UOGÓLNIONE STRUKTURY UPRAWNIEŃ Z HIERARCHIĄ

Andrzej Pragacz

Wydział Matematyki, Informatyki i Mechaniki
Uniwersytetu Warszawskiego (MIM UW)

Streszczenie. Struktury dostępu są używane przy zagadnieniach bezpieczeństwa związanych z sytuacjami gdzie jeden lub więcej podmiotów próbuje uzyskać pewien zasób. Przedstawimy uogólnienie struktur dostępu na przypadek wielu zasobów, co pozwala na zgrabne ujęcie schematów progowych i hierarchicznych.

Zaprezentujemy też użycie tzw. iloczynu dwuliniowego, definiowanego w grupie punktów n-torsyjnych krzywej eliptycznej nad ciałem skończonym na dwóch przykładowych hierarchicznych schematach przydzielania kluczy.

Słowa kluczowe: struktury dostępu, uprawnienia, dzielenie sekretu, schemat progowy, hierarchia, iloczyn Weila, grupy, krzywe eliptyczne, struktury monotoniczne.

1. Wprowadzenie

Kryptografia to nauka zajmująca się m.in. kontrolą dostępu do informacji. Szczególnym przykładem jest tu np. dzielenie sekretu, czyli podzielenie pewnej informacji (sekretu) między pewne podmioty w ten sposób, że tylko niektóre grupy podmiotów mogą w pewien sposób „poznać" sekret.

Innym przykładem może być przydzielanie pewnej informacji (tzw. klucza) każdemu podmiotowi, i umożliwienie określonym grupom podmiotów „poznanie" tego klucza. Przedstawiana praca dotyczy właśnie tego typu zagadnień, ze szczególnym naciskiem na aspekt hierarchiczny.

2. Ogólne strukury dostępu

2.1. Struktury monotoniczne

Definicja 2.1. Niech dany będzie porządek $\preceq$ zdefiniowany na elementach zbioru X. Wówczas **struktura monotoniczna** A względem porządku $\preceq$ to podzbiór X o własności:

$$\forall x, y \in X \quad x \in A \wedge x \preceq y \Rightarrow y \in A$$

W szczególności będziemy rozważać struktury monotoniczne będące rodzinami zbiorów, a porządek $\preceq$ będzie definiowany przez zawieranie się zbiorów (tj. $x \preceq y \Leftrightarrow x \subseteq y$).

Definicja 2.2. Niech X będzie pewnym zbiorem, a $A \subseteq X$ jego podzbiorem. Wówczas $cl(A)$ zwane **domknięciem monotonicznym** A będzie najmniejszym takim zbiorem, że $A \subseteq cl(A)$ i $cl(A)$ będzie strukturą monotoniczną.

Definicja 2.3. Jeśli $A \subseteq X$ jest strukturą monotoniczną, to zbiór $B \subseteq A$ o własności $cl(B) = A$ będzie zwany dalej **zbiorem baz** lub **rodziną zbiorów bazowych** A.

2.2. Ogólna struktura dostępu

Zdefiniujmy podstawowe pojęcie, które będzie szeroko wykorzystywane w niniejszej pracy:

Definicja 2.4. Ogólna struktura dostępu to trójka postaci $(\mathfrak{U}, \mathfrak{G}, (\mathfrak{B}_s)_{s\in\mathfrak{G}})$, gdzie $\mathfrak{U}$ to zbiór podmiotów, $\mathfrak{G}$ to zbiór sekretów, a $\mathfrak{B}_s \subseteq \mathcal{P}(\mathfrak{U})$ to rodziną takich zbiorów podmiotów, które mogą odtworzyć sekret s. Ponadto, dla każdego $s \in \mathfrak{G}$ rodzina $\mathfrak{B}_s$ zwana dalej **rodziną zbiorów dostępu** jest strukturą monotoniczną względem relacji $\subseteq$ (relacji zawierania).

Będziemy dodatkowo oznaczać przez $\mathfrak{B}'_s$ najmniejszą rodzinę zbiorów bazowych $\mathfrak{B}_s$, tj. $cl(\mathfrak{B}'_s) = \mathfrak{B}_s$.

2.3. Zbiór sekretów

Zdefiniujmy przydatne pojęcie:

Definicja 2.5. Zbiór sekretów, które może uzyskać zbiór podmiotów $U \subseteq \mathfrak{U}$, to funkcja $S : \mathcal{P}(\mathfrak{U}) \to \mathcal{P}(\mathfrak{G})$ zdefiniowana jako

$$S(U) = \{s \in \mathfrak{G} : U \in \mathfrak{B}_s\} \tag{2}$$

Zauważmy, że możemy równoważnie zapisać powyższą definicję jako:

$$S(U) = \{s \in \mathfrak{G} : \exists B \in \mathfrak{B}'_s \quad B \subseteq U\} \tag{3}$$

Inaczej mówiąc, $S(U)$ to zbiór takich sekretów, gdzie dla każdego sekretu można znaleźć w rodzinie zbiorów dostępu $\mathfrak{B}_s$ taki zbiór, który będzie podzbiorem U

Dla pojedynczego podmiotu $u \in \mathfrak{U}$ będziemy stosować skrót notacyjny $S(u) := S(\{u\})$

Fakt 2.6. *Funkcja $S(U)$ ma następujące własności:*

1. *Jeśli dla każdego $s \in \mathfrak{G}$ zachodzi $\emptyset \notin \mathfrak{B}_s$ to wówczas $S(\emptyset) = \emptyset$*
2. *Jeśli $U, V \in \mathcal{P}(\mathfrak{U})$ i $U \subseteq V$ to $S(U) \subseteq S(V)$*
3. *Dla każdego $\mathfrak{X} \subseteq \mathcal{P}(\mathfrak{U})$ zachodzi:*

$$\bigcup_{U \in \mathfrak{X}} S(U) \subseteq S(\bigcup_{U \in \mathfrak{X}} U) \tag{4}$$

Dowód.

1. Wynika wprost z definicji ($\emptyset$ nie należy do $\mathfrak{B}_s$).
2. Wynika z monotoniczności $\mathfrak{B}_s$. Jeśli zachodzi $U \in \mathfrak{B}_s$ to $V \in \mathfrak{B}_s$.
3. Prawdziwość wynika z poprzedniego punktu (dla każdego $U \in \mathfrak{X}$ zachodzi $S(U) \subseteq S(\bigcup_{U \in \mathfrak{X}} U)$). □

Definicja 2.7. $S(u)$ definiuje nam w sposób naturalny **relację częściowego porządku** $\leq_S$, $<_S$ oraz relację równoważności $\sim_S$ na podmiotach $u, u' \in \mathfrak{U}$:

$$u \leq_S u' \Leftrightarrow S(u) \subseteq S(u') \tag{5}$$

$$u <_S u' \Leftrightarrow S(u) \subsetneq S(u') \tag{6}$$

$$u \sim_S u' \Leftrightarrow S(u) = S(u') \tag{7}$$

Definicja 2.8. Relację częściowego porządku $\leq_S$, $<_S$ można rozszerzyć na podzbiory $\mathfrak{U}$ w sposób następujący:

$$U \leq_S U' \Leftrightarrow S(U) \subseteq S(U') \tag{8}$$

$$U <_S U' \Leftrightarrow S(U) \subsetneq S(U') \tag{9}$$

2.4. Przykłady

2.4.1. Schemat progowy

Schemat progowy (k, n) dzielenia sekretu opiera się na dzieleniu pojedynczego sekretu s między n podmiotów w taki sposób, że tylko k lub więcej podmiotów może uzyskać sekret s. Znanym przykładem implementacji jest klasyczny już schemat dzielenia sekretu Shamira opisany w pracy [6]

Struktura uprawnień dla takiego schematu opisuje się następująco:

- $\mathfrak{U} = \{u_1, u_2, \ldots, u_n\}$
- $\mathfrak{G} = \{s\}$
- $\mathfrak{B}_s = \{U \in \mathcal{P}(\mathfrak{U}) : \#U \geq k\}$

Łatwo zauważyć, że $S(U) = \{s\}$ wtedy i tylko wtedy, gdy $\#U \geq k$

2.4.2. Schemat hierarchiczny

Rozważmy teraz przypadek, który omówimy dokładniej ze względu na to, że będziemy się nim później szczegółowo zajmować.

Każdy podmiot u_i dysponuje swoim własnym sekretem s_i. Ponadto podmioty spełniają pewną częściową relację porządku $\preceq$, która ustala na nich hierarchię. Zachodzi $u_i \preceq u_j$ wtw. gdy u_j jest przodkiem u_i w DAGu (lub w drzewie).

- $\mathfrak{U} = \{u_1, u_2, \ldots, u_n\}$
- $\mathfrak{G} = \{s_1, s_1, \ldots, s_n\}$
- Przyjmujemy że:

$$\mathfrak{B}_{s_i} = \{U \in \mathcal{P}(\mathfrak{U}) : \exists u \in U \quad u_i \preceq u\} \tag{10}$$

Relacja $\preceq$ (definiująca) jest równoważna relacji $\leq_S$ (indukowanej przez wcześniej zdefiniowaną funkcję S). Dowód można znaleźć w dodatku A.

2.4.3. Schemat mieszany

Na koniec możemy omówić bardziej skomplikowany przypadek. Sekret s_1 jest dzielony na udziały między podmioty u_1, u_2, zaś s_2 jest dzielony na udziały między podmioty u_2, u_3. Ponadto podmiot u_4 ma dostęp do obu sekretów, a podmiot u_5, który jest przodkiem u_4, ma także dostęp do sekretu s_3.

Struktura uprawnień dla takiego przypadku wygląda następująco:

- $\mathfrak{U} = \{u_1, u_2, u_3, u_4, u_5\}$
- $\mathfrak{G} = \{s_1, s_2, s_3\}$
- $\mathfrak{B}_{s_1} =$
  $\{\{u_5\}, \{u_4\}, \{u_1, u_2\}, \{u_4, u_1, u_2\}, \{u_5, u_4\}, \{u_5, u_1, u_2\}, \{u_5, u_4, u_1, u_2\}\}$
- $\mathfrak{B}_{s_2} =$
  $\{\{u_5\}, \{u_4\}, \{u_2, u_3\}, \{u_4, u_2, u_3\}, \{u_5, u_4\}, \{u_5, u_2, u_3\}, \{u_5, u_4, u_2, u_3\}\}$
- $\mathfrak{B}_{s_3} = \{\{u_5\}\}$

2.5. Bezpieczeństwo systemu hierarchicznego

Nasuwa się pytanie, w jaki sposób można badać bezpieczeństwo hierarchicznych schematów przydzielania kluczy. Najczęściej używanym sposobem są gry między tzw. wyzywającym i tzw. atakującym. Poniżej zdefiniowano taką właśnie grę:

Definicja 2.9. Schemat przydzielania kluczy jest bezpieczny względem **odzyskiwania klucza** (ang. Key Recovery, za [1]), jeśli nie istnieje działający w czasie wielomianowym atakujący (ang. Adversary) $\mathcal{A}$, który ma niezaniedbywalną przewagę (ang. Advantage) w grze o następujących fazach:

1. **Setup**: Wyzywający (ang. Challenger) wykonuje $Setup(1^k, G)$, gdzie $G = \langle V, E\rangle$ jest skierowanym grafem acyklicznym i przekazuje całą informację publiczną Pub atakującemu.
2. **Attack**: Atakujący dokonuje zapytania $Corrupt(v_i)$ do wyzywającego, na które tenże odpowiada atakującemu, zwracając sekretną informację wierzchołka v_i czyli $Sec(v_i)$.
3. **Break**: Atakujący zwraca wierzchołek v^* wraz z odgadniętym kluczem prywatnym $Pr(v^*)'$. Wierzchołek v^* musi spełniać warunek: dla każdego v_i zachodzi $v^* \notin Desc(v_i)$ (nie należy do zbioru potomków)

Przewagę w tej grze, dalej zwaną **grą $\boldsymbol{KR}$**, definiujemy jako:

$$Adv_{\mathcal{A}}^{\boldsymbol{KR}} = \mathbb{P}(Pr(v^*)' = Pr(v^*))$$

Definicja 2.10. Schemat przydzielania kluczy jest bezpieczny względem **odzyskiwania klucza ustalonego wierzchołka** v, jeśli nie istnieje działający w czasie wielomianowym atakujący $\mathcal{A}$, który ma niezaniedbywalną przewagę w grze opisanej wyżej z następującą modyfikacją, że atakujący zwraca w fazie **Break** wierzchołek $v^* = v$. Grę tę będziemy nazywać dalej **grą $\boldsymbol{SKR_v}$**.

Innymi słowy, wierzchołek v^* zwrócony w fazie Break jest już znany w fazie Setup i wyzywający może wykorzystać tę informację.

3. Zastosowania iloczynu dwuliniowego

3.1. Definicja iloczynu dwuliniowego

Poniższą definicję przytaczamy za [2], [8].

Definicja 3.1. Niech będą dane grupy cykliczne G_1 i G_2, obie tego samego rzędu q. W przypadku grupy G_1 będziemy stosować notację addytywną, a w przypadku G_2 notację multiplikatywną.

Iloczyn dwuliniowy na grupie G_1 o wartościach w grupie G_2 definiujemy jako funkcję $G_1 \times G_1 \to G_2$ o następujących własnościach:

1. $\forall a, b \in \mathbb{Z} \quad \forall P, Q \in G_1 \quad \hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$ (dwuliniowość)
2. dla każdego P będącego generatorem $\hat{e}(P, P) \neq 1$ (niezdegenerowalność)
3. $\hat{e}$ jest efektywnie obliczalna

Konstrukcję takiego iloczynu z użyciem tzw. iloczynu Weila, definiowanym w grupie punktów n-torsyjnych krzywej eliptycznej nad ciałem skończonym można znaleźć w [2], [8].

Jednym z najbardziej znanych przykładów użycia jest schemat Boneha-Franklina ([2]), będący schematem asymetrycznego szyfrowania.

3.2. Dwuliniowy Problem Diffiego-Hellmana i jego warianty

Definicja 3.2. Generator parametrów BDH:
Niech k będzie parametrem bezpieczeństwa. Wówczas niech $\mathcal{G}$ będzie taką funkcją, że:

$$\mathcal{G}(1^k) = \langle q, G_1, G_2, \hat{e} \rangle$$

gdzie G_1 jest grupą cykliczną, G_2 jest grupą cykliczną, obie są rzędu q, natomiast $\hat{e} : G_1 \times G_1 \to G_2$ jest iloczynem dwuliniowym zdefiniowanym na tych grupach.

Należy rozumieć tutaj, że $\mathcal{G}(1^k)$ generuje nam „opis" grup G_1, G_2 oraz iloczynu dwuliniowego $\hat{e}$, który ma rozmiar wielomianowy względem k (czyli nie są np. generowane wszystkie elementy grupy G_1). „Opis" grup pozwala na wyznaczenie w czasie wielomianowym przykładowych generatorów, zaś „opis" iloczynu $\hat{e}$ pozwala na obliczenie $\hat{e}$ w czasie wielomianowym. k jest przekazywane do $\mathcal{G}$ jako ciąg k jedynek. dzięki tej sztuczce notacyjnej można powiedzieć, że $\mathcal{G}$ jest wielomianowy względem k, gdyż rozmiar danych wejściowych to właśnie k.

Definicja 3.3. Obliczeniowy dwuliniowy problem Diffiego-Hellmana ($BCDH$):
Niech będą dane cykliczne grupy G_1 i G_2, obie rzędu q. Ponadto niech będzie zdefiniowany iloczyn dwuliniowy $\hat{e}$ na G_1 i G_2. Dla zadanego generatora $P \in G_1$ i losowych $a, b, c \in \mathbb{Z}_q^*$ oblicz $\hat{e}(P, P)^{abc}$ na podstawie

wyłącznie P, aP, bP i cP. Innymi słowami:

$$BCDH_{G_1,G_2,\hat{e}}(P, aP, bP, cP) = \hat{e}(P,P)^{abc}$$

Definicja 3.4. Obliczeniowy dwuliniowy kwadratowy problem Diffiego-Hellmana ($BSCDH$):
Niech będą dane cykliczne grupy G_1 i G_2, obie rzędu q. Ponadto niech będzie zdefiniowany iloczyn dwuliniowy $\hat{e}$ na G_1 i G_2. Dla zadanego generatora $P \in G_1$ i losowych $a, b \in \mathbb{Z}_q^*$ oblicz $\hat{e}(P,P)^{a^2 b}$ na podstawie wyłącznie P, aP i bP. Innymi słowami:

$$BCDH_{G_1,G_2,\hat{e}}(P, aP, bP) = \hat{e}(P,P)^{a^2 b}$$

Definicja 3.5. Obliczeniowy dwuliniowy odwrócony problem Diffiego-Hellmana ($BICDH$):
Niech będą dane cykliczne grupy G_1 i G_2, obie rzędu q. Ponadto niech będzie zdefiniowany iloczyn dwuliniowy $\hat{e}$ na G_1 i G_2. Dla zadanego generatora $P \in G_1$ i losowych $a, b \in \mathbb{Z}_q^*$ oblicz $\hat{e}(P,P)^{a^{-1} b}$ na podstawie wyłącznie P, aP i bP Innymi słowami:

$$BCDH_{G_1,G_2,\hat{e}}(P, aP, bP) = \hat{e}(P,P)^{a^{-1} b}$$

Twierdzenie 3.6. *Problemy $BCDH$, $BSCDH$, $BICDH$ są obliczeniowo równoważne.*

Dowód tego faktu można znaleźć w dodatku B.

Definicja 3.7. Decyzyjny dwuliniowy problem Diffiego-Hellmana ($BDDH$):
Niech będą dane cykliczne grupy G_1 i G_2, obie rzędu q. Ponadto niech będzie zdefiniowany iloczyn dwuliniowy $\hat{e}$ na G_1 i G_2. Dla zadanego generatora $P \in G_1$ i losowych $a, b, c, d \in \mathbb{Z}_q^*$ zadecyduj, czy $\hat{e}(P,P)^d = \hat{e}(P,P)^{abc}$, na podstawie wyłącznie P, aP, bP, cP i $\hat{e}(P,P)^d$. Innymi słowami:

$$BDDH_{G_1,G_2,\hat{e}}(P, aP, bP, cP, \hat{e}(P,P)^d) = \begin{cases} 1 & \text{gdy } \hat{e}(P,P)^{abc} = \hat{e}(P,P)^d \\ 0 & \text{w p.p.} \end{cases}$$

Definicja 3.8. Dwuliniowe założenie Diffiego-Hellmana (ang. Bilinear Diffie-Hellman, BDH):

Dwuliniowy problem Diffiego-Hellmana (obliczenie $\hat{e}(P,P)^{abc}$ na podstawie wyłącznie P, aP, bP i cP) jest trudny.

Formalnie mówiąc, zdefiniujmy przewagę (ang. advantage) dla algorytmu $\mathcal{A}$ rozwiązującego problem $BCDH$.

$$Adv_{\mathcal{A}}^{BCDH}(k) = \mathbb{P}\left(\mathcal{A}_{G_1,G_2,\hat{e}}(P,aP,bP,cP) \middle| \begin{array}{l} \langle q, G_1, G_2, \hat{e}\rangle \leftarrow \mathcal{G}(1^k) \\ a,b,c \leftarrow \mathbb{Z}_q^*, P \leftarrow G_1 \end{array}\right)$$

Wówczas przewaga

$$\epsilon_{BCDH} = Adv^{BCDH}(k) = \max_{\mathcal{A}} Adv_{\mathcal{A}}^{BCDH}(k)$$

jest zaniedbywalna.

Definicja 3.9. Decyzyjne dwuliniowe założenie Diffiego-Hellmana (ang. Decisional Bilinear Diffie-Hellman, DBDH):
Decyzyjny dwuliniowy problem Diffiego-Hellmana (rozstrzygnięcie $\hat{e}(P,P)^d = \hat{e}(P,P)^{abc}$ na podstawie wyłącznie P, aP, bP, cP i $\hat{e}(P,P)^d$) jest trudny.

Formalnie mówiąc, zdefiniujmy przewagę (ang. advantage) dla algorytmu $\mathcal{A}$ rozwiązującego problem $BDDH$.

$$\rho_1 = \mathbb{P}\Bigg(\mathcal{A}_{G_1,G_2,\hat{e}}(P,aP,bP,cP,\hat{e}(P,P)^d) = 1 \Bigg| \begin{array}{l} \langle q, G_1, G_2, \hat{e}\rangle \leftarrow \mathcal{G}(1^k) \\ a,b,c \leftarrow_R \mathbb{Z}_q^*, d = abc, P \leftarrow G_1 \end{array}\Bigg)$$

$$\rho_0 = \mathbb{P}\left(\mathcal{A}_{G_1,G_2,\hat{e}}(P,aP,bP,cP,\hat{e}(P,P)^d) = 1 \middle| \begin{array}{l} \langle q, G_1, G_2, \hat{e}\rangle \leftarrow \mathcal{G}(1^k) \\ a,b,c,d \leftarrow_R \mathbb{Z}_q^*, P \leftarrow G_1 \end{array}\right)$$

$$Adv_{\mathcal{A}}^{BDDH}(k) = |\rho_0 - \rho_1|$$

Wówczas przewaga

$$\epsilon_{BDDH} = Adv^{BDDH}(k) = \max_{\mathcal{A}} Adv_{\mathcal{A}}^{BDDH}(k)$$

jest zaniedbywalna.

3.3. Schemat pierwszy: Liu et al.

W tym systemie [4] zakładamy, że każdy podmiot jest wierzchołkiem DAGu. Ten system opiera się na użyciu iloczynu dwuliniowego $\hat{e} : G_1 \times G_1 \rightarrow G_2$, ale wykorzystuje go w inny sposób (można odnaleźć pewne podobieństwo do schematu Boneha-Franklina [2]).

Każdy „zasób" t (w naszej terminologii sekret) ma swój klucz prywatny $DK_t \in \mathbb{N}$.

Dostęp do niego mogą uzyskać podmioty S_i. Każdy podmiot jest identyfikowany przez publiczne $Q_{S_i} = H_1(ID_{S_i})$

Niech będą dane funkcje haszujące $H_1 : \{0,1\}^* \rightarrow G_1$, $H_2 : G_2 \rightarrow \{0,1\}^*$. Podczas tworzenia kluczy wybierane jest $\alpha \in \mathbb{Z}_q^*$, $r \in_R Z_q^*$, gdzie $q = |G_1|$. Następnie upubliczniane jest $P_0 \in G_1$, $U = rP_0$, $P_{pub} = \alpha P_0$. Dla każdego podmiotu obliczany jest jego prywatny klucz $D_{S_i} = \alpha Q_{S_i}$. Dla każdego zasobu jest generowana publiczna funkcja:

$$F_{DK_t} = DK_t \oplus \left\{ \prod_{S_j \,\mathrm{ma\,dostęp\,do}\, t} (x \oplus H_2(g_{S_j}^r)) \right\} \quad g_{S_j} = \hat{e}(Q_{S_j}, P_{pub})$$

Obliczenie klucza DK_t odbywa się następująco:

$$DK_t = F_{DK_t}(H_2(\hat{e}(D_{S_i}, U)))$$

Wystarczy bowiem, że tylko jedno z wyrażeń $x \oplus H_2(g_{S_j}^r)$ się wyzeruje, co spowoduje wyzerowanie całego iloczynu a w konsekwencji:

$$F_{DK_t}(H_2(\hat{e}(D_{S_i}, U))) = DK_t \oplus 0 = DK_t$$

Dowód bezpieczeństwa można znaleźć w [4].

3.4. Schemat drugi

W tym przypadku ograniczamy się do hierarchii drzewiastej (każdy podmiot z wyjątkiem jednego ma swój podmiot nadrzędny, czyli rodzica).

Niech 1^k będzie parametrem bezpieczeństwa. Dane jest drzewo $T = \langle V, E \rangle$ reprezentujące hierarchię podmiotów (każdemu podmiotowi odpowiada wierzchołek).

Procedura $Setup(1^k, T)$:

1. oblicza $\mathcal{G}(1^k) = \langle q, G_1, G_2, \hat{e} \rangle$

2. wyznacza dwa losowe generatory $Q, P \in_R G_1$ o tej własności, że $\hat{e}(Q, P)$ nie jest elementem neutralnym G_2.
3. upublicznia informację: $q, G_1, G_2, \hat{e}, Q, P$
4. Dla każdego wierzchołka v w porządku preorder:
   (a) wybiera losowo $s_v \in_R \mathbb{Z}_q^*$
   (b) wylicza ukrytą informację

$$Sec(v) = \begin{cases} s_v Q & \text{dla korzenia} \\ s_v Sec(u) & \text{gdy rodzicem } v \text{ jest } u \end{cases}$$

   (c) Ukryta informacja $Sec(v)$ jest przekazywana do wierzchołka v przez zaufany kanał.
5. Dla każdej pary v, u dla której $v \prec u$, w porządku od najkrótszej odległości między v i u do najdłuższej:
   (a) oblicza kluczy publiczny (z użyciem wcześniej wylosowanego s_v):

$$Pub(v, u) = \begin{cases} s_v P & \text{gdy } u \text{ jest rodzicem } v \\ s_v Pub(w, v) & \text{gdy pewne } w \prec u \text{ jest rodzicem } v \end{cases}$$

   (b) upublicznia $Pub(v, u)$

W powyższym schemacie klucz prywatny efektywnie może obliczyć jedynie wierzchołek v bądź wierzchołek u będący przodkiem v.

Klucz prywatny dla v to

$$Pr(v) = DeriveKey(v, v) = \hat{e}(Sec(v), P)$$

W przypadku, jeśli u będący przodkiem v chce obliczyć klucz prywatny v:

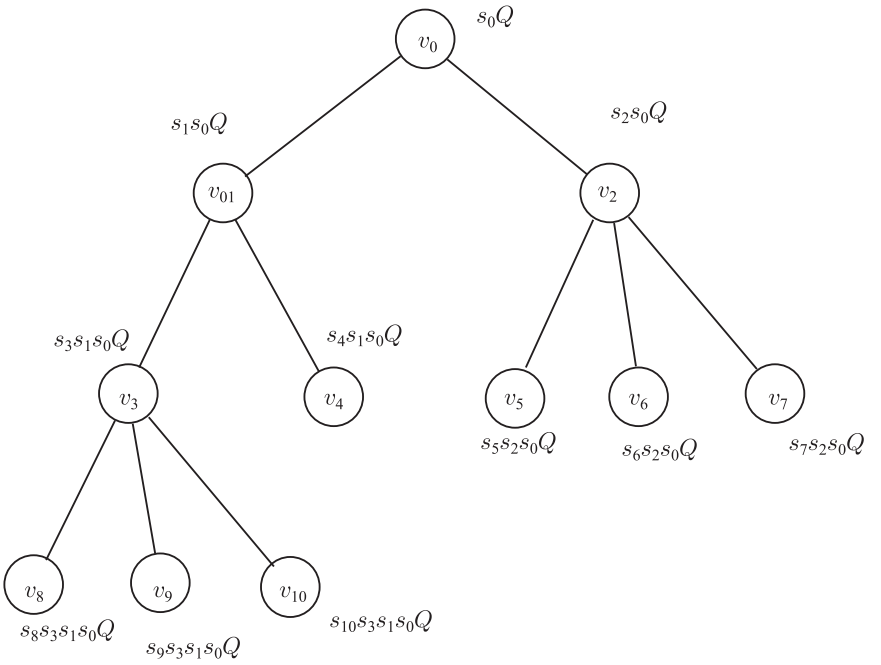
$$Pr(v) = DeriveKey(v, u) = \hat{e}(Sec(u), Pub(v, u))$$

3.4.1. Przykład

Aby uczynić powyższą definicję schematu przydzielania kluczy czytelniejszą, zaprezentujmy ją na przykładowym drzewie.

Na tym drzewie wykonujemy opisaną wcześniej procedurę $Setup(1^k, T)$, gdzie 1^k będzie parametrem bezpieczeństwa. W wyniku jej każdy z wierzchołków otrzymuje swoją informację ukrytą, co zostało zaprezentowane na rysunku 1.

Ponadto w repozytorium kluczy publicznych zostają umieszczone klucze publiczne, ukazane na rysunku 2. Każdy z tych kluczy jest dostępny dla każdego podmiotu.



Rysunek 1. Drzewo T z przypisaną do każdego wierzchołka jego informacją ukrytą. Dla uproszczenia przyjęto, że $s_i = s_{v_i}$. Należy zwrócić uwagę, że dana informacja ukryta jest znana tylko danemu wierzchołkowi

Zobrazujmy obliczanie kluczy na przykładzie. Przyjmijmy, że wierzchołek v_1 chce obliczyć klucz prywatny wierzchołka v_9. Wówczas bierze swoją informacje ukrytą $Sec(v_1) = s_1s_0Q$ oraz klucz publiczny $Pub(v_9, v_1) = s_9s_3P$ i oblicza:

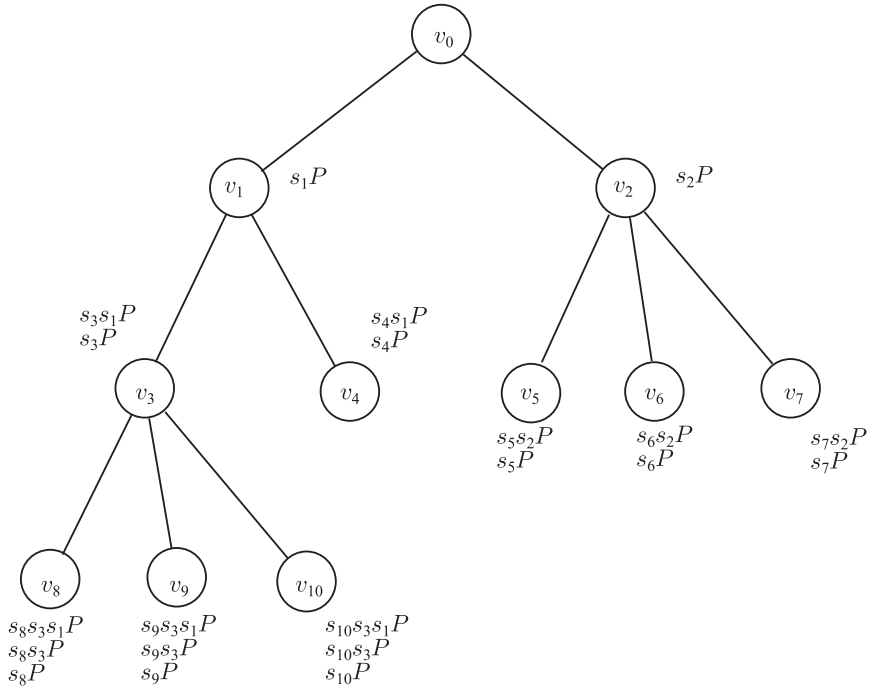
$$\hat{e}(Sec(v_1), Pub(v_9, v_1)) = \hat{e}(s_1s_0Q, s_9s_3P) = \hat{e}(Q, P)^{s_9s_3s_1s_0} = Pr(v_9)$$

3.4.2. Bezpieczeństwo

Twierdzenie 3.10. *Schemat zaproponowany w niniejsze pracy jest bezpieczny względem odzyskiwania klucza (**KR**), jeśli decyzyjne dwuliniowe założenie Diffiego-Hellmana jest prawdziwe.*

Przedstawimy tu jedynie schemat dowodu, wzorowany na pracy [1].

Posłużymy się najpierw grą $\boldsymbol{SKR}_{\boldsymbol{v}*}$ w celu udowodnienia bezpieczeństwa wyżej opisanego schematu.



Rysunek 2. Drzewo T z kluczami publicznymi wierzchołków. Klucze publiczne, które są używane do obliczania klucza prywatnego danego wierzchołka zostały umieszczone obok tego wierzchołka. Przykładowo, $Pub(v_9, v_1) = s_9 s_3 P$. Dla uproszczenia przyjęto że $s_i = s_{v_i}$

Dla uproszczenia przyjmujemy że $q, G_1, G_2, \hat{e}, Q, P$ są już ustalone i że $Q = P$.

Niech v' będzie pewnym wierzchołkiem z T.

Niech $s_{v,u} \in_R \mathbb{Z}_q^*$ będzie losowo wybrany dla każdego $v \prec u$.

Zdefiniujmy $RandomSetup(1^k, T, v')$ analogicznie do $Setup(1^k, T)$ z tą różnicą że $Sec(v') = \perp$ (jest nie zdefiniowany) oraz:

$$Pub(v,u) = \begin{cases} s_{v,u}P & \text{gdy rodzicem } v \text{ jest } u = v' \\ s_{v,w}Pub(w,v) & \text{gdy pewne } v' = w \prec u \text{ jest rodzicem } v \\ s_v P & \text{gdy rodzicem } v \text{ jest } u \neq v' \\ s_v Pub(w,v) & \text{gdy pewne } v' \neq w \prec u \text{ jest rodzicem } v \end{cases}$$

Zdefiniujmy $BDHSetup(1^k, T, v', aP, bP)$ analogicznie do $Setup(1^k, T)$ z tą

różnicą że:

$$Sec(v) = \begin{cases} \perp & \text{dla } v = v' \\ s_v Q & \text{dla korzenia} \\ s_v bP & \text{gdy rodzicem } v \text{ jest } v' \\ s_v Sec(u) & \text{gdy rodzicem } v \text{ jest } u \neq v' \end{cases}$$

$$Pub(v,u) = \begin{cases} s_v aP & \text{gdy rodzicem } v \text{ jest } u = v' \\ s_{v,w} Pub(w,v) & \text{gdy pewne } v' = w \prec u \text{ jest rodzicem } v \\ s_v P & \text{gdy rodzicem } v \text{ jest } u \neq v' \\ s_v Pub(w,v) & \text{gdy pewne } v' \neq w \prec u \text{ jest rodzicem } v \end{cases}$$

Łatwo zauważyć, że jeśli użyjemy $BDHSetup$ zamiast $Setup$ do inicjalizacji systemu, to $Sec(v') = a^{-1}bP$ i w konsekwencji $Pr(v') = \hat{e}(P,P)^{a^{-1}b}$.

Zdefiniujemy dwie gry, $\hat{G}$ oraz $\hat{G}'$ w następujący sposób:

- Gra $\hat{G}_0$: Gra $\boldsymbol{SKR}_{v*}$, gdzie w fazie Setup jest wykonywana procedura $Setup(1^k, T)$
- Gra $\hat{G}_1$: Gra $\boldsymbol{SKR}_{v*}$, gdzie w fazie Setup jest wykonywana procedura $RandomSetup(1^k, T, r)$ gdzie r to korzeń T.

Będziemy oznaczać przez T_j zdarzenie takie że $Pr(v^*) = Pr(v^*)'$ w grze $\hat{G}_j$.

Lemat 3.11. *Niech ϵ_{BDDH} będzie zdefiniowany tak jak w (12). Wówczas:*

$$|\mathbb{P}(T_0) - \mathbb{P}(T_1)| \leq \epsilon_{BDDH}$$

Dowód. Niech będzie dany algorytm $\mathcal{A}$, który jest w stanie rozróżnić między grą $\hat{G}_0$ i $\hat{G}_1$. Wówczas skonstruujemy algorytm $\mathcal{B}$ taki, że jest w stanie rozróżnić, czy dla zadanych parametrów P, aP, $bP, \hat{e}(P,P)^c$ będzie potrafił zadecydować czy $\hat{e}(P,P)^c = \hat{e}(P,P)^{a^{-1}b}$. W konsekwencji, korzystając z faktu 3.6, możemy zbudować algorytm $\mathcal{B}'$ taki że dla zadanych parametrów P, $a'P$, $b'P, c'P, \hat{e}(P,P)^{d'}$ będzie potrafił zadecydować, czy $\hat{e}(P,P)^{d'} = \hat{e}(P,P)^{a'b'c'}$.

Przeprowadźmy zatem konstrukcję $\mathcal{B}$. Algorytm otrzymuje na wejściu parametry „środowiskowe" $G_1, G_2, \hat{e}$ oraz właściwe parametry $x, y, z \in G_1$, $w \in G_2$ (w domyśle $x = P$, $y = aP$, $z = bP$, $w = \hat{e}(P,P)^c$). Algorytm ten będzie „interpolować" zachowanie między grą $\hat{G}_0$ i grą $\hat{G}_1$ w następujący sposób:

1. **Setup**:
   Wyzywający:

(a) wykonuje procedurę $BDHSetup(1^k, T, r, y, z)$
(b) ustala $Pr(r) = w$
(c) przekazuje atakującemu informację publiczną:

$$\langle q, G_1, G_2, \hat{e}, x, x, Pub \rangle$$

2. **Attack**:
   Algorytm $\mathcal{A}$ jako atakujący wykonuje zapytania $Corrupt(v_i)$. Wyzywający odpowiada, zwracając mu informacją ukrytą $Sec(v_i)$. Zauważmy, że $\mathcal{A}$ nie może pytać o $v \in Anc(v^*)$, w szczególności o r.
3. **Break**:
   Algorytm $\mathcal{A}$ zwraca $\langle v^*, Pr(v^*)' \rangle$

Po zakończeniu fazy Break algorytm $\mathcal{B}$ oblicza $Pr(v^*)$ (np. na podstawie wcześniej wyznaczonego $Sec(v^*)$, lub jeśli $v^* = r$, to $Pr(v^*) = w$).

Następnie $\mathcal{B}$ zwraca 1 jeśli $Pr(v^*)' = Pr(v^*)$ a 0 w przeciwnym przypadku. Stąd:

$$\begin{aligned}
\epsilon_{BDDH} &\geq Adv_{\mathcal{B}'}^{BDDH} \\
&= |\mathbb{P}(\mathcal{B}' \text{ zwrócił } 1 | d' \text{ jest losowe }) - \mathbb{P}(\mathcal{B}' \text{ zwrócił } 1 | d' = a'b'c')| \\
&= |\mathbb{P}(\mathcal{B} \text{ zwrócił } 1 | c \text{ jest losowe }) - \mathbb{P}(\mathcal{B} \text{ zwrócił } 1 | c = a^{-1}b)| \\
&= |\mathbb{P}(T_1) - \mathbb{P}(T_0)|
\end{aligned}$$

Dalej dowód przebiega analogicznie jak w [1].

4. Podsumowanie

W niniejszej pracy zdefiniowaliśmy ogólną strukturę uprawnień oraz opisaliśmy przykładowe rodzaje takich struktur: progowe, hierarchiczne i mieszane.

Opisaliśmy również dwa przykładowe systemy przydzielania kluczy (jeden oparty na pracy Liu et al., drugi autorski) oparte na iloczynie dwuliniowym definiowanym w grupie punktów n-torsyjnych krzywej eliptycznej nad ciałem skończonym.

A. Dowód równoważności relacji $\preceq$ i $\leq_S$

Pokażemy teraz, że relacja $\preceq$ (definiująca) jest równoważna relacji $\leq_S$ (indukowanej przez wcześniej zdefiniowaną funkcję S).

Niech $G = \langle V, E\rangle$ będzie DAGiem odpowiadającym relacji $\preceq$. Oznaczymy dalej **Zbiór przodków**:

$$A_u^{\preceq} = Anc(u) = \{v \in \mathfrak{U} : u \preceq v\}$$

Będziemy dalej pomijać $\preceq$ w zapisie (tj. $A_u = A_u^{\preceq}$)

Lemat A.1. *Zachodzi równoważność:* $A_y \subseteq A_x \Leftrightarrow y \in A_x$

Dowód. ($\Rightarrow$) wynika natychmiast z definicji.
($\Leftarrow$) Dowód przez sprzeczność. Niech $A_y \not\subseteq A_x$. Wówczas zachodzi jeden z wymienionych przypadków:

- $A_x \subsetneq A_y$. Jest to równoważne $y \prec x$, skąd mamy że $y \notin A_x$, co daje pożądaną sprzeczność.
- x i y są względem siebie nieporównywalne ($x \not\preceq y$ i $y \not\preceq x$). Wówczas mamy dwa przypadki:
  - Istnieje takie w będące najmniejszym wspólnym przodkiem ($x \prec w$ oraz $y \prec w$) i zachodzi $A_w = A_x \cap A_y$. Z poprzedniego podpunktu wiemy, że $y \notin A_w$, co w konsekwencji daje nam że $y \notin A_x$.
  - Nie istnieje żaden wspólny przodek x i y, stąd A_x i A_y są rozłączne, co dowodzi $y \notin A_x$.

Lemat A.2. *Niech $\mathfrak{B}_{s_i}$ będzie zdefiniowana jak w równaniu (10). Wówczas dla każdego $u_i, u_j \in \mathfrak{U}$:*

$$u_i \preceq u_j \Leftrightarrow \mathfrak{B}_{s_j} \subseteq \mathfrak{B}_{s_i}$$

Dowód. Możemy zdefiniować $\mathfrak{B}_{s_i}$ jako:

$$\mathfrak{B}_{s_i} = \{U \subseteq \mathfrak{U} : U \cap A_{u_i}^{\preceq} \neq \emptyset\}$$

Wówczas łatwo zauważyć, że:

$$u_i \preceq u_j \Leftrightarrow A_{u_j} \subseteq A_{u_i} \Leftrightarrow \mathfrak{B}_{s_j} \subseteq \mathfrak{B}_{s_i}$$

Lemat A.3. *Niech $\mathfrak{B}_{s_i}$ będzie zdefiniowana jak w równaniu (10). Wówczas dla każdegou$_i, u_j \in \mathfrak{U}$:*

$$\mathfrak{B}_{s_j} \subseteq \mathfrak{B}_{s_i} \Leftrightarrow \forall u_k \in \mathfrak{U} \quad \{u_i\} \in \mathfrak{B}_{s_k} \Rightarrow \{u_j\} \in \mathfrak{B}_{s_k}$$

Dowód. ($\Rightarrow$) niech $\{u_i\} \in \mathfrak{B}_{s_k}$. Wówczas $\mathfrak{B}_{s_i} \subseteq \mathfrak{B}_{s_k}$, czyli $\mathfrak{B}_{s_j} \subseteq \mathfrak{B}_{s_i} \subseteq \mathfrak{B}_{s_k}$, co daje nam ostatecznie $\{u_j\} \in \mathfrak{B}_{s_k}$ ($\Leftarrow$) Dowód przez sprzeczność. Załóżmy, że $\mathfrak{B}_{s_j} \not\subseteq \mathfrak{B}_{s_i}$. Jest to równoważne $A_{u_j} \not\subseteq A_{u_i}$ i w konsekwencji również $u_j \notin A_{u_i}$ (korzystamy tutaj z lematu A.1). W konsekwencji $\{u_j\} \notin \mathfrak{B}_{s_i}$. Jednak ponieważ $\{u_i\} \in \mathfrak{B}_{s_i}$, stąd z prawej strony równoważności (tezy) mamy, że $\{u_j\} \in \mathfrak{B}_{s_i}$, co prowadzi nas do pożądanej sprzeczności.

Fakt A.4. *Relacja $\preceq$ (definiująca) jest równoważna relacji $\leq_S$ (indukowanej przez wcześniej zdefiniowaną funkcję S), tj.*

$$u_i \preceq u_j \Leftrightarrow u_i \leq_S u_j$$

Dowód.

$$\begin{aligned} u_i \preceq u_j \Leftrightarrow\ & \text{(korzystamy z lematu A.2)} \\ \Leftrightarrow\ & \mathfrak{B}_{u_j} \subseteq \mathfrak{B}_{u_i} \\ \\ \Leftrightarrow\ & \text{(korzystamy z lematu A.3)} \\ \Leftrightarrow\ & \forall u_k \in \mathfrak{U} \quad \{u_i\} \in \mathfrak{B}_{s_k} \Rightarrow \{u_j\} \in \mathfrak{B}_{s_k} \\ \\ \Leftrightarrow\ & \{s \in \mathfrak{G} : \{u_i\} \in \mathfrak{B}_s\} \subseteq \{s \in \mathfrak{G} : \{u_j\} \in \mathfrak{B}_s\} \\ \Leftrightarrow\ & S(\{u_i\}) \subseteq S(\{u_j\}) \\ \Leftrightarrow\ & u_i \leq_S u_j \end{aligned}$$

B. Dowód równoważności problemów $BCDH$ i $BICDH$

Przystąpimy teraz do pokazania zależności między wyżej opisanymi problemami:

Twierdzenie B.1. $BSCDH_{G_1,G_2,\hat{e}} =_P BCDH_{G_1,G_2,\hat{e}}$

Dowód. Pokażemy, że powyższe problemy są równoważne poprzez wielomianowe redukcje.

- $BSCDH_{G_1,G_2,\hat{e}} \leq_P BCDH_{G_1,G_2,\hat{e}}$

$$\begin{aligned} BCDH_{G_1,G_2,\hat{e}}(P, aP, aP, bP) &= \hat{e}(P,P)^{a^2 b} \\ &= BSCDH_{G_1,G_2,\hat{e}}(P, aP, bP) \end{aligned}$$

- $BCDH_{G_1,G_2,\hat{e}} \leq_P BSCDH_{G_1,G_2,\hat{e}}$
  Niech:

$$\begin{aligned} x &= BSCDH_{G_1,G_2,\hat{e}}(2P, aP, cP) \\ &= BSCDH_{G_1,G_2,\hat{e}}(2P, \frac{a}{2}(2P), \frac{c}{2}(2P)) \\ &= \hat{e}(2P, 2P)^{\frac{a^2}{4}\cdot\frac{c}{2}} \\ &= \hat{e}(P, P)^{\frac{1}{2}a^2c} \end{aligned}$$

$$\begin{aligned} y &= BSCDH_{G_1,G_2,\hat{e}}(2P, bP, cP) \\ &= \hat{e}(P, P)^{\frac{1}{2}b^2c} \end{aligned}$$

$$\begin{aligned} z &= BSCDH_{G_1,G_2,\hat{e}}(2P, aP + bP, cP) \\ &= \hat{e}(P, P)^{\frac{1}{2}(a+b)^2c} \\ &= \hat{e}(P, P)^{\frac{1}{2}a^2c+abc+\frac{1}{2}b^2c} \end{aligned}$$

  wówczas:

$$BCDH_{G_1,G_2,\hat{e}} = z(xy)^{-1}$$

Twierdzenie B.2. $BICDH_{G_1,G_2,\hat{e}} =_P BSCDH_{G_1,G_2,\hat{e}}$

Dowód. Pokażemy, że powyższe problemy są równoważne poprzez wielomianowe redukcje.

- $BICDH_{G_1,G_2,\hat{e}} \leq_P BSCDH_{G_1,G_2,\hat{e}}$

$$\begin{aligned} BSCDH_{G_1,G_2,\hat{e}}(aP, P, bP) &= BSCDH_{G_1,G_2,\hat{e}}(aP, \frac{1}{a}(aP), \frac{b}{a}(aP)) \\ &= \hat{e}(aP, aP)^{\frac{1}{a^2}\frac{b}{a}} \\ &= \hat{e}(P, P)^{\frac{1}{a}b} \\ &= BICDH_{G_1,G_2,\hat{e}}(P, aP, bP) \end{aligned}$$

- $BSCDH_{G_1,G_2,\hat{e}} \leq_P BICDH_{G_1,G_2,\hat{e}}$

$$\begin{aligned} BICDH_{G_1,G_2,\hat{e}}(aP, P, bP) &= BICDH_{G_1,G_2,\hat{e}}(aP, \frac{1}{a}(aP), \frac{b}{a}(aP)) \\ &= \hat{e}(aP, aP)^{a\frac{b}{a}} \\ &= \hat{e}(aP, aP)^{b} \\ &= \hat{e}(P, P)^{a^2b} \\ &= BSCDH_{G_1,G_2,\hat{e}}(P, aP, bP) \end{aligned}$$

Wniosek B.3. $BCDH_{G_1,G_2,\hat{e}} =_P BICDH_{G_1,G_2,\hat{e}}$

Dowód. Wynika natychmiastowo z B.2, B.1 i przechodniości relacji $=_P$

Literatura

[1] M. J. Atallah, M. Blanton, N. Fazio, K. B. Frikken, *Dynamic and Efficient Key Management for Access Hierarchies*, ACM Transactions on Information and System Security, Vol. 12, No. 3, Article 18, January 2009.

[2] D. Boneh, M. Franklin, *Identity-Based Encryption from the Weil Pairing*, SIAM J. of Computing, Vol. 32, No. 3, pp. 586–615, 2003

[3] *Announcing the Advanced Encryption Standard (AES) Federal Information Processing Standards Publication 197*, United States National Institute of Standards and Technology (NIST). November 26, 2001.

[4] C-H Liu, Y-F Chung, T-S Chen, S-D Wang, *An Id-based Access Control In A Hierarchical Key management For Mobile Agent*, International Journal of Innovative Computing, Information and Control Volume 7, Number 3, March 2011

[5] S. J. MacKinnon, P. D. Taylor, H. Meijer, S. G. Akl, *An Optimal Algorithm for assigning cryptographic keys to control access in a Hierarchy*, IEEE Transactions on Computers, vol C-34, no. 9 (1985)

[6] A. Shamir, *How to share a secret*, Communications of the ACM, Volume 22 Issue 11, Nov. 1979, 612–613

[7] J. H Silverman, it The Arithmetic of Elliptic Curves, 2nd Edition, Springer (2009)

[8] L. C. Washington, *Elliptic curves. Number theory and Cryptography*, Chapman & Hall/CRC(2003)

GENERALIZED ACCESS STRUCTURES WITH HIERARCHY

Abstract. Access structures are used in cases associated with situations when one or more entities are trying to get a resource. We will present a generalization to the case of access structures many resources, which allows for a nice description of thresholds and hierarchical schemes. We will also present the use of the so-called bilinear product, defined in the group of n-torsion points of an elliptic curve over a finite field on two exemplary hierarchical allocation key schemes.

Keywords: access structures, privileges, secret sharing, threshold scheme, hierarchy, Weil pairing, bilinear pairing, groups, elliptic curves, monotonic structures.