KEY AGREEMENT IN DYNAMIC PEER GROUPS

Seminar Kryptographische Protokolle SS 2009

Motivation

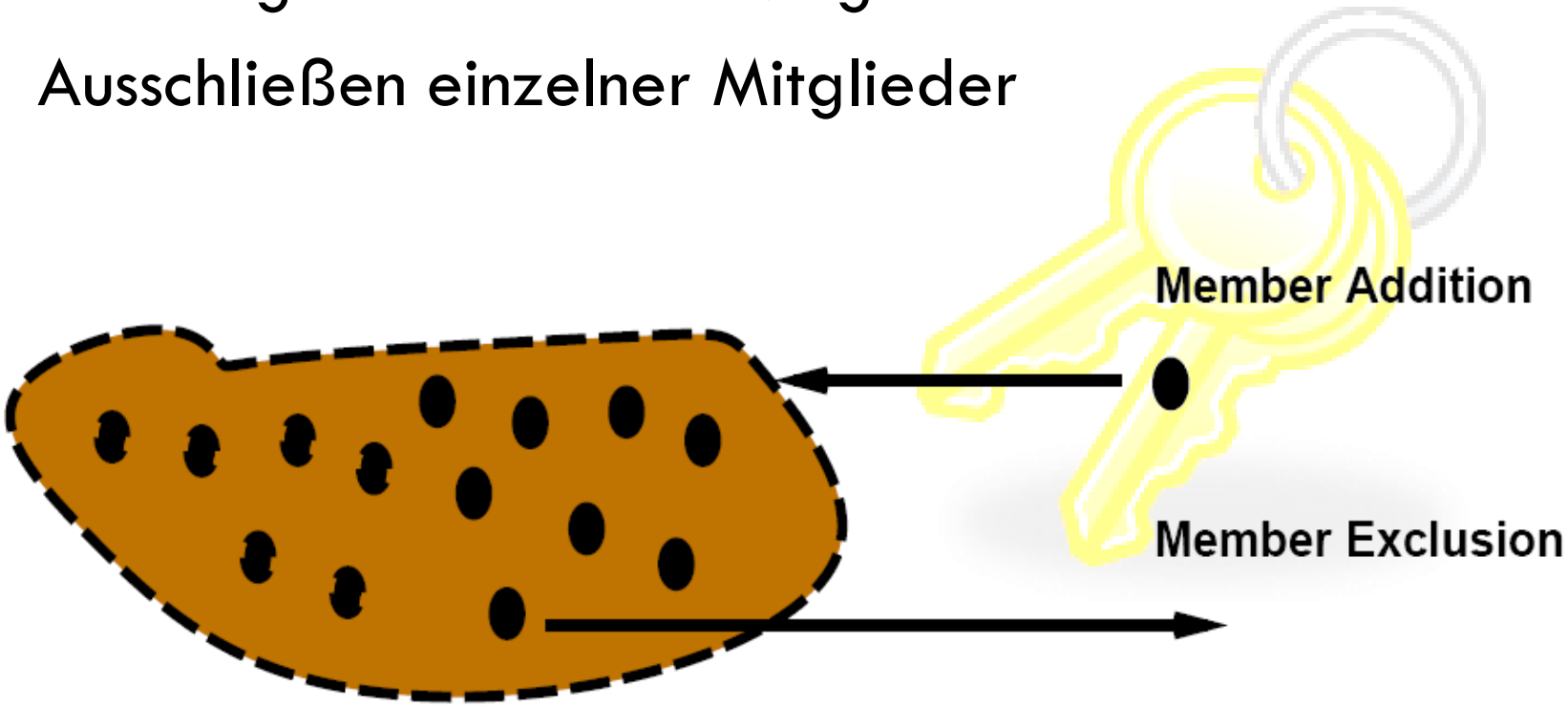
- Gruppenorientierte Anwendungen, Protokolle und Kommunikation treten in vielen Umgebungen auf:
  - Netzwerk-Schicht: Multicasting
  - Anwendungs-Schicht: Telefon- und Videokonferenzen
- Peer-to-Peer Sicherheit ist ausreichend erkundet, Gruppenkommunikation ist jedoch weitgehend unerforscht
- Gruppenkommunikation wird meist als triviale Erweiterung des 2-Personen falls gesehen

Konkretisierung des Problems

- Hier entstehen jedoch die Unterschiede
  - Protokolleffizienz wichtiger aufgrund der Menge an Teilnehmern
  - Dynamik der Gruppen
- Grundlegende Initialisierung geht auf den 2-Party-Diffie-Hellman Schlüsselaustausch zurück
- Folgende dynamische Operationen müssen unter dem Sicherheitsaspekt ermöglicht werden:

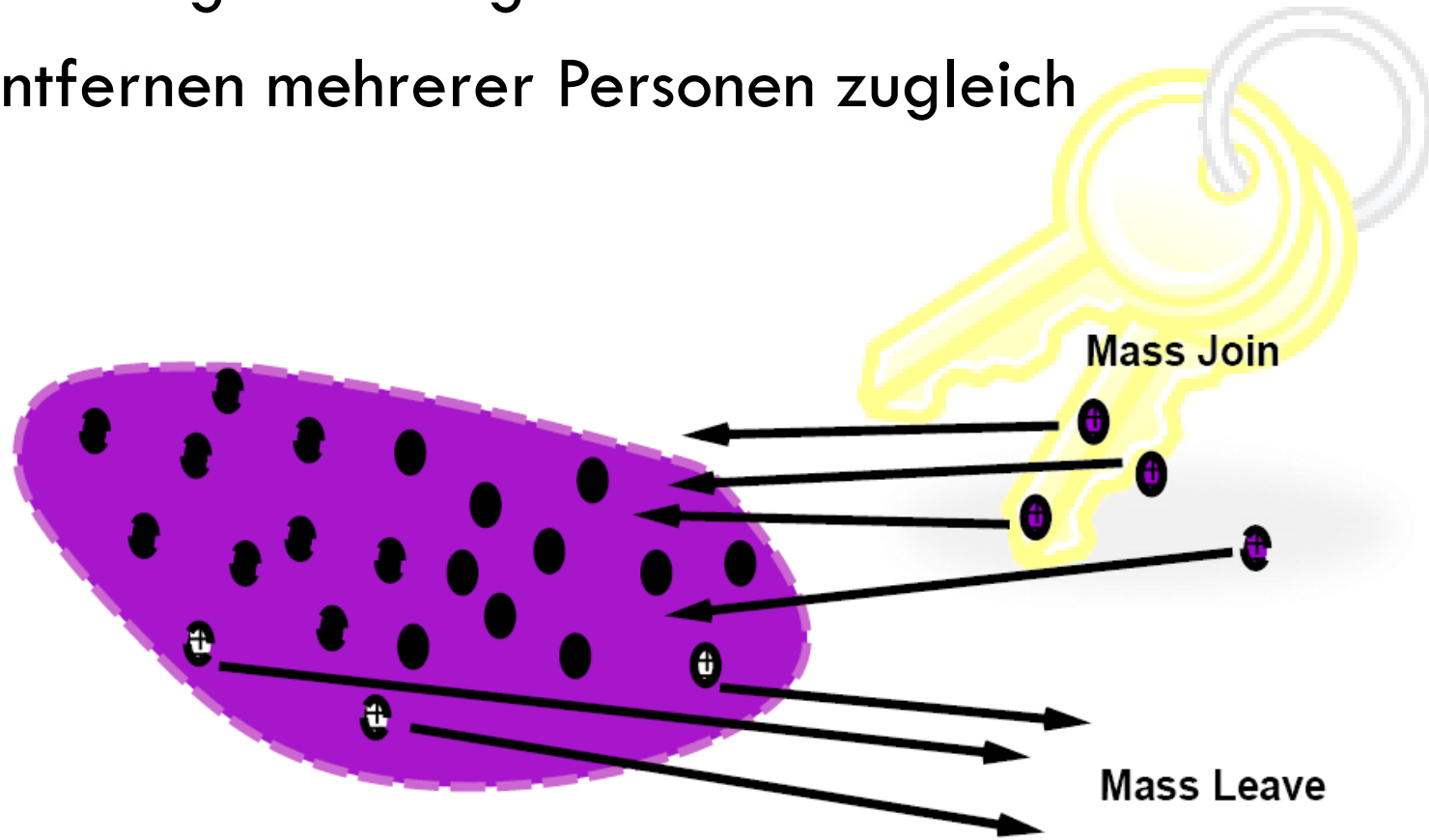
Operationen für einzelne Mitglieder

- Hinzufügen eines neuen Mitglieds
- Ausschließen einzelner Mitglieder



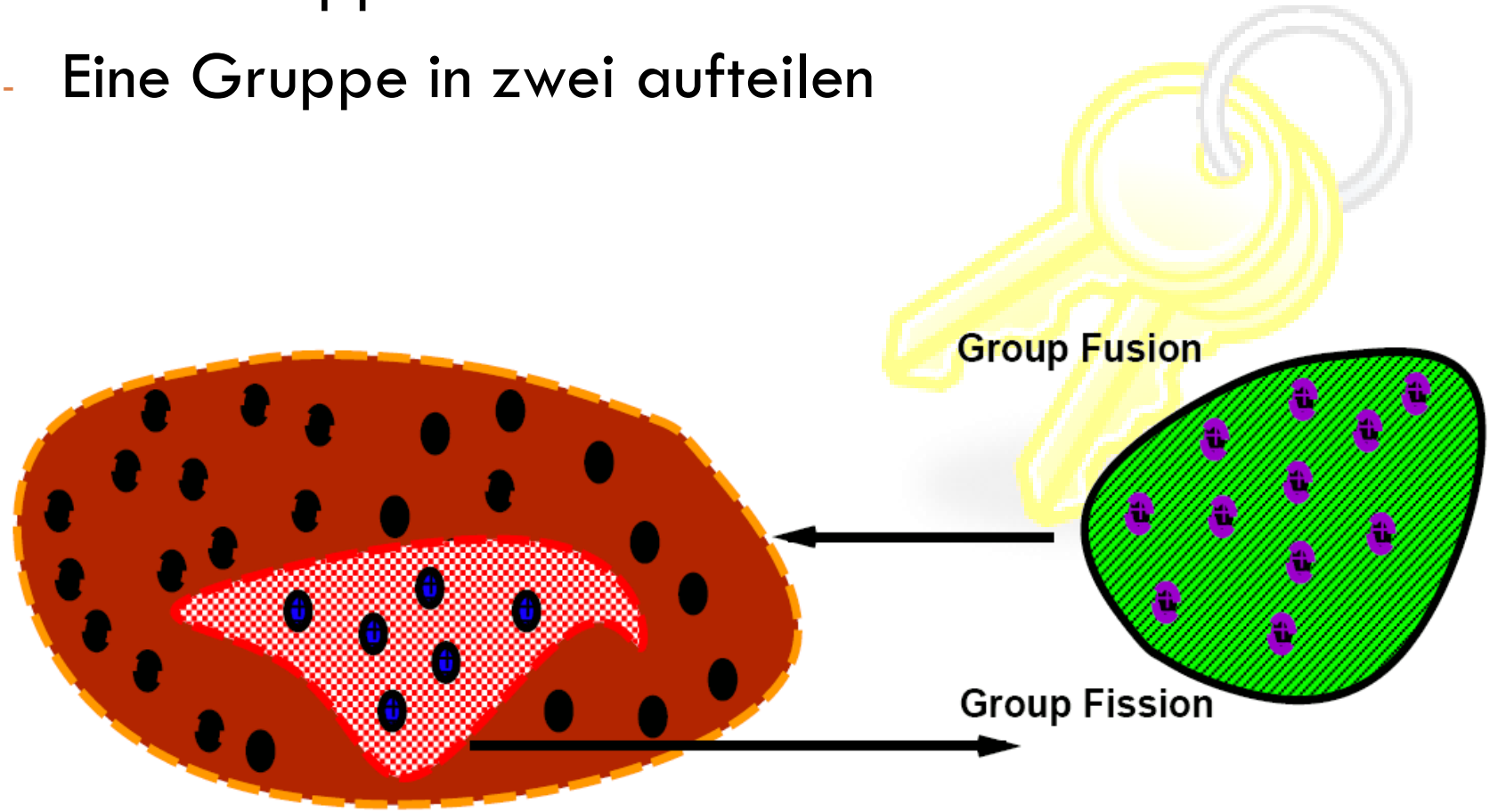
Operationen für mehrere Mitglieder

- Hinzufügen einer großen Anzahl an Personen
- Entfernen mehrerer Personen zugleich



Operationen für mehrere Mitglieder

- Zwei Gruppen fusionieren
- Eine Gruppe in zwei aufteilen



Lösung für den Gruppenansatz

Erweiterung des Diffie-Hellman Ansatzes auf mehrere als nur zwei Teilnehmer:

- Alle Mitglieder tragen zu einer zyklischen Gruppe bei, mit  als Generator
- Entspricht Gruppen die bei Diffie-Hellman Verschlüsselung ebenfalls verwendet werden

Gegebener Rahmen

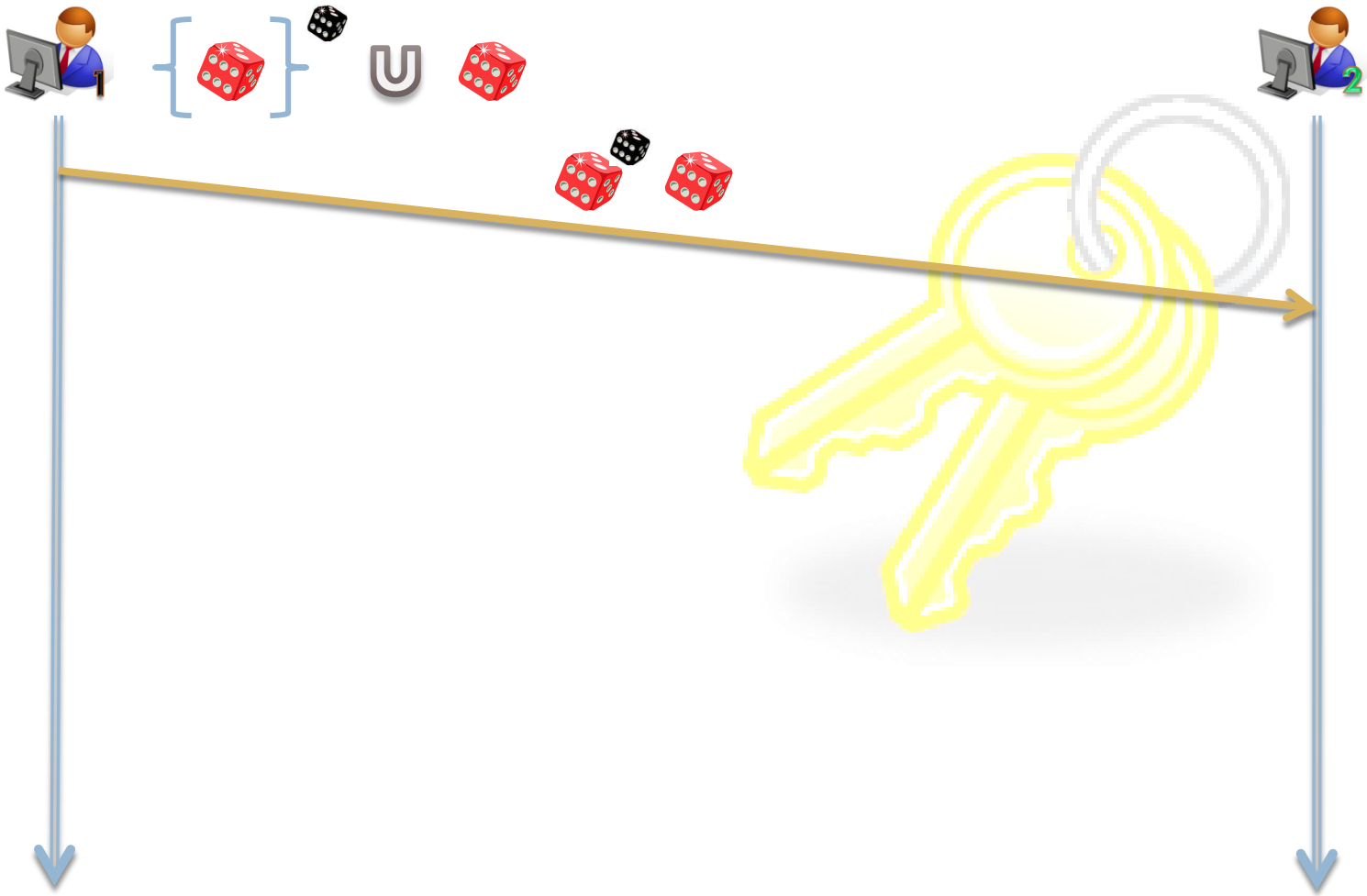
- In den Folgenden Beispielen gehen wir von 4-Mitgliedern aus:

Schlüssel entspricht dann:

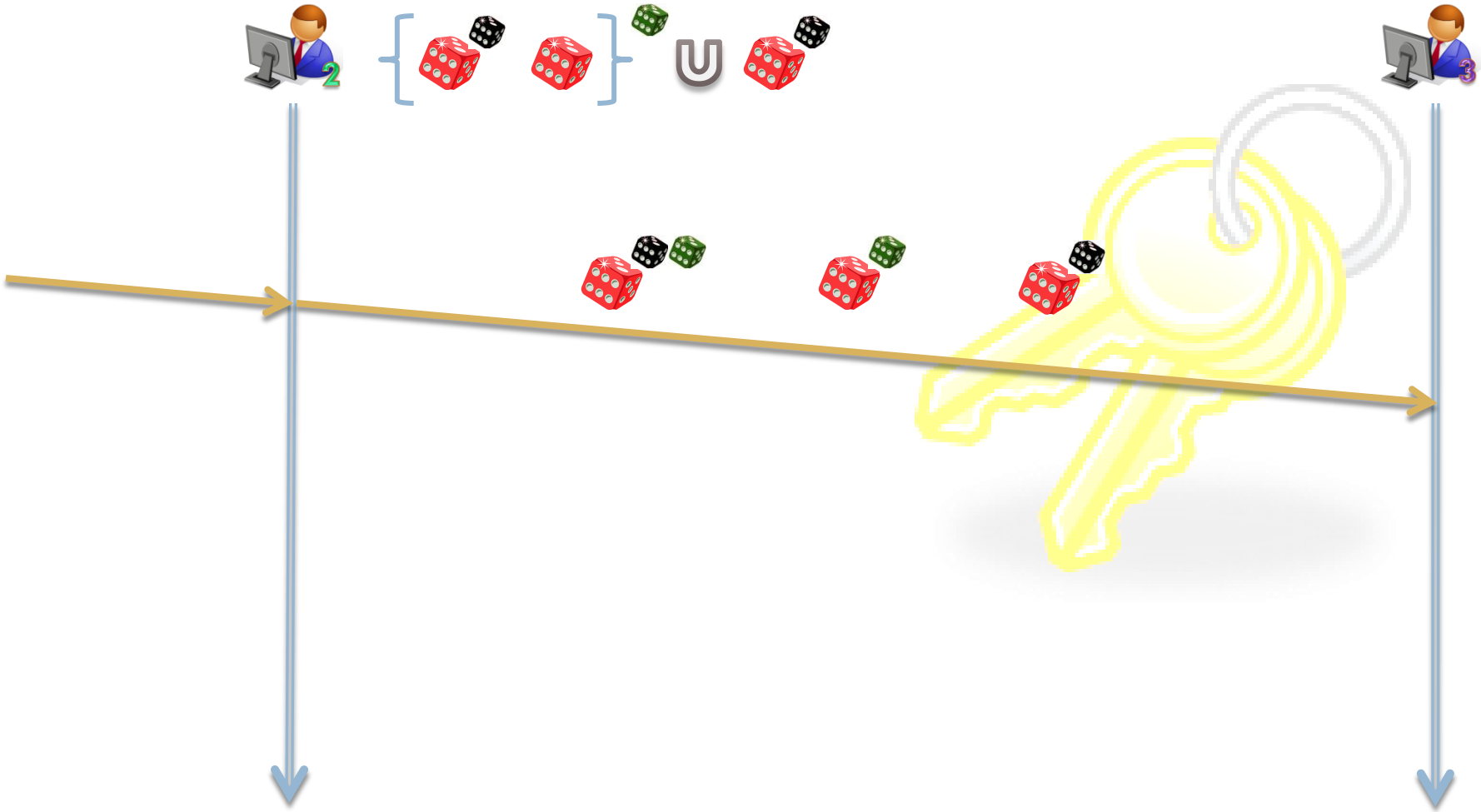
IKA (Initial Key Agreement)

- Findet bei der Generierung einer Gruppe statt
- Focus auf Sicherheit, nicht auf Effizienz
- Das beisteuern eines Teilschlüssels ist gleichbedeutend mit der Gruppenteilnahme und dem Erhalt von Schlüsselinformationen der anderen Mitglieder
- Einzig nötige Operationen in statischen Gruppen

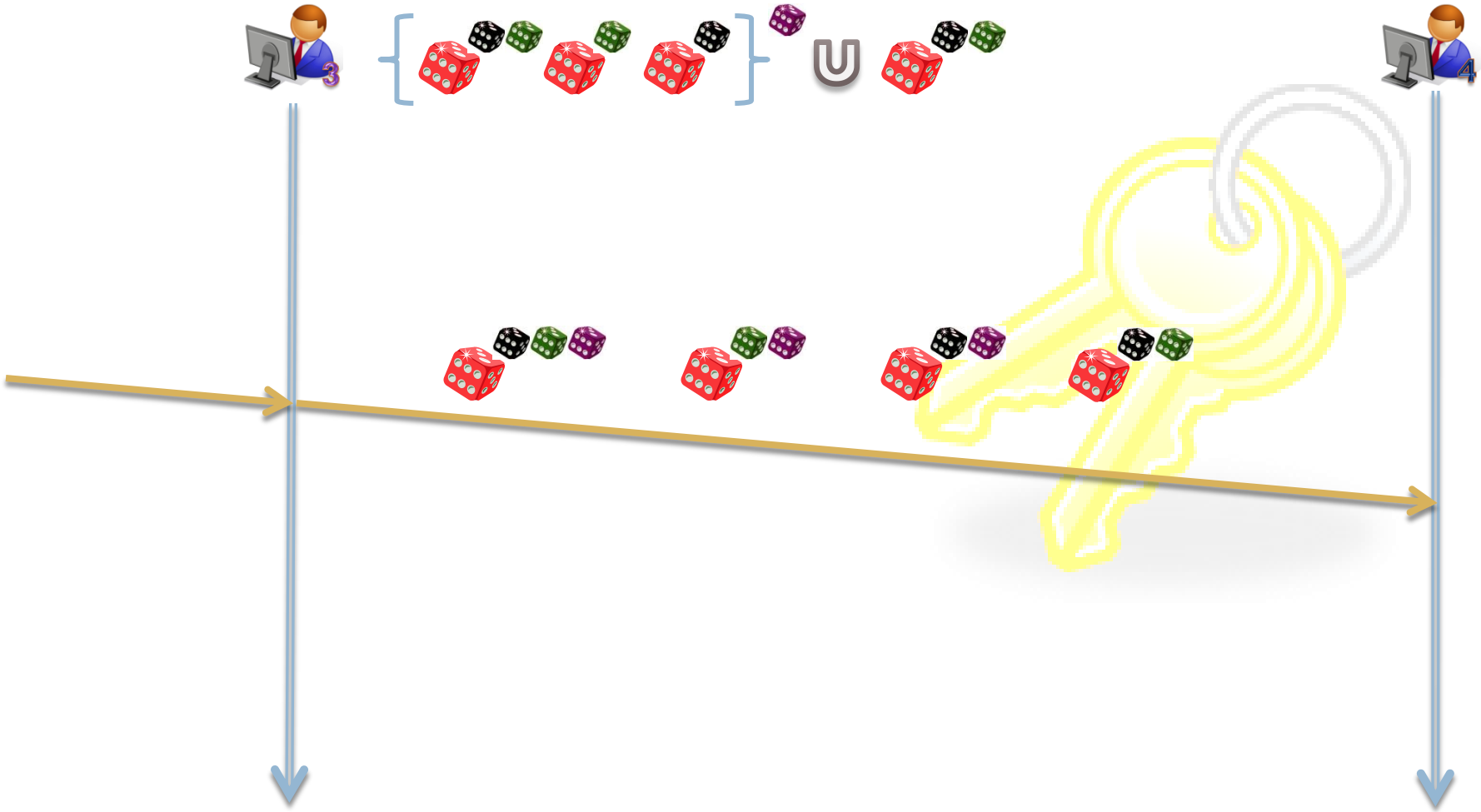
CLIQUES: IKA.1



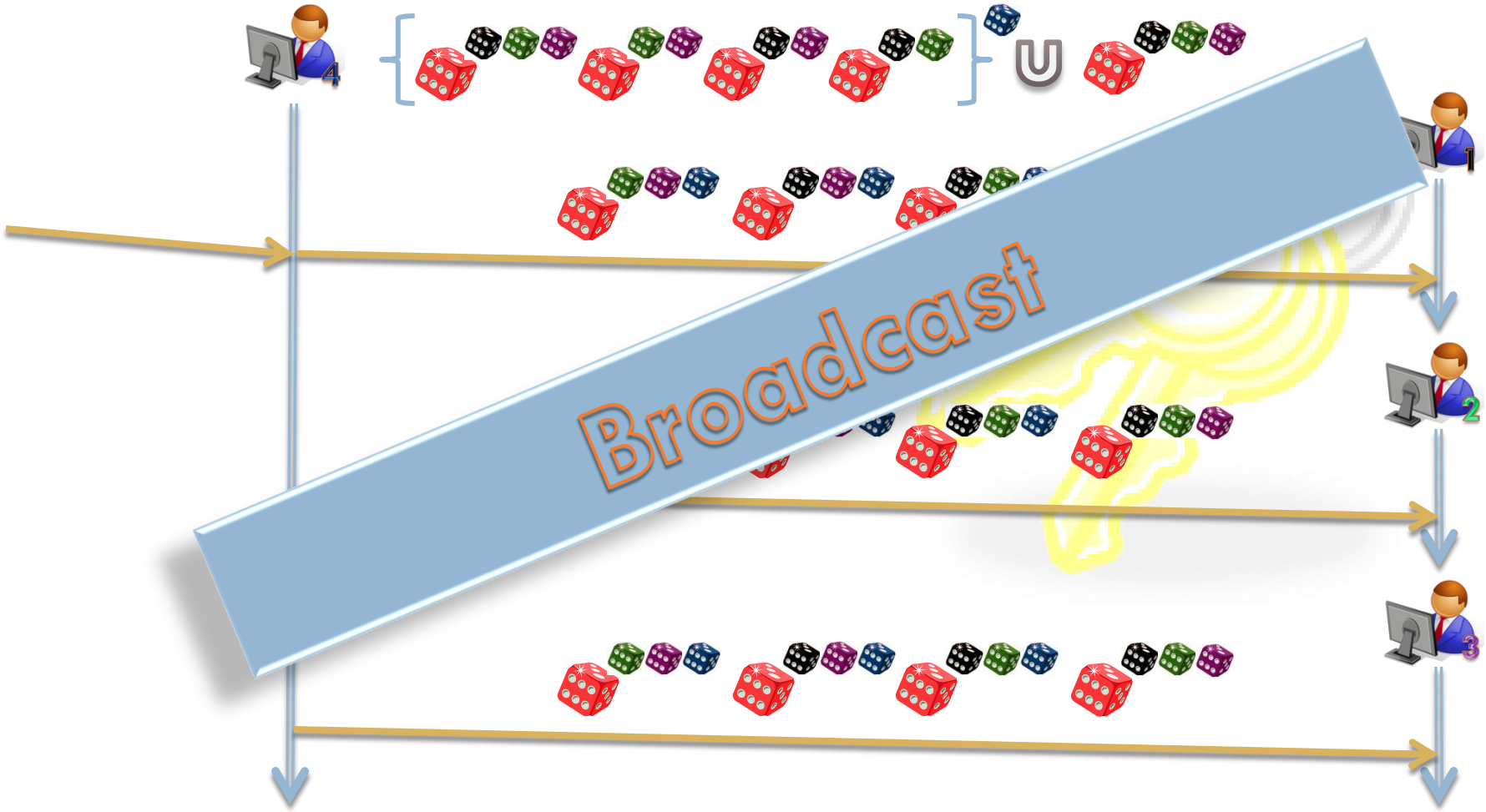
CLIQUES: IKA.1



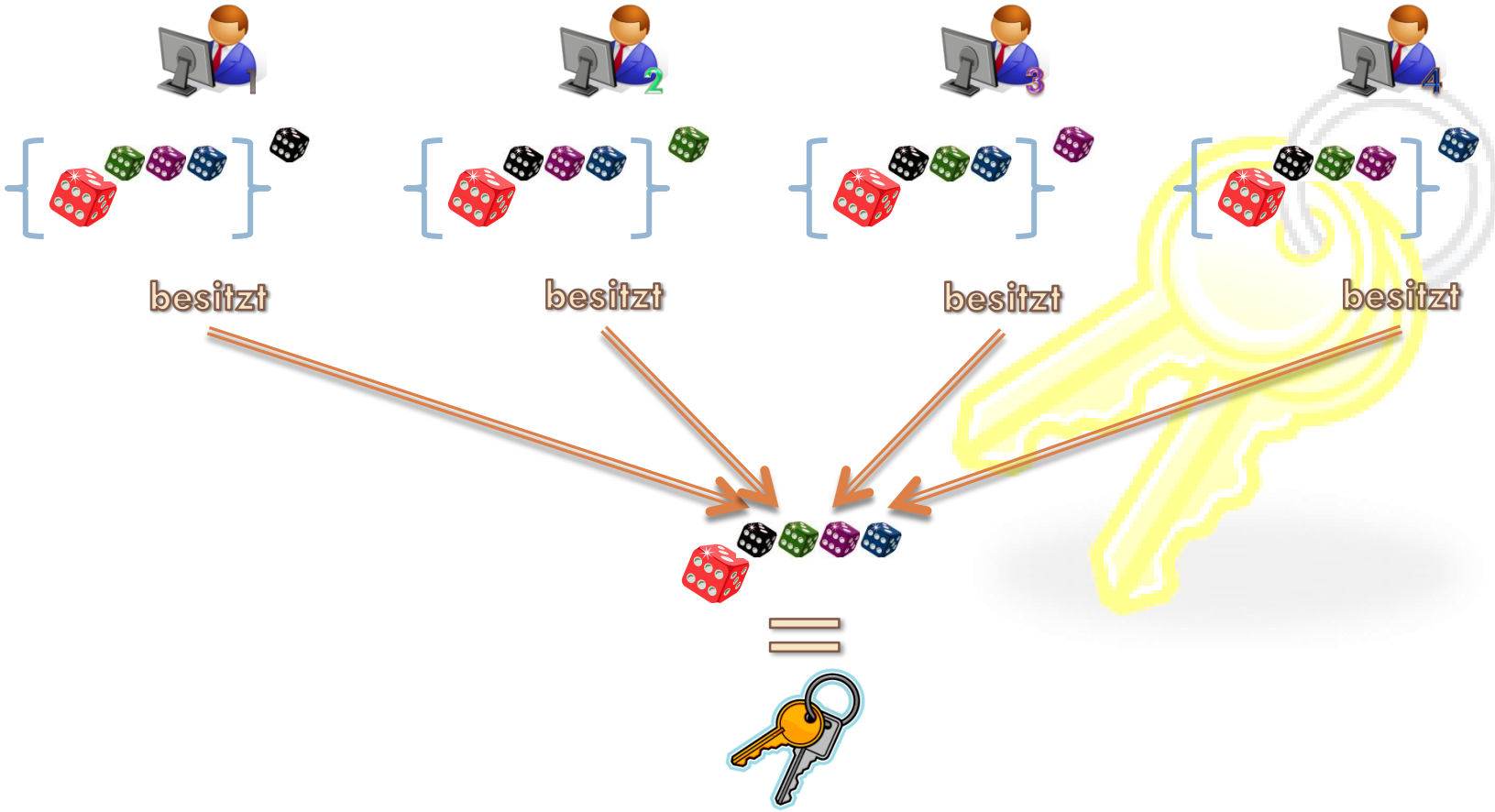
CLIQUES: IKA.1



CLIQUES: IKA.1



CLIQUES: IKA.1



CLIQUES: IKA.1 Zusammenfassung

Summe der Potenzen

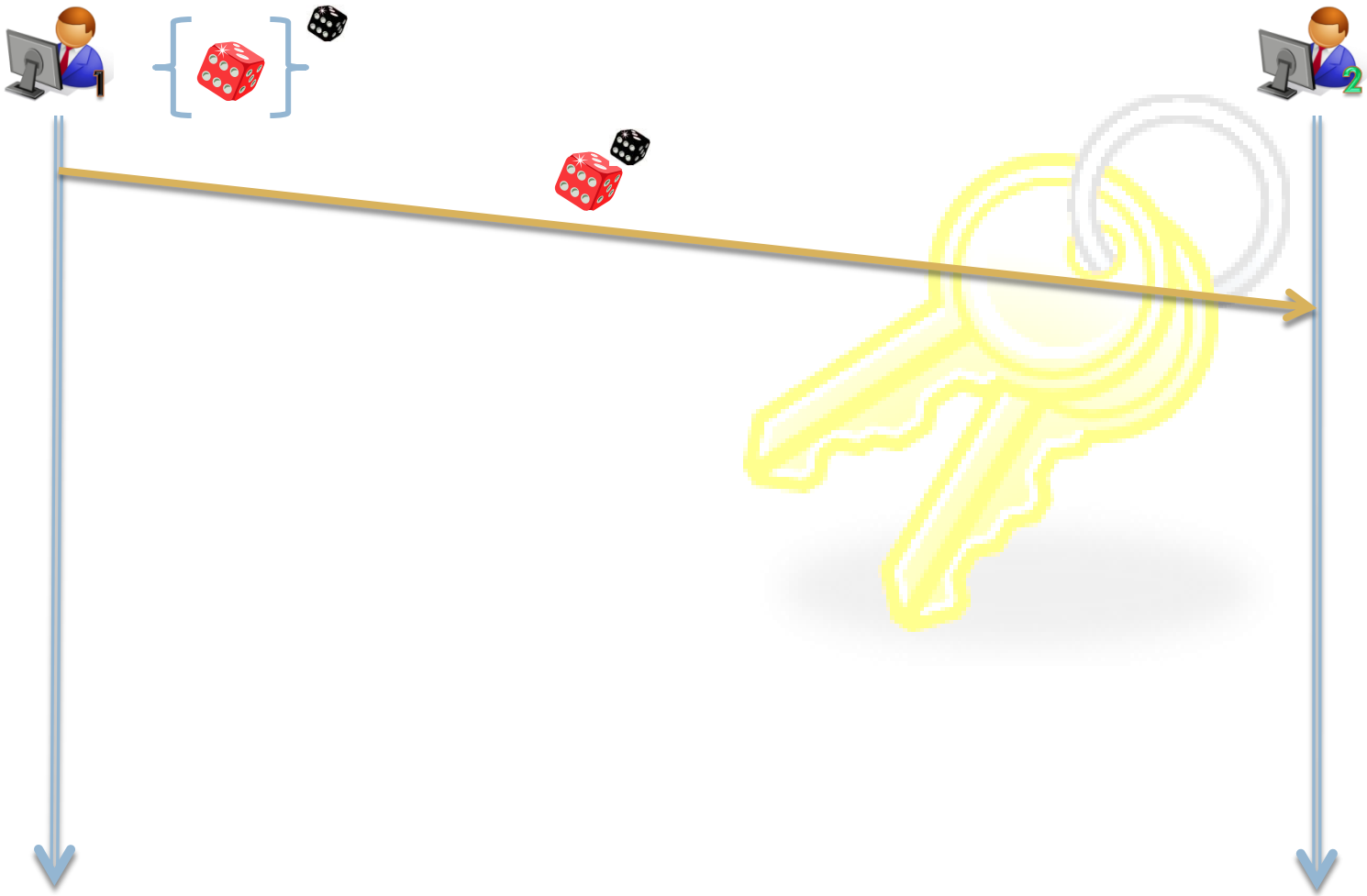
- ((n + 3) n) / 2) - 1

- Problem der nicht-linearen Anzahl an Potenzen

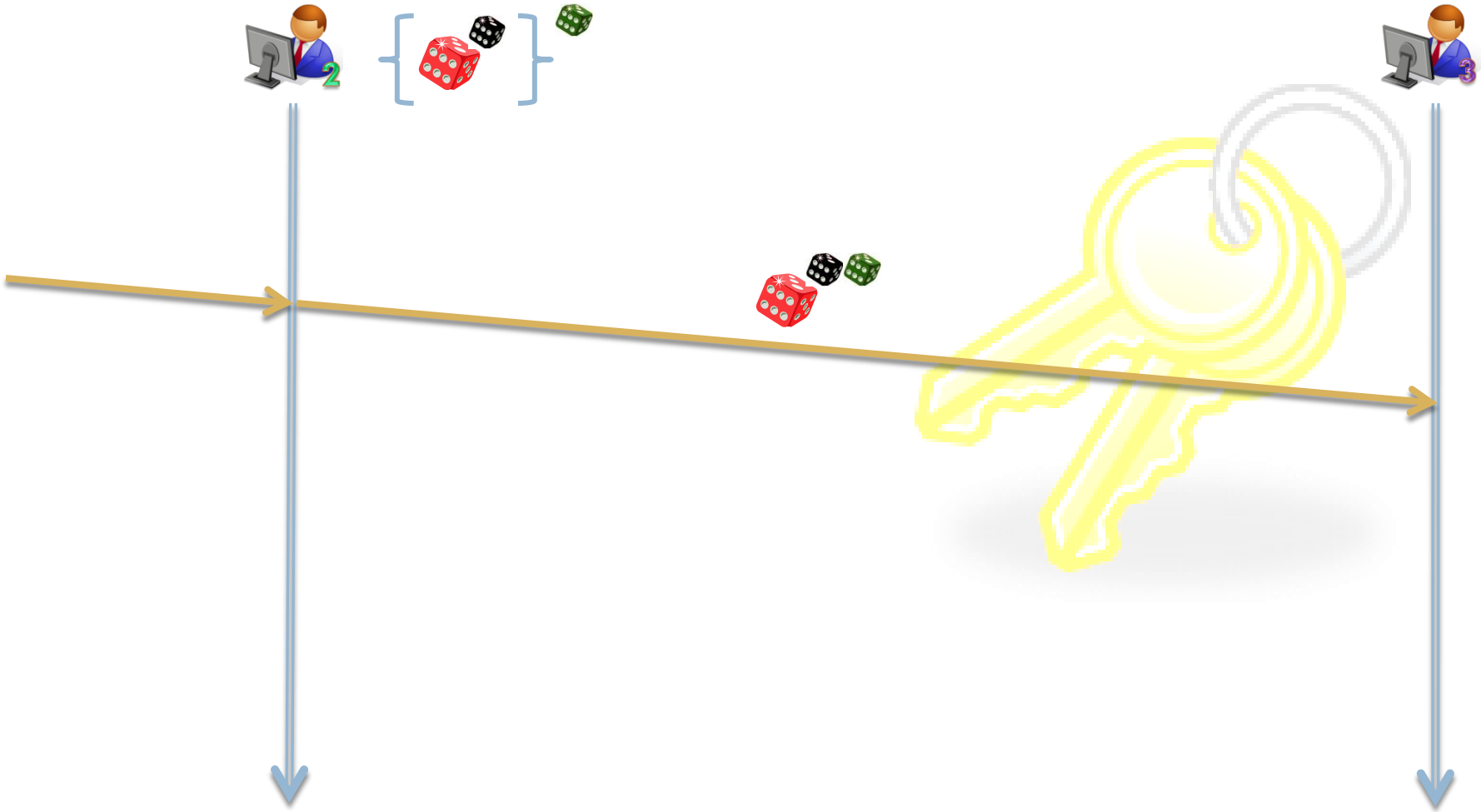
⇨ Probleme bei großen Gruppen

- Gibt es eine bessere Lösung ?

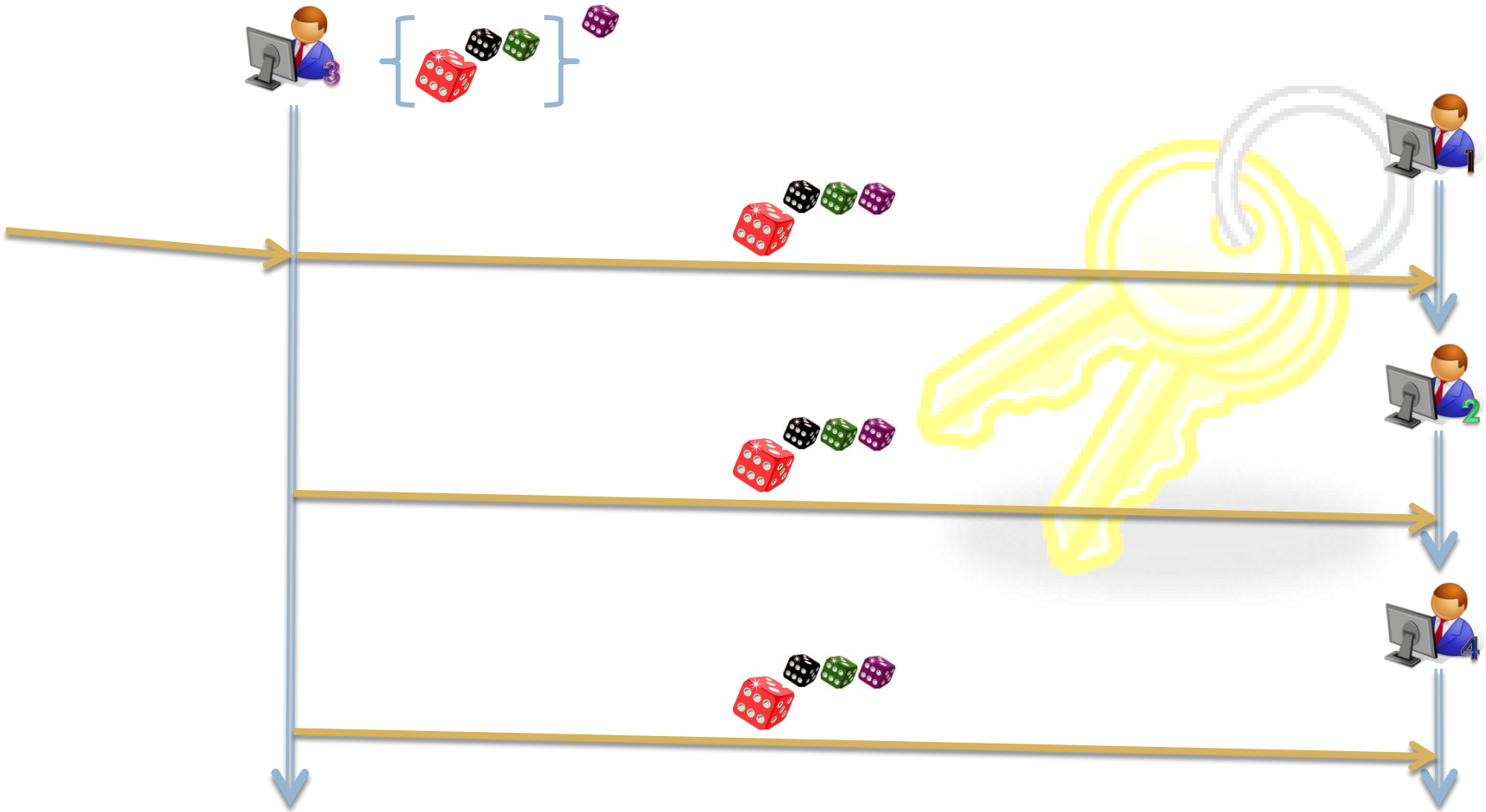
CLIQUES: IKA.2



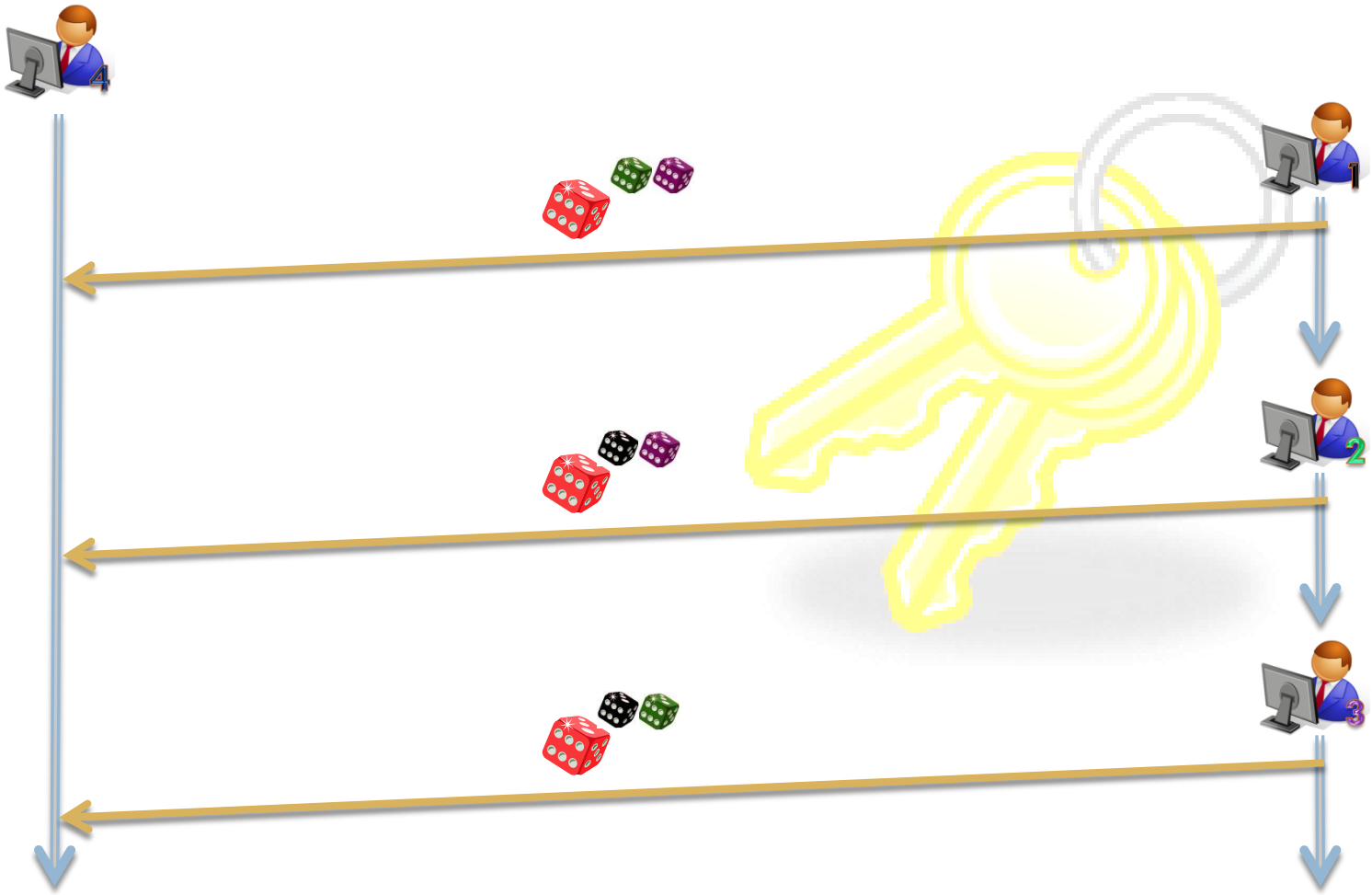
CLIQUES: IKA.2



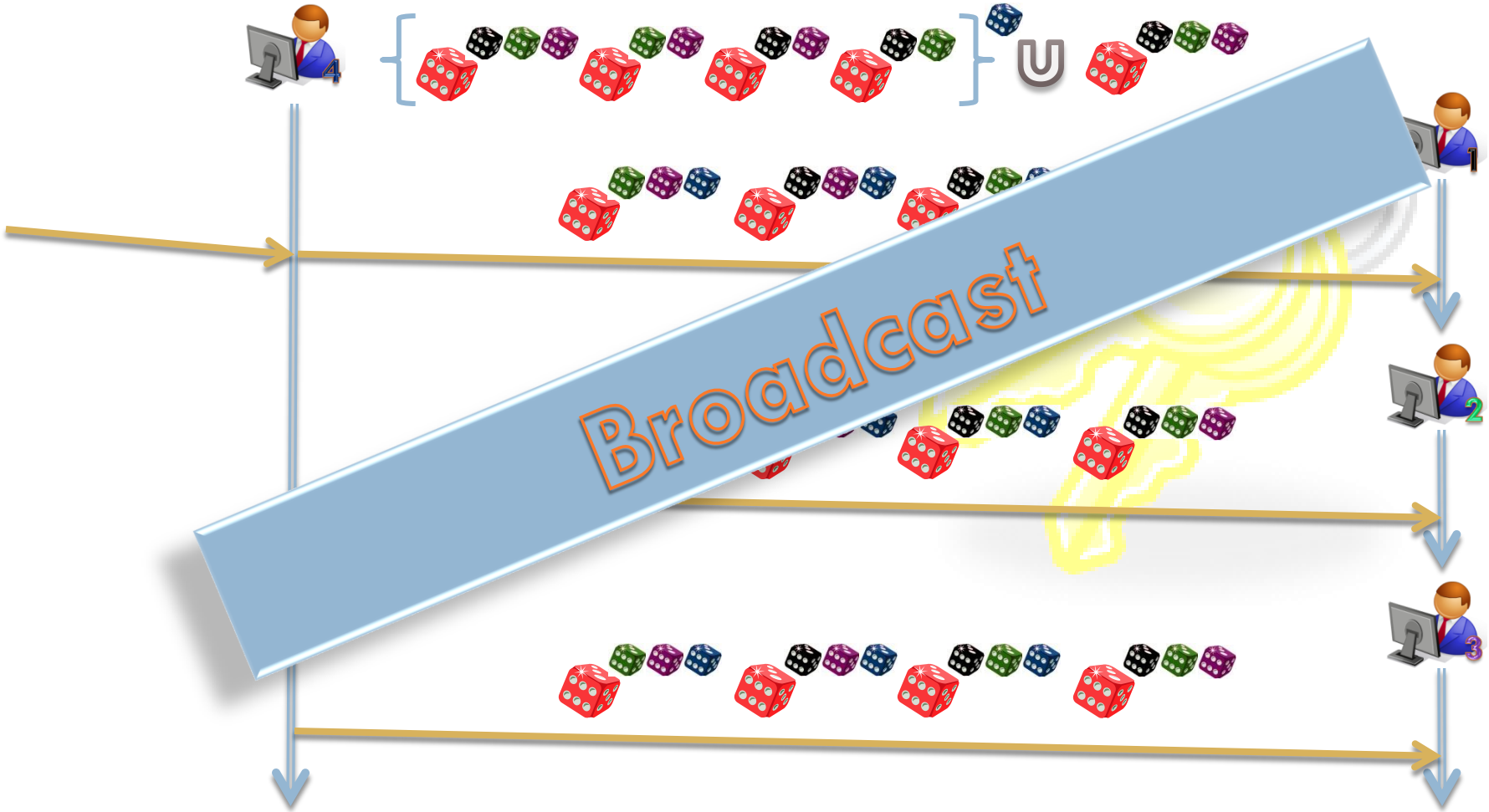
CLIQUES: IKA.2



CLIQUES: IKA.2



CLIQUES: IKA.2



CLIQUES: IKA-Vergleich

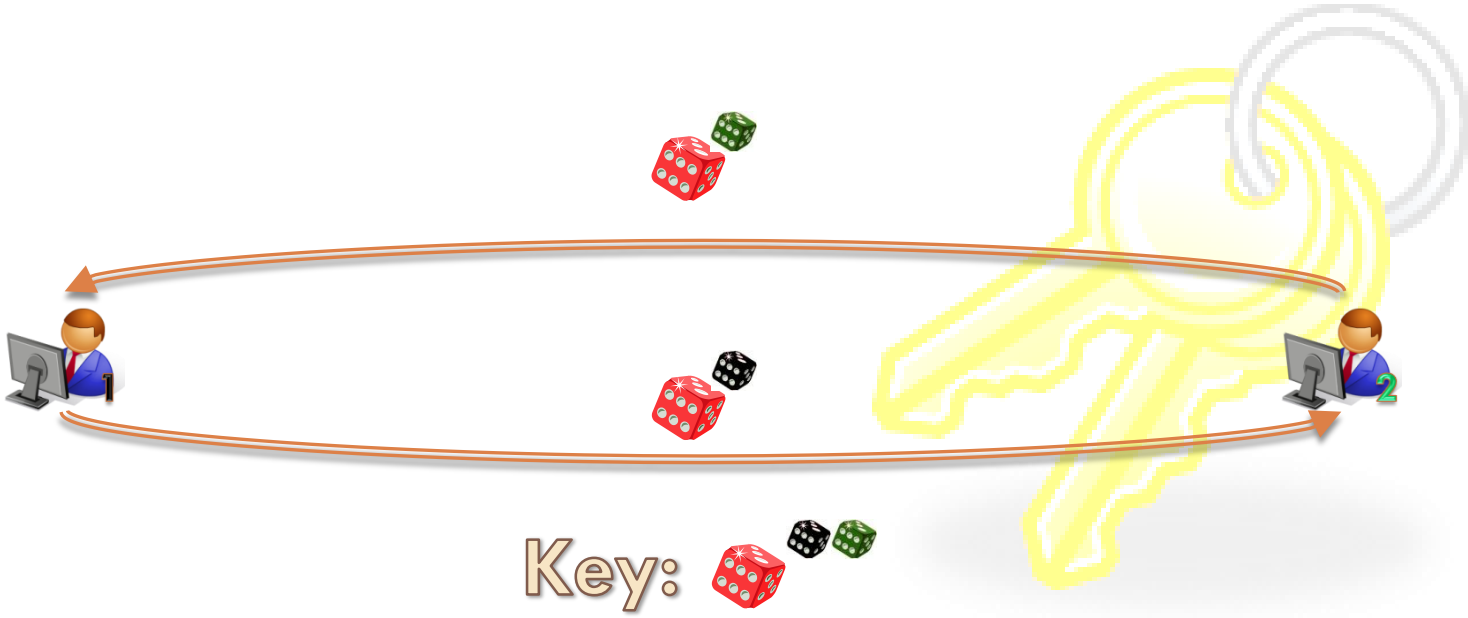
Summe der Potenzen	• 5n - 6

- IKA.2 benötigt weniger Potenzen
- IKA.2 besitzt eine konstante Nachrichtenlänge
- Jedoch kann es bei IKA.2 zu einem Engpass kommen, da vor dem Broadcast n-1 Unicast Nachrichten an einen Empfänger geschickt werden

Sicherheit des n-Party Diffie-Hellman Problems

Um die Sicherheit des Problems zu zeigen, müssen wir uns mit seiner Grundlage dem 2-Party Diffie-Hellman beschäftigen.

Die Diffie-Hellman Probleme



Sicherheit des Diffie-Hellman Problems

- **Computitional Problem: (CDH)**
  - Ist der Angreifer in der Lage den Schlüssel aus den beiden abgefangenen Werten zu berechnen.
  - Reduktion auf Diskreter Logarithmus Problem (DL):
    - Multiplikative Gruppe G und ein Element g gegeben
    - Gegeben ist ein $y \varepsilon \langle g \rangle$, finde ein x für das gilt $y=g^x$
  - CDH wie auch DL ist für eine gut gewählte Gruppe hart

Sicherheit des Diffie-Hellman Problems

- CDH Annahme:
  - Wenn CDH gilt, dann ist H() semantisch sicher, unter Annahme des Random-Oracle-Modells.
  - Random-Oracle-Modell:
    - Eine vereinfachte Annahme einer Hashfunktion die einen perfekten Zufallsgenerator simuliert

Sicherheit des Diffie-Hellman Problems

- **Decisional Problem: (DDH)**
  - Kann der Angreifer den Schlüssel, mithilfe der abgefangenen Werte, von einer gegebenen Zufallszahl unterscheiden.
- Konzept der polynomiellen Ununterscheidbarkeit :
  - Ein Verfahren ist sicher wenn es keinen Algorithmus gibt, der in polynomieller Laufzeit zwischen einem Diffie-Hellman Schlüssel und einer Zufallszahl mit einer Wahrscheinlichkeit signifikant größer als ½ unterscheiden kann
  - Dies trifft für gut gewählte Gruppen auf Diffie-Hellman zu

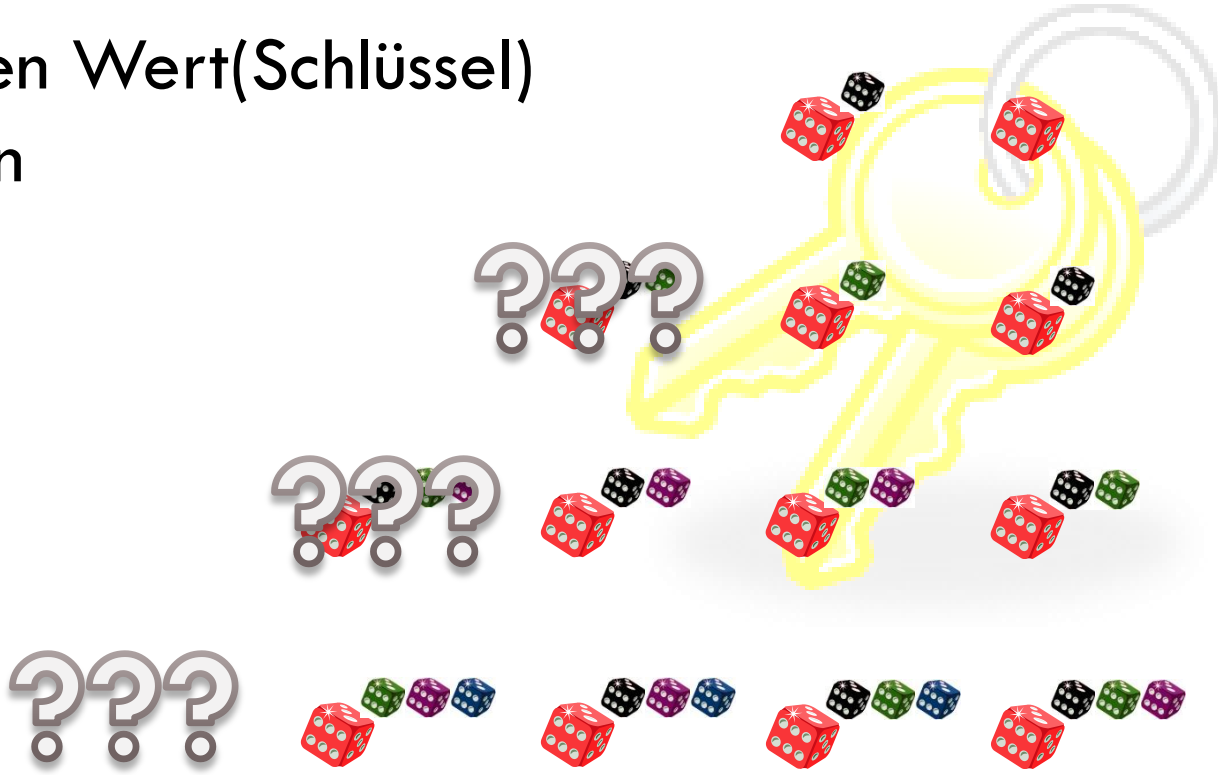
Sicherheit des Diffie-Hellman Problems

- DDH Annahme:
  - Wenn DDH gilt, dann ist  semantisch sicher.
  - DDH stärker als CDH

- Daraus folgt:
  - DL ≥ CDH ≥ DDH

Group-Computitional-DH

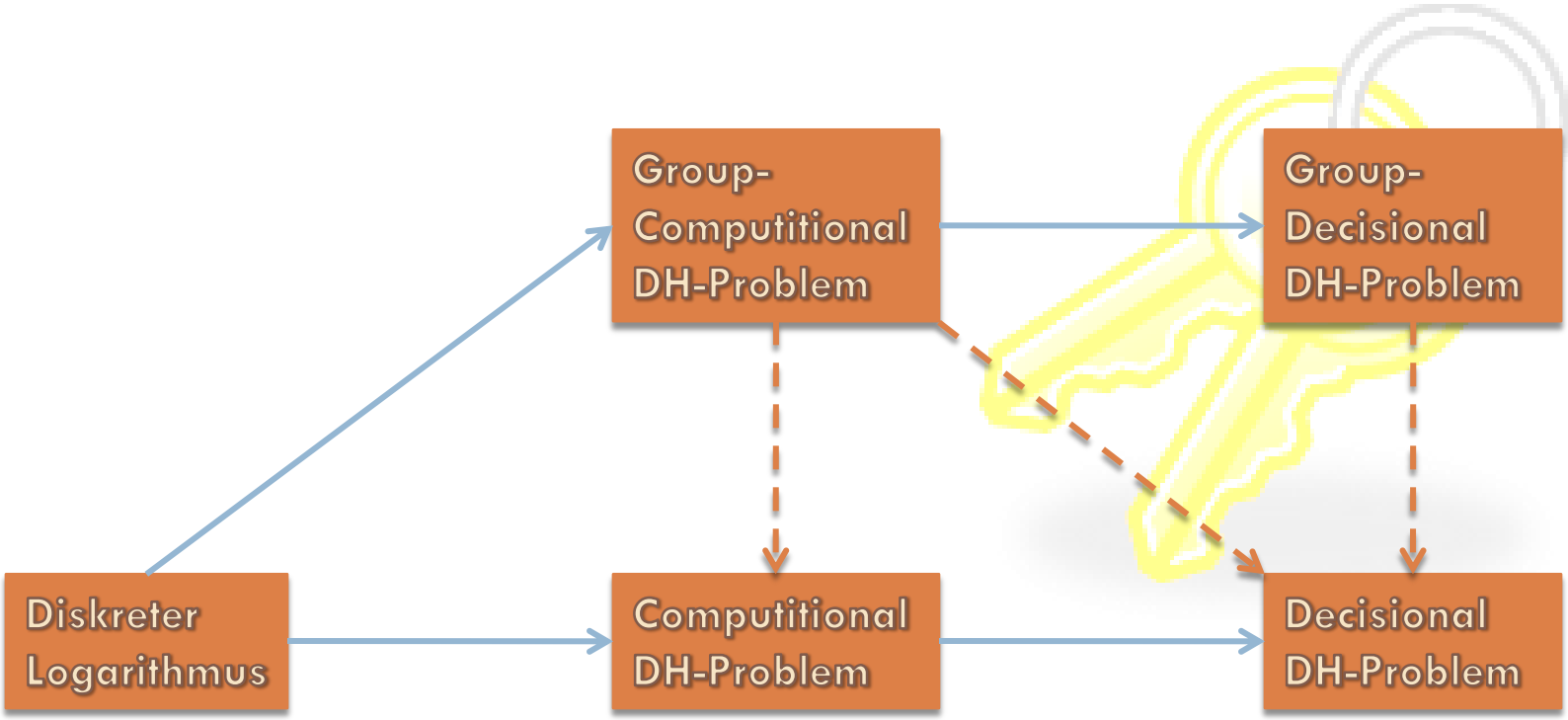
- Problem:
  Den letzten Wert(Schlüssel) berechnen



Group-Decisional-DH

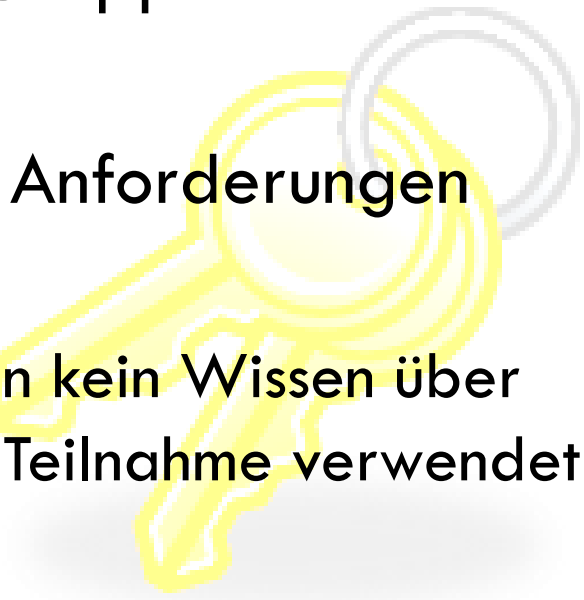
- Problem:
  Den letzten Wert(Schlüssel)
  von einem Zufallswert
  unterscheiden

Problem-Hierarchie



AKA (Auxiliary Key Agreement)

- Kann jeder Zeit während der Gruppenexistenz in beliebiger Häufigkeit erfolgen
- Alle Operationen müssen zwei Anforderungen erfüllen:
  - Neue Gruppenmitglieder müssen kein Wissen über Schlüssel besitzen, die vor ihrer Teilnahme verwendet wurden
  - Neue Schlüssel dürfen ehemaligen Mitgliedern nicht bekannt sein



CLIQUES: AKA

- Dieses Protokoll unterstützt alle oben genannten Operationen

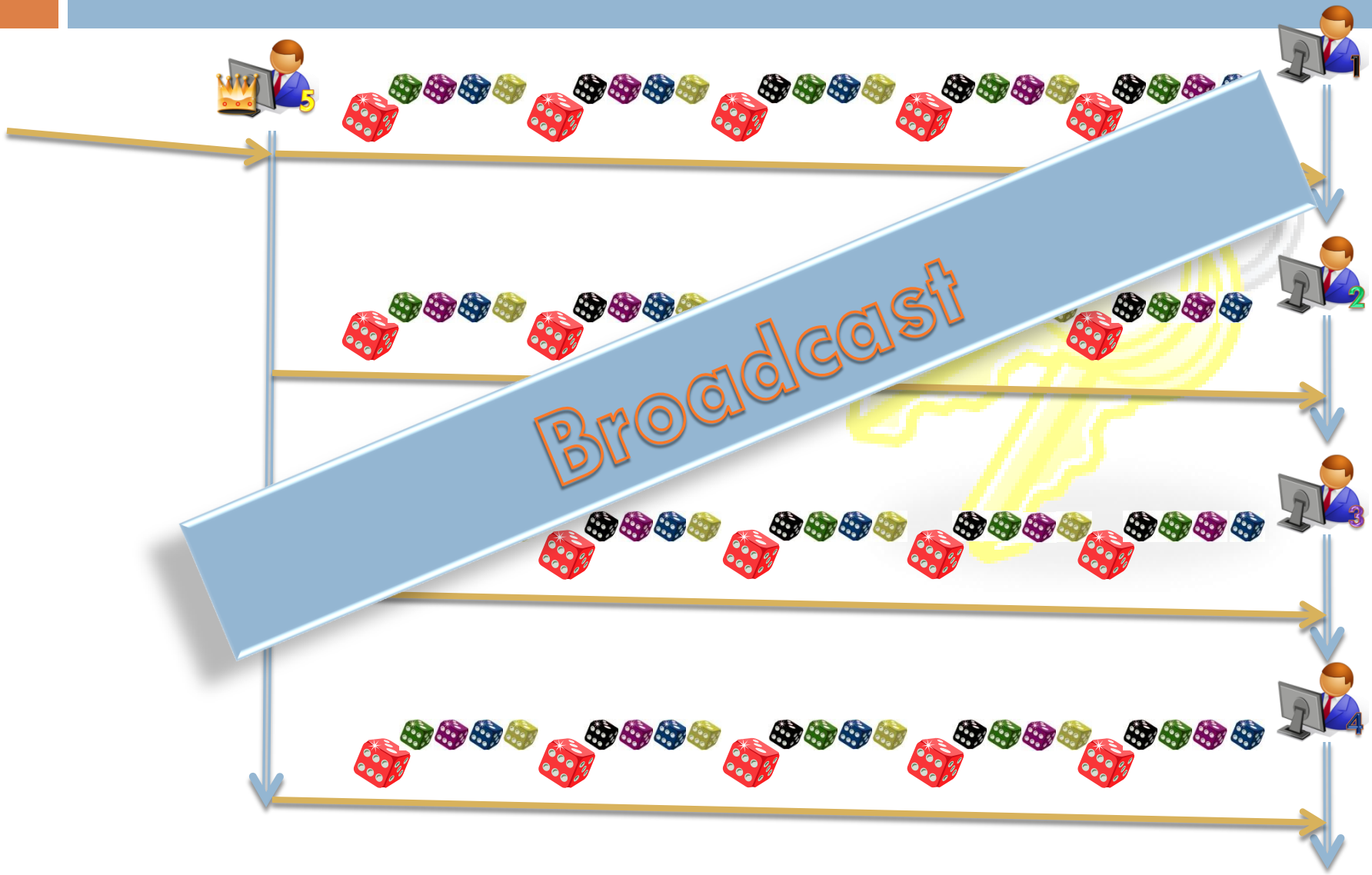
- Zusätzlich wird eine Key Refresh Operation eingeführt um:
  - Die Menge an verschlüsselten Texten, die mit einem Schlüssel generiert werden, zu minimieren
  - Verlorene Schlüssel durch Neue zu ersetzen

Member Addition Operation (fließenden Gruppenleiter)

- Wir führen ein neues Symbol () um den Gruppenleiter zu kennzeichnen
- Entspricht dem Mitglieder der den Broadcast durchführt
- Der alte Gruppenleiter ändert zuvor seinen Schlüsselanteil

Member Addition Operation (fließenden Gruppenleiter)

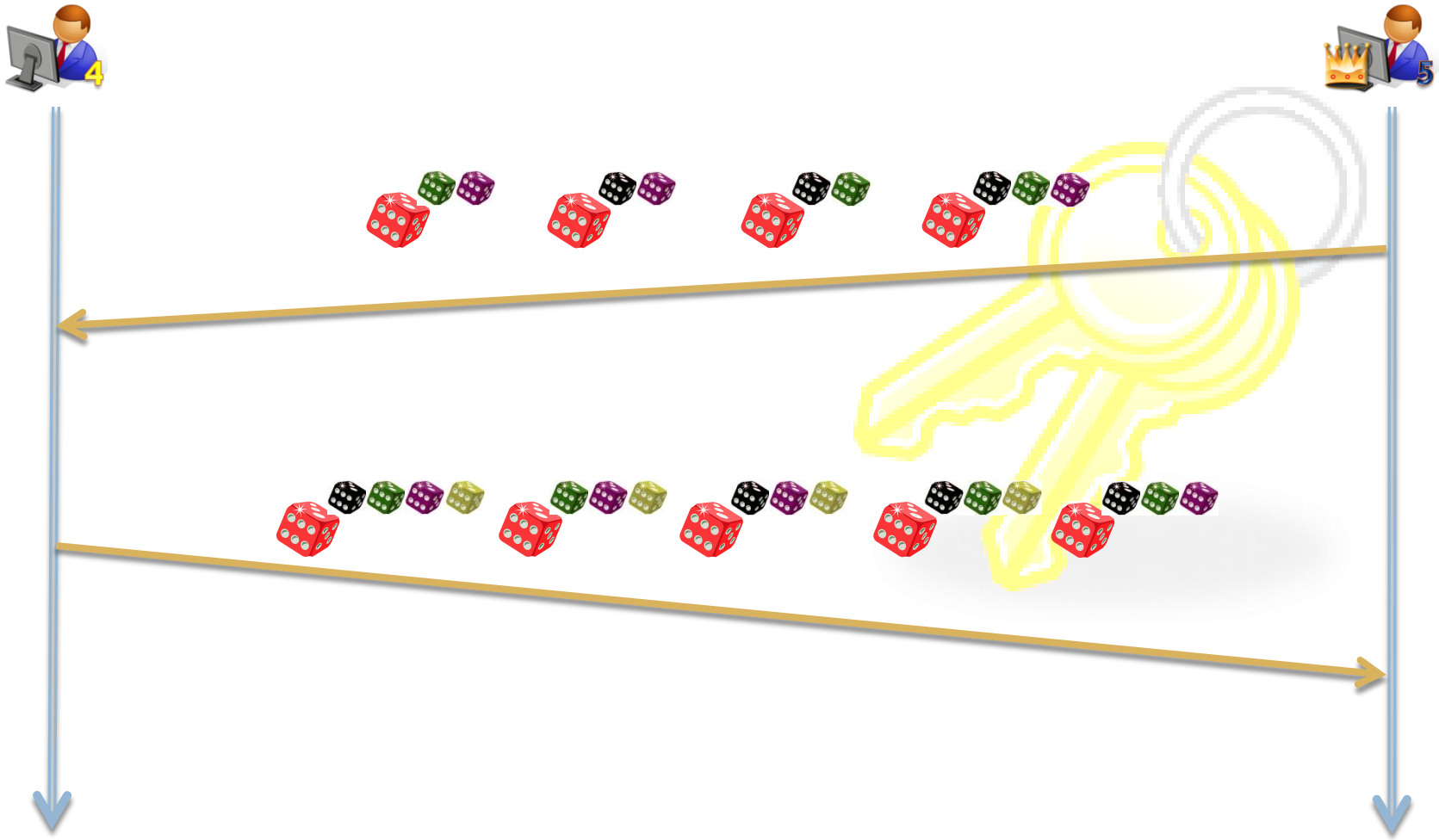
Member Addition Operation (fließenden Gruppenleiter)



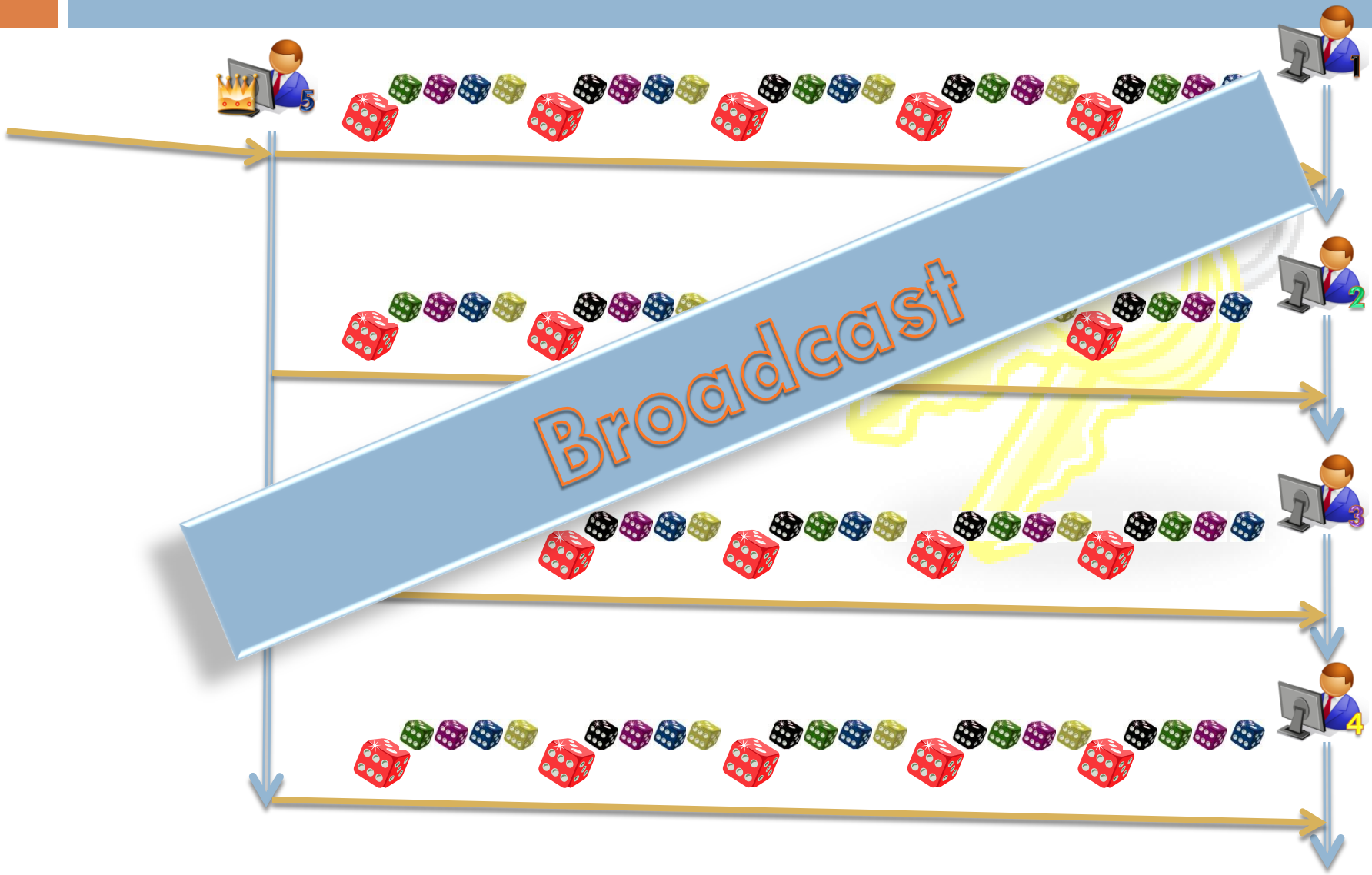
Member Addition Operation (festen Gruppenleiter)

- Bei fließenden Ansatz wird die Rolle des Gruppenleiters an das am wenigsten vertrauenswürdigen Mitglied weitergegeben
- Ansatz des festen Gruppenleiters:
  - Der Gruppenleiter bleibt fest
  - Der Index in der Reihenfolge erhöht sich stetig
  - Somit ist der letzte Index immer der Gruppenleiter

Member Addition Operation (festen Gruppenleiter)



Member Addition Operation (festen Gruppenleiter)



Mass Join Operation

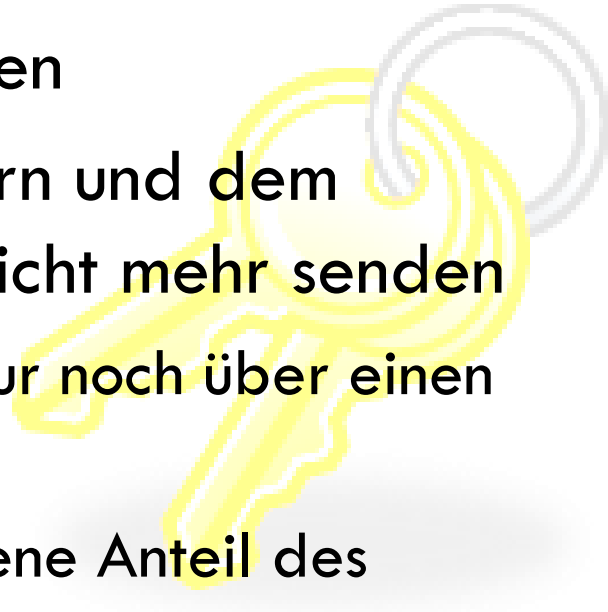
- Ziel: Eine Menge an Mitgliedern gleichzeitig hinzufügen
- Lösung 1: Mehrfachausführung Member-Addition
  - Viel zu viele Broadcast-Nachrichten
- Lösung 2: Verkettung der Member-Additions
  - Alle neuen Gruppenmitglieder werden angefügt und der Letzte ist der neue Gruppenleiter bzw. der Alte
  - Welcher den Broadcast ausführt

Group Fusion Operation

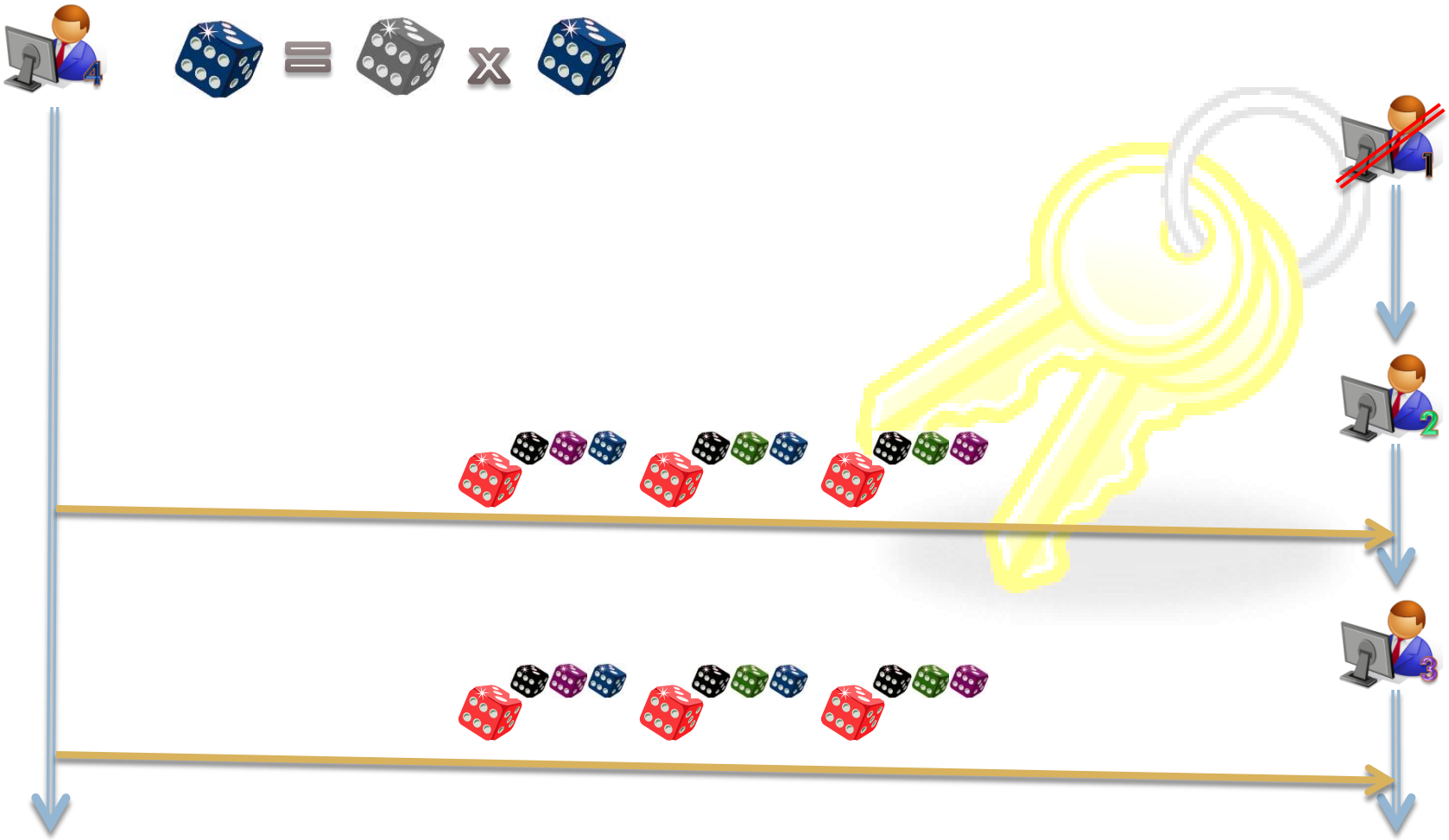
- Ziel: Eine schon existierende Gruppen mit einer anderen zu verknüpfen
- Lösung 1: Ein Spezialfall des Mass-Joins
  - Gruppen müssen relativ klein sein
  - In diesem Fall schneller
- Lösung 2: Eine Neuausführung des IKA
  - Sicherer, da alle Schlüssel-Beiträge neu errechnet
- Keines von beiden besser, da abhängig von Anwendungssituation

Member Exclusion Operation

- Ziel: Das entfernte Mitglied darf den neuen Schlüssel nicht kennen
- Lösung: Einen Beitrag ändern und dem Ausgeschlossenen nicht mehr senden
  - Der Ausgeschlossene verfügt nur noch über einen veralteten Schlüssel
  - Der im neuen Schlüssel enthaltene Anteil des Ausgeschlossenen ist irrelevant



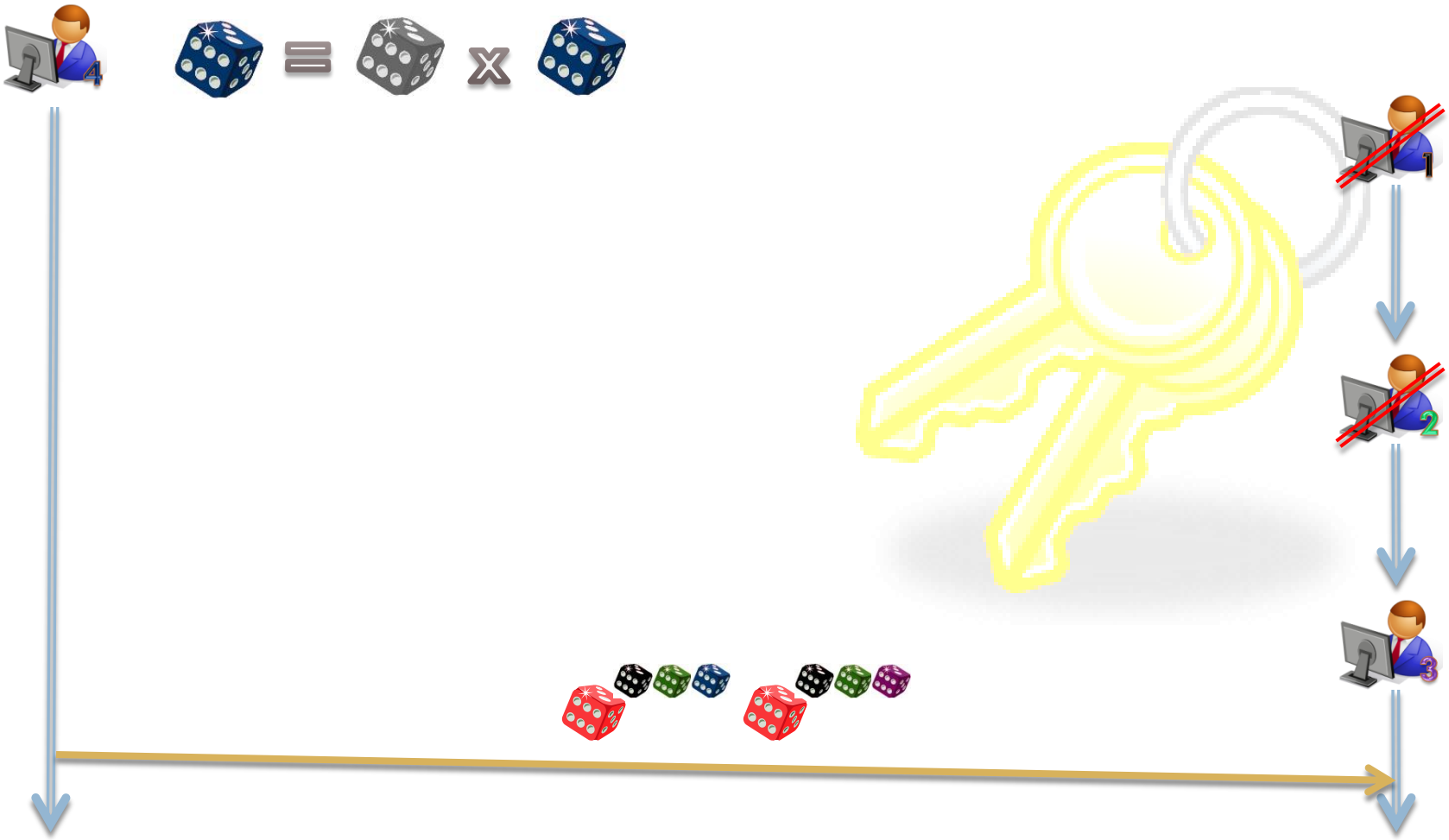
Member Exclusion Operation



Subgroup Exclusion Operation

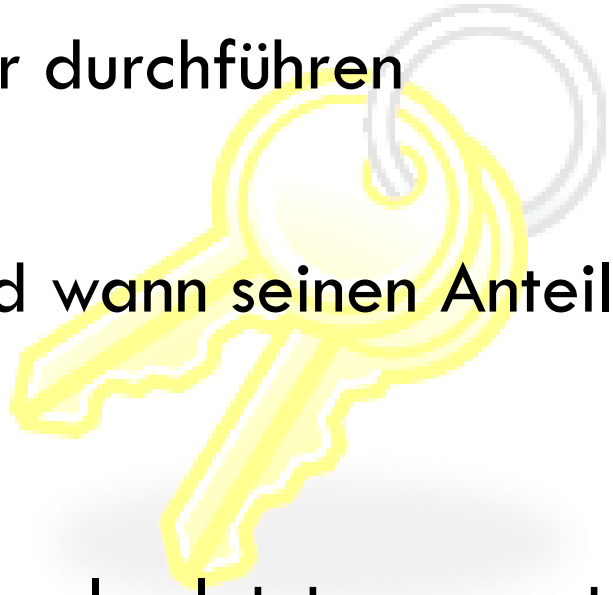
- Einfachste Lösung:
  - Statt nur einen Mitglieder den Schlüssel nicht mehr zu schicken, wird dies bei mehreren vollzogen

Subgroup Exclusion Operation

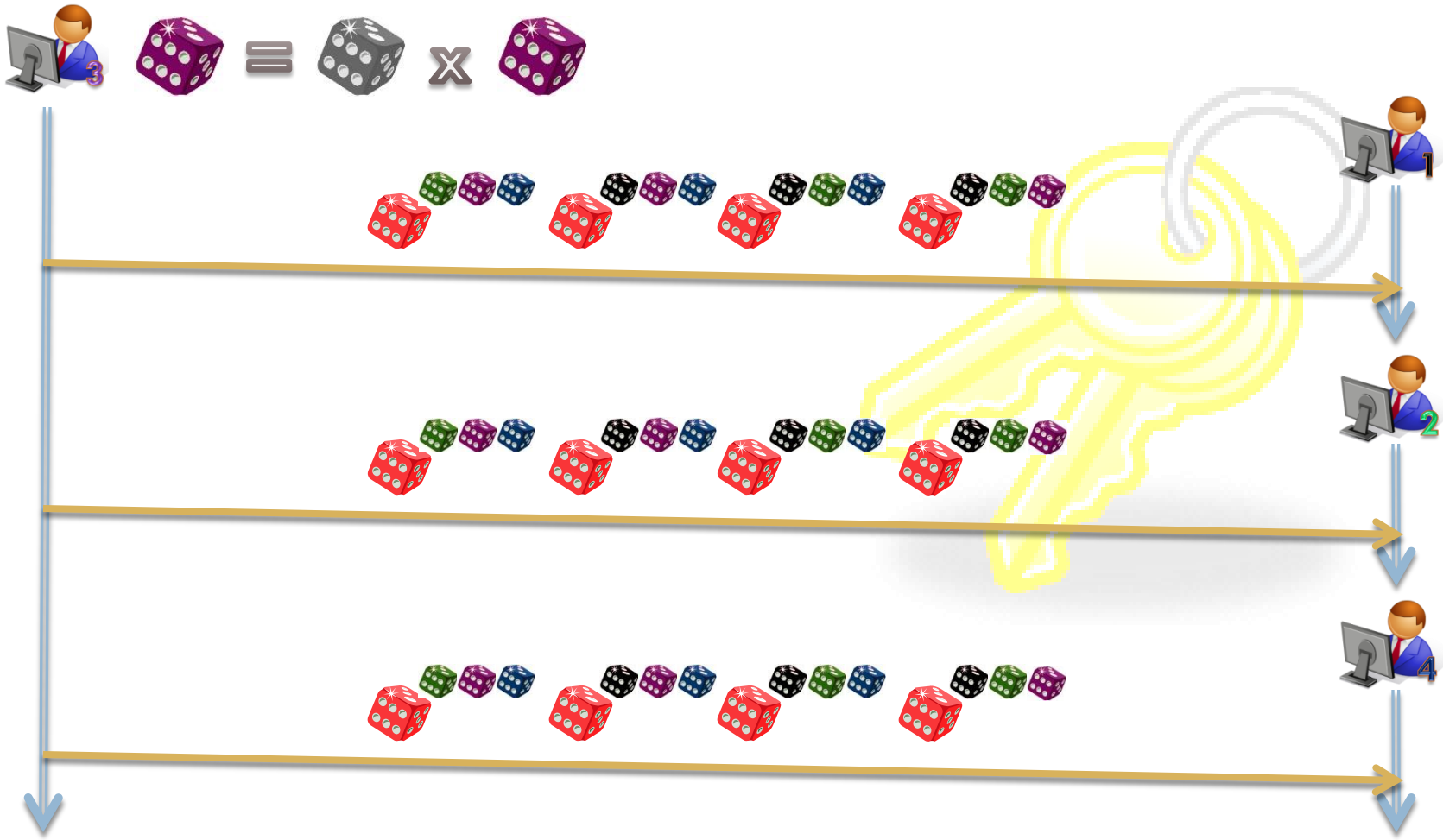


Key Refresh Operation

- Da jedes Mitglied die Broadcast-Nachricht gespeichert hat kann ihn jeder durchführen
- Man hält fest welches Mitglied wann seinen Anteil zuletzt erneuert hat
- Das Mitglied der seinen Schlüssel zuletzt erneuert hat, ist der Nächste für einen Refresh



Key Refresh Operation



AKA Sicherheitsbetrachtungen

- Es existieren folgende Entitäten:

 Alle Mitglieder, die ausgeschlossen wurden

 Alle momentanen Mitglieder

 Alle zukünftigen Mitglieder

 {Eve}

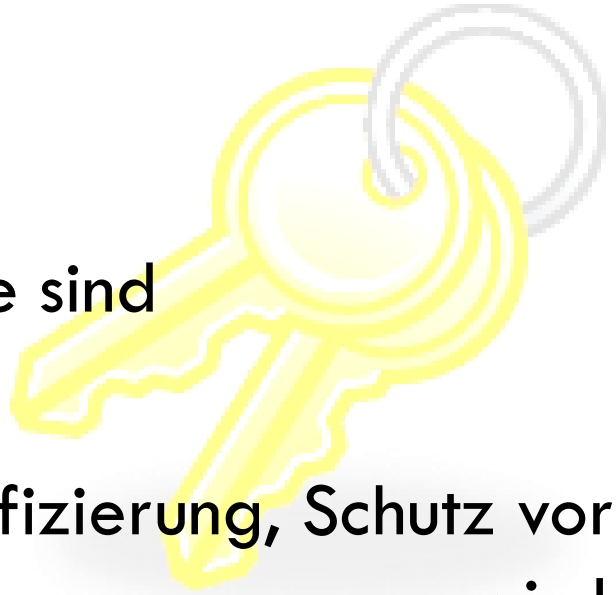
  - Eve‘s Ziel ist es  zu berechnen

AKA Sicherheitsbetrachtungen

- Nun kann Eve alle Nachrichten abfangen und weiß, dass der letzte Schlüsselbeitrag nur in der Broadcast Nachricht der Member-Exclusion bzw. Member Addition enthalten ist.
- Dann besitzt Eve maximal dieses Set:
- Problem:
  Den letzten Wert(Schlüssel)
  von einem Zufallswert
  zu unterscheiden

AKA Sicherheitsbetrachtungen

- Was wir aus dem IKA kennen
- Daraus folgt dass alle AKA Operationen GDDH-Probleme sind
- Zu beachten ist, dass Authentifizierung, Schutz vor Time-Attacks usw. als gegeben angenommen wird



Weitere Ansätze

- Ansatz von Steer[Str]:

  - Vorteil: Member-Addition gut geeignet
  - Nachteil: Member-Exclusion sehr aufwendig
- Protokoll von Burmester und Desmedt:
  - Key:

  - Vorteil: IKA Sehr effizient
  - Nachteil: Hohe Menge an Nachrichten, die zugleich gesendet und empfangen werden, was die Synchronisation komplex macht somit AKA ineffizient

VIELEN DANK FÜR DIE AUFMERKSAMKEIT

Dienersberger Maximilian