virus e Anti-virus, l'eterna lotta tra il bene e il male*, Hwupgrade (sito internet).

[7] V. Grienti (2009), *Chiesa e Web*, prima edizione, Cantalupa (TO), Afattà Editrice.

[8] A. Bottoni (2006), *Trusted Computing, Promessa o Minaccia*, E-Book.

[9] TCG (2011), *TPM Main Part 1 Design Principles*, TCG Published.

[10] Autori Microsoft (2009), *Panoramica tecnica di Crittografia unità BitLocker*, Microsoft TechNet.

[11] Autori Microsoft (2011), *Microsoft Announces Microsoft BitLocker Administration and Monitoring (MBAM)*, Microsoft Blog.

[12] ARM, *Hardware Architecture & Software Architecture*, ARM Sito Ufficiale.

[13] Liqun Chen, Moti Yung (2009), *Trusted System*, Beijing (China), Springer.

[14] Emscb (2006), *Turaya*, EMSCB Sito Ufficiale.

[15] TCG Trusted Network Connect (2009), *TNC Architecture for Interoperability*, Versione 1.4 Revisione 4, TCG Published.

[16] TCG (2011), *Trusted Multi-Tenant Infrastructure Use Cases FAQ*, TCG Sito Ufficiale.

[17] TCG (2011), *TCG Trusted Multi-Tenant Infrastructure Use Cases*, Versione 1.0 Revisione 1.0, TCG Published.

[18] Renato Iannella, Susanne Guth, Helge Hundacker, Daniel Paehler, Andreas Kasten, *Draft Specification of ODRL v 2.0 Core Model*, ODRL Initiative Sito Ufficiale.

[19] Naftaly H. Minsky (2010), *Reducing Spam via Trustworthy Self Regulation by Email Senders*, Department of Computer Science Rutgers University.

[20] Richard M. Stallman (2007), *Can You Trust Your Computer?*, Articolo.

Ringraziamenti

Desidero regalare un grazie speciale ai miei genitori, due persone che mi hanno permesso di studiare e, allo stesso tempo, non hanno mai fatto mancare il giusto incoraggiamento in grado di supportarmi. Assieme a loro, ringrazio in particolare maniera la mia ragazza Sara, che è riuscita a sopportarmi nelle occasioni in cui un "pizzico" di stress si è fatto sentire nella mia persona, facendomi rendere conto di quanto io possa essere stato fortunato ad averla incontrata. Desidero altresì manifestare la mia gratitudine verso i professori dell'Ateneo, che oltre a donare nozioni riescono a trasmettere anche un valido esempio di vita. Inoltre un pensiero va a tutti i parenti e gli amici che mi hanno sostenuto in questi ultimi anni.
Grazie.

Firma