Circom

https://iden3.io/circom https://github.com/iden3/circom	**Programmeerbare zero-knowledge bewijzen**	
	Systeemvereisten:	Linux
	Ontwikkeld door:	Iden3
Open source / GPL-3.0 license	Contactpersoon:	Kristof.Verslype@Smals.be

Functionaliteiten

Zero-knowledge bewijzen (ZKPs) stellen in staat kennis te bewijzen zonder die kennis prijs te geven. De bewijzende partij creëert daartoe een bewijs dat door een verifiërende partij gevalideerd wordt. Een populaire use case is bewijzen dat je ouder bent dan 18 zonder je exacte geboortedatum prijs te geven.

De vandaag meest populaire technologie - vooral in de context van blockchain – zijn *zk-SNARKs* (Zero-Knowledge Succinct Non-Interactive Argument of Knowledge), wat toelaat compacte bewijzen te genereren die efficiënt te valideren zijn.

Circom is een raamwerk voor programmeerbare ZKPs. Het omvat een domeinspecifieke programmeertaal, compiler en tools om bewijzen te creëren en te valideren. Circom wordt voornamelijk gebruikt op blockchain netwerken. Het wordt onder meer gebruikt door privado.id, wat een infrastructuur op Ethereum aanbiedt om selectief persoonsgegevens over jezelf prijs te geven. Andere projecten die op Circom beroep doen zijn Tornado Cash – om op Ethereum de herkomst van je virtuele munten te verbergen – en Semaphore – een anoniem protocol voor verkiezingen en klokkenluiders.

Circom is een programmeertaal voor zk-SNARKs. Dit type ZKPs hebben het nadeel dat ze een vertrouwde setup vereisen en niet resistent zijn tegen cryptografisch relevante kwantumcomputers. De recentere, voorlopig nog minder populare zk-STARKs (zero-knowledge Scalable Transparent Argument of Knowledge) hebben deze nadelen niet, zijn schaalbaarder maar resulteren in grotere bewijzen. Voor zk-STARKs worden andere domeinspecifieke programmeertalen zoals Cairo gebruikt.

Conclusies & Aanbevelingen

De installatie, net zoals het creëren en valideren van eenvoudige ZKPs ging vrij vlot, dankzij de tutorial. Om effectief systemen te bouwen die het niveau van de experimenten overstijgt is echter heel wat kennis vereist.

Testen & Resultaten

We hebben drie testen uitgevoerd m.b.v. Circom.

1) Gegeven een getal *c*, bewijzen we dat we twee priemfactoren a en b kennen, zodat *a* x *b* = *c*. Men veronderstelt dat een dergelijke factorisatie voor grote priemfactoren onhaalbaar is m.b.v. klassieke computers. Men maakt heel vaak gebruik van deze aanname in moderne cryptografie.
2) Gegeven een waarde *h* van 32 bytes, bewijzen we dat we een waarde *v* kennen, zodat *h* ← f(*v*), waarbij f de cryptografische hashfunctie SHA-256 is. Het berekenen van de inverse van f is in de praktijk onhaalbaar, indien *v* voldoende entropie bevat en dus niet via brute-force te achterhalen is.
3) Gegeven een leeftijd, bewijzen we dat we meerderjarig zijn. Bewijzen van meerderjarigheid is een typische use case voor ZKPs in de context van identiteitsbeheer.

```
1   pragma circom 2.0.0;
2
3   template Multiplier2 () {
4
5      // Declaration of signals.
6      signal input a;
7      signal input b;
8      signal output c;
9
10     // Constraints.
11     c <== a * b;
12  }
```

Figuur 1. Circom circuit om te bewijzen dat je gegeven *c*,waardes *a* en b kent zodat *a x b = c*

De bovenstaande code – men noemt ze doorgaans *circuits* – worden bij compilatie omgezet naar een reeks wiskundige vergelijkingen. Als onderliggend zk-SNARK protocol werd met Groth16 gewerkt. Op basis van die wiskundige vergelijkingen creëert Groth16 dus een bewijs dat eveneens met Groth16 geverifieerd kan worden.

Figuur 1 toont ons eerste circom circuit. Enerzijds zijn er de *signals* (private invoer en te bewijzen uitvoer) en anderzijds zijn er de *constraints*; de relaties tussen deze signals die altijd geldig moeten zijn.

Voor testen 2 en 3 werd beroep gedaan op voorgedefinieerde circuits; bouwblokken die beschikbaar zijn in circomlib. Dergelijke voorgedefinieerde circuits zijn het equivalent van functies in imperatieve programmeertalen.

Het genereren van een bewijs in test 3 duurt ongeveer 1 seconde, het verifiëren van het bewijs slechts een tiental milliseconden. De grootte van het bewijs was minder dan 400 bytes. In meer complexe settings, zoals leeftijdsverificatie op basis van de huidige eID kaart, wat recent door de VUB gerealiseerd werd, hebben zk-SNARKs 22 seconden nodig om een bewijs te genereren en 0,2 seconden om het te verifiëren. Verifiëren is in het algemeen sneller dan bewijzen, maar de tijd nodig voor elk van deze operaties verschilt sterk naargelang het circuit.

Gebruiksvoorwaarden & Budget

Circom is open source onder de GPL-3.0 licentie.